ALMA MATER STUDIORUM – UNIVERSITY OF BOLOGNA

Department of Classical Philology and Italian Studies

Second Cycle Degree Program in

Digital Humanities & Digital Knowledge

Named Entity Recognition: Progression from Traditional Models to Large Language Models and Emerging Perspectives.

Final dissertation in:

Natural Language Processing

Supervisor
Prof. Fabio Tamburini

Co-Supervisor
Prof. Silvio Peroni

Defended by
Anita Liviabella

Graduation Session: 2
Academic Year 2023/2024

INDEX

ABSTRACT 4

INTRODUCTION 5

1. DEFINITION AND EVOLUTION OF NER 7

1.1 Task introduction 7

1.2 Evaluation methods 14

1.3 Datasets and classic benchmarks 19

2. FROM RULE-BASED TO MACHINE LEARNING 21

2.1 Rule-based and knowledge-based approaches 21

2.2 Overview of ML approaches: broad subdivision of machine learning approaches, workflow, and types of models. 25

2.3 Supervised Machine Learning Systems: Hidden Markov Models, Maximum Entropy Models, combined learning systems, and other techniques. 27

2.4 Semi-supervised Learning: key techniques, further innovations, advanced techniques. 33

2.5 Unsupervised Machine Learning Systems: early approaches, statistical methods, recent developments, and evaluation challenges. 34

3. DEEP LEARNING AND FINE-TUNING TRANSFORMERS 36

3.1 Overview 36

3.2 Deep learning and transformers for NER: reasons for applying deep learning in NER tasks, deep learning steps and popular approaches. 45

4. LARGE LANGUAGE MODELS, PROMPTING AND EMERGING PERSPECTIVES. 51

4.1 Introduction 51

4.2 The behavior of large language models and prompting on the NER task 53

4.3 Ethical considerations: data protection, privacy risks, and biases..... 62

4.4 The process of research through a bibliometric analysis: research hotspots, and low resource NER. 65

CONCLUSION 68

BIBLIOGRAPHY 70

SITOGRAPHY 80

ABSTRACT

This thesis describes Named Entity Recognition (NER) a fundamental task for Information Extraction (IE) in the field of Natural Language Processing (NLP).
The structure of the paper is divided into four main chapters, respectively regarding the task introduction, the advent of Machine Learning (ML) systems, the Deep Learning (DL) revolution, and the recent advancement of Large Language Models (LLMs). All the different approaches have been described and analyzed to understand their meaningful impact on NLP and consequently on NER.
The thesis follows a chronological order, to better allow the comprehension of the process of research, underlying the most important hotspots and trends, as well as the path in building more sophisticated systems for named entity recognition, underlying their strengths, weaknesses, and challenges.
The paper aims to be a resource for those interested in the topic, describing a detailed reconstruction of the task evolution and offering an updated picture of the current scenario.

INTRODUCTION

Named Entity Recognition (NER) is a relatively old and traditional task in Natural Language Processing (NLP); however, it has gained popularity again for its importance in Information Extraction (IE), especially with the development of Large Language Models (LLMs) in Artificial Intelligence (AI). The purpose is to detect and classify named entities from unstructured texts. These can be names of people, locations, and organizations, but also predefined categories specific to domains such as healthcare, finance, social media, and many others. Named entity recognition is a pre-step for different NLP applications like machine translation, question answering, text summarization, and semantic annotation.

The goals of this paper are to describe the historical evolution of the task, analyze the more possible different methodologies, and discuss the modern directions of the research in this field. The main questions are how the evolution of the methodologies influenced the performance in terms of efficiency but also how to address the problem of data scarcity, a predominant theme in NLP.

This thesis explores NER literature, from the definition of the task to the emerging perspectives of research. The work is divided into four chapters; the first one aims to broadly introduce the task, tracing its historical origins, and generally describing its approaches, utilities, and application fields. Moreover, it explains the evaluation techniques, metrics, and strategies for named entity recognition and the classic datasets and benchmarks that have been mostly used to train and evaluate the task in many different languages, domains, and different types of entities.

The following chapters are centered on the main techniques for NER; The second chapter introduces the subdivision of the rule-based and machine-learning (ML) systems used in natural language processing. Then it goes into the specificities of their approaches and evolution over the named entity recognition task. It describes the most popular supervised ML systems like Hidden Markov Models (HMM), Maximum Entropy Models (ME), Support Vector Machines (SVM), and Conditional Random Fields (CRF) underlying their strengths and weaknesses. It explores semi-supervised learning and unsupervised learning techniques as well, to understand which kind of systems may be appropriate in resource-scarce scenarios, where the lack of labeled data is limited.

The third chapter presents the concepts of deep learning and the impact of transformers on NLP and consequently in NER. It follows a similar structure to the previous chapters, discussing the most popular and innovative neural network architectures. The revolution of deep learning allows the system to learn complex patterns from data, reducing the necessity for manual feature engineering. The most used architectures are Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN). Moreover, the chapter introduces the concept of

attention and self-attention which characterize transformers, which can be pre-trained on large text corpora and then fine-tuned on specific NER tasks, dominating the state-of-the-art in NER.

Another goal for the thesis was to explore and understand the role of Large Language Models (LLMs) in NER. For this purpose, the last chapter explores the role of Large Language Models (LLMs), specifically analyzing the most recent prompting approaches. Furthermore, the chapter gives space to ethical considerations, and it reconstructs the process of NER's research and the modern directions and challenges.

1. DEFINITION AND EVOLUTION OF NER

1.1 Task introduction

Named Entity Recognition (NER) stands as a subtask within the field of Information Extraction (IE), a cornerstone of language technology aimed at distilling structured data from unstructured or semi-structured sources, such as electronic documents, to convert the raw material into machine-readable format, often databases (Wilks 1997). NER is comprehended in Natural Language Processing (NLP)' tasks, a field of Artificial Intelligence (IA), in which the main purpose is for machines to understand, interpret and generate human language.

Historically speaking, NER origins date back to the early 1990s, with Lisa F. Rau proposing company names' identification within the text, pointing out their importance in information extraction and NLP tasks in general (Rau 1991). From 1987 onwards, the Defense Advanced Research Projects Agency (DARPA)[1] spearheaded the Message Understanding Conferences (MUC)[2] to standardize the evaluation of IE systems, by examining common tasks. These events focused on the English language and aimed to analyze unstructured texts, identifying specific events to populate the database with the retrieved information. In November 1965 the Named Entity (NE) shared task was introduced at the sixth MUC conference. The NE task at MUC-6 comprised three subtasks: entity names, temporal expressions, and numeric expressions[3]. These expressions served as unique identifiers for entities (such as organizations, persons, and locations), times (dates and times), and quantities (monetary values and percentages). Recognized as distinct textual features with semantic significance, these identifiers were fundamental for scholarly purposes and database creation from textual data (Palmer 1996).

Adhering to the Text Encoding Initiative (TEI) guidelines was important for ensuring the smooth recognition and annotation of various expressions within specific text sections marked by SGML tags. The Standard Generalized Markup Language (SGML) is a non-proprietary declarative meta-language, which supports readable, generic, structural, and hierarchical markup.[4] Temporal expressions (TIMEX), numeric expressions (NUMEX), and entity names (ENAMEX) were annotated using the pertinent tags, with a TYPE attribute specifying the subcategory of each expression. TIMEX phrases included both date and time expressions, while NUMEX phrases covered

[1] https://www.darpa.mil.
[2] https://dl.acm.org/conference/muc.
[3] https://cs.nyu.edu/~grishman/muc6.html.
[4] http://www.cs.unibo.it/~fabio/corsi/sdc/SGMLintro.pdf.

percent and money expressions. ENAMEX phrases, on the other hand, denoted proper names referring to persons, locations, and organizations (Grishman 1996) (Fig. 1).

```
Mr.  <ENAMEX TYPE="PERSON">Dooner</ENAMEX> met with <ENAMEX TYPE="PERSON">Martin
Puris</ENAMEX>, president and chief executive officer of <ENAMEX
TYPE="ORGANIZATION">Ammirati & Puris</ENAMEX>, about <ENAMEX
TYPE="ORGANIZATION">McCann</ENAMEX>'s acquiring the agency with billings of <NUMEX
TYPE="MONEY">$400 million</NUMEX>, but nothing has materialized.
```

Figure 1: Sample sentence with named entity annotation (Grishman 1996).

Named entity recognition is a sequence tagging problem, a broader category of tasks encompassing chunking and Part-of-Speech Tagging (POS tagging) (Tamburini 2022). In NLP sequence, tagging breaks down the input language into smaller units and assigns labels based on different grammar, semantic, or linguistic features. Chunking involves the recognition and assembly of sequences of words carrying syntactic elements (chunks). Similarly, POS tagging involves assigning a part-of-speech label to every word within a sentence. Both tasks are important bases for other applications like syntactic parsing, automatic translation, sentiment analysis, etc.

Practically speaking, as a sequence labeling problem, NER involves a series of systematic steps, such as tokenization, which breaks down the text into manageable units like words or multi-word expressions. In this way, potential named entities are spotted using linguistic rules or statistical methods, for instance, capitalization cues or specific formats aid in recognizing entities such as names or dates. Moving forward follows a proper identification step with the application of specific techniques, in which entities are classified into predefined categories like “person”, “organization” or “location” (Fig. 2). Context is fundamental to assure accuracy, NER systems are trained to detect the proper meaning of entities. Lastly, post-processing steps are meant to resolve ambiguities, to refine and enrich results.

The development of NER systems continues to evolve, and over the years different methods have been thought to accomplish its challenge, therefore each technique has its strengths and limitations. Rule-based methods serve as a foundation, relying on a collection of manually crafted rules, linguistic patterns, regular expressions, or dictionaries, which specialists have carefully designed to execute the method. Rules are meant to be domain-specific; they are effective in specific domains with well-defined entities, but they struggle with scalability when faced with diverse or large datasets due to their rigidity.

The transition from manual rules has been helped by statistical methods, which predict named entities based on likelihoods derived from training data, by tagging words in a test corpus based on the annotations present in the training corpus. Indeed, supervised learning techniques (SL) comprehend Hidden Markov Models (HMM), Conditional Random Fields (CRF), Support Vector Machines (SVM), Decision Trees, and Maximum Entropy Models (ME). These models typically involve the ingestion of large, annotated corpora, which may be computationally demanding.

Systems that involve a small degree of supervision are semi-supervised machine learning systems (SSL), which involve a main technique called "bootstrapping". They employ a combination of labeled and unlabeled data, to enhance model performance, the learning process involves using a set of rules on a small, labeled dataset, to discover extraction patterns based on relevant markers. These kinds of methods generally offer advantages in low-resource scenarios.

Due to the lack of data labeling, the common approach for unsupervised learning is clustering, by gathering named entities from clustered groups based on the similarity of the context.

The advent of deep learning changed radically the scenario, exploiting the power of neural networks such as Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), and transformers. Especially transformers models, like BERT (Bidirectional Encoder Representation from Transformers), have emerged as powerful tools for NER tasks. BERT, along with its derived models, is pre-trained on large text corpora to handle long-distance dependencies successfully. Transformers use a self-attention mechanism to weigh the importance of each word in a sequence, based on its relationship with all the other words in it. Conversely, RNNs have difficulty capturing long-range dependencies, information from earlier time steps in the sequence may not be propagated effectively through the network, influencing later time steps. This happens similarly for CNNs since they operate on a fixed size of input data.

State-of-the-art results in NER are mostly accomplished by BERT-derived models or hybrid approaches. Plus, Large Language Models (LLMs), including models like GPT, employing original user interfaces for text generation, have demonstrated progress in transforming the NER task into a text generation task, employing prompting and Few-shot learning.

Concerning low-resource NER, techniques like Transfer Learning, Data Augmentation, Active Learning, Few-shot learning, and Zero-shot learning have been implemented (Keraghel et al. 2024).

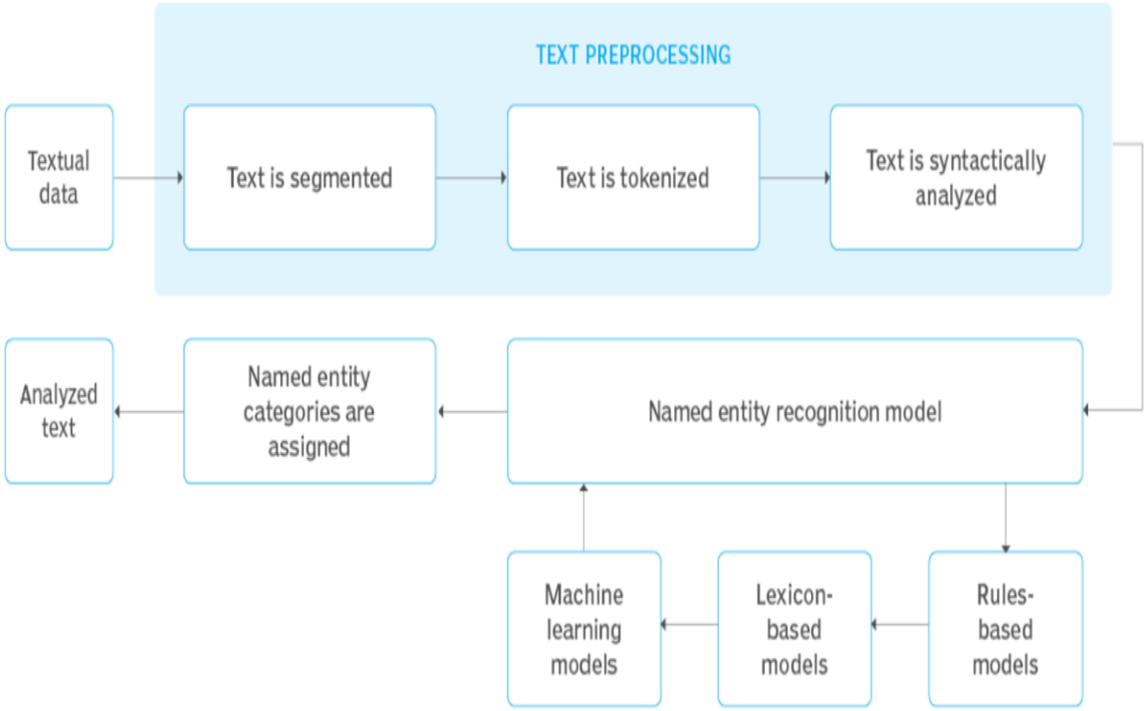


Figure 2: Named entity recognition workflow.[5]

To test and accomplish goals using these methodologies, evaluating the systems is a mandatory step to achieve goals in NLP tasks. Therefore, NER evaluation refers to the "gold-standard" format, involving the development of annotation guidelines, creation of a gold-standard corpus, validation of inter-annotator agreement, and the implementation of a scoring tool to compute selected metrics such as precision and recall. Consequently, the development of NER benchmarks has led to the creation of datasets, involving events such as MUC, Automatic Content Extraction (ACE)[6], the shared task at the Conference on Computational Natural Language Learning (CoNLL)[7], and Evaluation of NLP and Speech Tools for Italian (EVALITA)[8], and many others, which have contributed to the advancement of NER techniques, by providing annotated corpora, evaluation platforms, different strategies, and metrics. Different languages, domains, types of annotations, and categories of named entities are covered by these benchmarks.

In addition, NER is a fundamental base for other NLP tasks like Question Answering (QA), Machine Translation (MT), text summarization, and semantic annotation. For instance, QA systems, involving systems that give automatic responses to answered questions, require the detection of

[5] https://www.techtarget.com/whatis/definition/named-entity-recognition-NER
[6] https://www.ldc.upenn.edu/collaborations/past-projects/ace.
[7] https://www.conll.org.
[8] https://www.evalita.it

named entities. In essence, NER helps in identifying relevant information that can be used to formulate query questions effectively. Indeed, the quality of NER directly affects the system's ability to find named entities in sentences, which are essential for constructing accurate answers and facilitating their selection. Therefore, in TREC-8 (Text Retrieval Conference) about 80% of the queries used to evaluate this kind of system were of the type who, where, and when, which used to be answered with named entities of type, person, organization, location, and date (Marrero et al. 2013).

The utility of NER extends in MT systems, used for automatically translating text from one language to another, preserving its meaning.[9] Named entity recognition improves the quality of the translation, for instance, machine translation systems often mistranslate or drop named entities, this is an issue because it impacts the readability of translated sentences. NER leads to more accurate translation since helps the precise identification of NEs and their appropriate translation. Moreover, treating NEs differently from regular words during the process of translation ensures coherences of sentences in the result text.

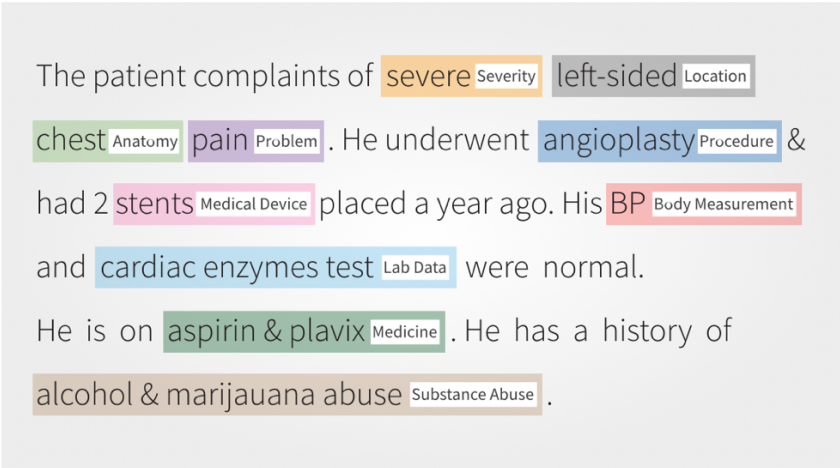
NER is a resource also for text summarization, to give readers pertinent information about the input text in a short amount of time. However, extractive summarizers can only anticipate topics from the original material, so it’s not so effective, even if the generated summary is syntactically correct. On the other side, for semantic annotations, which is the process of tagging documents with relevant concepts (Rahman et al. 2021), NER is a base for automating the task, instead of relying only on manual annotations, it is essential for scaling the process across large documents collections, increasing scalability. Consequently, extracted entities may then be ulteriorly processed and linked to concepts in an ontology or knowledge graph. Moreover, the Semantic Web is built on ontologies to enhance the interoperability of information. Data in ontologies are structured and organized to let computers understand them, however, the manual work in building them is not feasible. NER participates in terms of the automatization identifying the instances of concepts required for building the ontology (Marrero et al. 2013).

Named entity recognition’s application covers different domains: from social media analysis to biomedical research, finance, and cybersecurity, demonstrating its versatility. For instance, in the proliferation of social media, NER plays a role in extracting information from unstructured texts, such as tweets, which are interesting for sentiment analysis, to analyze significant opinions on products or political matters, since people on social media are free, or almost free, to express their idea on specific topics.In biomedical research, even if there are still open challenges due to the complexity of the medical lexicon and the scarcity of label datasets, NER models have been trained to identify entities

[9] https://nlp.stanford.edu/courses/cs224n/2010/reports/singla-nirajuec.pdf

like chemicals, diseases, and proteins from unstructured texts, facilitating research in healthcare and drug discovery (Jehangir et al. 2023).

Lately, there have been many new tools for Named Entity Recognition (NER). OpenAI[10] uses its GPT models for this. SpaCy[11], which is popular with Python users, can do many NLP tasks like breaking down text, finding parts of speech, and identifying named entities. NLTK[12] is another popular Python library for natural language processing. It has many text samples and flexible algorithms, making it easy to customize. Stanford CoreNLP[13], a Java-based tool, is good at breaking down text and recognizing named entities, but it supports fewer languages. Apache OpenNLP[14] focuses on recognizing various types of named entities, which improves its accuracy. Polyglot, also Java-based, supports over 40 languages, making it very versatile. Flair is an open-source tool that supports many languages and uses advanced models like BERT (Devlin et al. 2019) and ELMo (Peters et al. 2019). Hugging Face[15] offers many pre-trained models and a platform for sharing and improving models. Gate is a Java tool with a system called ANNIE that recognizes many types of entities in different languages. TNER[16], a Python library based on PyTorch, focuses on training and fine-tuning models. There are also tools like OpenNLP's R package[17] and the spacyr package that help integrate these libraries with other programming languages, making them more accessible (Keraghel et al. 2024).



[10] https//openai.com
[11] https://spacy.io.
[12] https://www.nltk.org
[13] https://github.com/stanfordnlp/CoreNLP.
[14] https://opennlp.apache.org
[15] https://huggingface.co
[16] https://huggingface.co/tner.
[17] https://www.r-project.org.

Figure 3: Example of NER in clinical notes, where a large quantity of unstructured information is contained. The classifications of named entities allow the transition from unstructured to structured and organize data.[18]

[18] https://it.shaip.com/solutions/clinical-named-entity-recognition-ner/

1.2 Evaluation methods

An objective evaluation of the system's performance is mandatory to attest to the progress, quality, and utility of NLP methodologies and to draw the evolution of certain tasks in the field, which is fundamental for researchers. The methodologies can be tested and categorized based on the type of results produced by the application of the system under evaluation. A standardized evaluation is fundamental to present an automatic tool for the system, and additionally, to attest to how much the performance is fast in achieving goals and how to proceed to obtain better results in the future. Many international events have been organized to deal with these challenges, discovering some of the most used datasets for NER. Such events allow researchers to be aware of progress, moreover, they are based on collaboration within the research community in which participants share resources and techniques, to innovate the NLP scenario.

Generally, the analysis starts by dividing the type of system. First, there are systems such as NER, part-of-speech tagging, parsing, etc. that take language input and generate abstract representation or classification. On the other side, other systems produce language output as translation and summarization. Lastly, there are interactive systems such as conversational agents and chatbots.

The evaluation of NLP systems sticks to the employment of two approaches, which are extrinsic and intrinsic evaluation. The first one accesses the system quality indirectly, integrating it into downstream tasks, while the second one is a direct analysis of the results against a set of correct examples, utilizing annotated corpora as benchmarks. Moreover, evaluation requires metrics and the most used are accuracy, error rate, precision, recall, and F-score. The metrics interpretation helps in assessing correct detection, insertions, deletions, and overall system performance. Multiclass averaging techniques such as macro-averaging and micro-averaging are employed to handle class imbalances (Tamburini 2022).

For recognition and classification problems such as NER the evaluation sticks to the "gold-standard" format. Gold-standard-based evaluations involve various steps, in the first place, the development of annotation guidelines, to specify how and what to annotate. Then, the development of a gold-standard corpus and the validation of the gold-standard inter-annotator agreement to finish the development of a scoring tool that computes the selected metric.

As already specified, named entity recognition belongs to the "chucking" class of tasks, the goal is for the system to identify and classify a fragment of text belonging to a specific category.[7] Since evaluation required an annotation scheme, specifically for NER these are used to detect the type and the position of named entities in the sequence of tokens. The BIO/IOB format, first

introduced in 1995 (Ramshaw et al. 1999), is a widely used convention that is employed to annotate tokens in text fragments and it's a base for the identification of entities. It stands for "Inside", "Outside" and "Beginning", respectively, the tokens prefixed with "B-" denote that the tag is the beginning of a chunk, while tokens labeled with "I-", indicate the inside of a chunk, lastly, the "O" tag signifies that a token does not pertain to any entity. The "B-" tag is used only when a tag is followed by a tag of the same type without "O" tokens between them (Tamburini 2022). The BIO/IOB schema provides a delineation of entity bounding, the choice of the tag scheme involves NER performance, outperforming one scheme over another, depending also on the language in question and from the data.

Words	IO	BIO	IOE	IE	IOBES	BIES
Emma	I-PER	B-PER	I-PER	I-PER	B-PER	B-PER
Charlotte	I-PER	I-PER	I-PER	I-PER	I-PER	I-PER
Duerre	I-PER	I-PER	I-PER	I-PER	I-PER	I-PER
Watson	I-PER	I-PER	E-PER	E-PER	E-PER	E-PER
was	O	O	O	I-O	B-O	B-O
born	O	O	O	I-O	I-O	I-O
in	O	O	O	E-O	E-O	E-O
Paris	I-ORG	B-ORG	I-ORG	I-ORG	S-ORG	S-ORG
.	O	O	O	I-O	B-O	S-O

Figure 4: Examples of annotation schemes. (Keraghel et al. 2024)

Common schemes other BIO than are IO, IOE, IOBES, IE, and BIES. They are all based on BIO principles with some additions or variations. For instance, the IO scheme is the basic approach, however, this notation septically struggles to distinguish consecutive entities of the same type. Moreover, the IOE scheme marks the end of entity "E" instead of the beginning. IE functions akin to IOE but differs in labeling non-entity word boundaries. (Fig.4) On the other side, IOBES is a more detailed version, it uses "S" to denote a single token entity. BIES is an extension of IOBES, it includes tags for non-entity words as well.

The second step of the format involves developing an annotated benchmark corpus, divided into three parts: training, validation, and test set. The first one contains the largest amount of annotated data, the validation set is used to adjust parameters and the test set to effectively evaluate the performance of the system. (Keraghel et al. 2024).

The organizer of the evaluation distributes the annotated training corpus as well as the task definition guidelines and the scoring tool, to let developers build, adapt, and tune their systems to perform the task. Finally, on a defined date the test set is distributed without annotations, consequently, in a few days the participants must return the output of their systems. In the end, all the outputs of the different systems are compared with the correct test set annotations using the established metrics, and the final ranking is drawn up based on the performance of the systems (Tamburini 2022).

While doing the comparison between two systems, there are two types of strategies: the exact evaluation and the relaxed evaluation. In the first one, the boundaries of the named entity and its corresponding type must precisely match the gold standard. On the other side, the relaxed evaluation is based on a scoring system where any entity with the correct type earns a certain type of points, even if the contours may not be correct. Entities may also earn points where the contours are correct, but not the type (Fig. 5)

As for other NLP tasks, classical metrics such as precision, recall, and F1-score are often used for NER evaluation. In detail, precision corresponds to the proportion of named entities successfully recognized by a model concerning the total number of named entities. Recall is about the number of relevant named entities retrieved by a model against the total number of relevant named entities. Lastly, the F1-score reflects a model's ability to effectively detect named entities, with a trade-off between precision and recall (Fig. 6). Additionally, the macro-average corresponds to the mean of the metrics computed for each class separately, while the micro-average gives equal weight to each sample (Keraghel et al. 2024).

English data	Articles	Sentences	Tokens	LOC	MISC	ORG	PER
Training set	946	14,987	203,621	7140	3438	6321	6600
Development set	216	3,466	51,362	1837	922	1341	1842
Test set	231	3,684	46,435	1668	702	1661	1617

Number of articles, sentences, tokens and entities (locations, miscellaneous, organizations, and persons) in English data files.

German data	Articles	Sentences	Tokens	LOC	MISC	ORG	PER
Training set	553	12,705	206,931	4363	2288	2427	2773
Development set	201	3,068	51,444	1181	1010	1241	1401
Test set	155	3,160	51,943	1035	670	773	1195

Number of articles, sentences, tokens and entities (locations, miscellaneous, organizations, and persons) in German data files.

Figure 5: CoNLL-2003 dataset (Sang et al. 2003).

English	precision	recall	F
[FIJZ03]	88.99%	88.54%	88.76±0.7
[CN03]	88.12%	88.51%	88.31±0.7
[KSNM03]	85.93%	86.21%	86.07±0.8
[ZJ03]	86.13%	84.88%	85.50±0.9
[CMP03b]	84.05%	85.96%	85.00±0.8
[CC03]	84.29%	85.50%	84.89±0.9
[MMP03]	84.45%	84.90%	84.67±1.0
[CMP03a]	85.81%	82.84%	84.30±0.9
[ML03]	84.52%	83.55%	84.04±0.9
[BON03]	84.68%	83.18%	83.92±1.0
[MLP03]	80.87%	84.21%	82.50±1.0
[WNC03]*	82.02%	81.39%	81.70±0.9
[WP03]	81.60%	78.05%	79.78±1.0
[HV03]	76.33%	80.17%	78.20±1.0
[DD03]	75.84%	78.13%	76.97±1.2
[Ham03]	69.09%	53.26%	60.15±1.3
baseline	71.91%	50.90%	59.61±1.2

German	precision	recall	F
[FIJZ03]	83.87%	63.71%	72.41±1.3
[KSNM03]	80.38%	65.04%	71.90±1.2
[ZJ03]	82.00%	63.03%	71.27±1.5
[MMP03]	75.97%	64.82%	69.96±1.4
[CMP03b]	75.47%	63.82%	69.15±1.3
[BON03]	74.82%	63.82%	68.88±1.3
[CC03]	75.61%	62.46%	68.41±1.4
[ML03]	75.97%	61.72%	68.11±1.4
[MLP03]	69.37%	66.21%	67.75±1.4
[CMP03a]	77.83%	58.02%	66.48±1.5
[WNC03]	75.20%	59.35%	66.34±1.3
[CN03]	76.83%	57.34%	65.67±1.4
[HV03]	71.15%	56.55%	63.02±1.4
[DD03]	63.93%	51.86%	57.27±1.6
[WP03]	71.05%	44.11%	54.43±1.4
[Ham03]	63.49%	38.25%	47.74±1.5
baseline	31.86%	28.89%	30.30±1.3

Figure 6: Example of results metrics (Sang et al. 2003).

1.3 Datasets and classic benchmarks

The interest in named entity recognition kept growing from MUC-6 onwards, throughout the years, scholars have curated NER datasets via conferences, initiatives, and contests, striving to propel the discipline forward and attain ever-higher levels of model precision. Since the inception of the first English-only NER dataset at MUC-6 in 1995, datasets have evolved along various dimensions, including language, domain, entity types, etc.

Over nearly three decades, both the development of NER datasets and the thorough examination of NER systems have transformed and evolved through time creating specific datasets for named entity recognition. Currently, approximately ten prominent conferences or initiatives feature NER tasks and offer datasets frequently utilized to train NER models. Examples comprehend MUC, MET (Merchant et al. 1996), IREX (Sekine et al. 2000), CoNLL (Sang et al. 2003), ACE (Doddington et al. 2004), GENIA (Kim et al. 2003, 2006), StemNet (Faessler et al. 2020), OntoNotes (Marcus et al. 2011), BioCreative V (Wei et al. 2016), WNUT (Strauss et al. 2016), and SemEval (Augenstein et al. 2017, 2018). Additionally, NER datasets address specific domains or languages, such as GENE-TAG (Tanable et al. 2005) (Ohta et al. 2009), BBN (Weischedel et al. 2005), WikiGold (Balasuriya 2009), WiNER (Ghaddar et al. 2017), NCBI-Disease (Lakshmana et al. 2018), N3 (Röder et al. 2014), SCI-ERC (Luan et al. 2018), and NNE (Ringland et al. 2019). Indeed, researchers have developed datasets to handle to niche domains, such as addressing the absence of NER datasets for the art domain by curating one based on digitized art historical documents (Jain et al.).

The predominant language in dataset creation is English, however, the MET conference in 1966 introduced multilingual NER, diversifying datasets with Chinese, Spanish, and Japanese. This last one gained more importance through the IREX conference. Similarly, CoNLL 2002 and CoNLL 2003 gave space to multilingual NER as well, working with Spanish, Dutch, and German corpora, rather than just English. Moreover, considering the ranking of the most spoken languages globally, commonly used NER datasets already cover a complete portion of the world's most frequently spoken languages. Moving forward, expanding the scope of NER dataset creation to cover more languages will be mandatory, this means not only adhering to mainstream languages but also following the linguistic needs of diverse research directions.

Generic domain datasets sourced from news-based corpora have traditionally been prevalent, these datasets typically feature familiar text and are relatively accessible. Datasets such as CoNLL predominantly utilized news-like texts. However, research expanded into domains like biomedicine, facilitated by projects such as GENIA, GENETAG, StemNet, NCBI-Disease, and BioCreative V, which regard biomedical corpora. The proliferation of NER datasets in the biomedical domain is

useful for new entity-type formulation and further annotation methodologies. Additionally, datasets retrieved from sources such as Wikipedia, WikiGold, and WiNER have been created to shift from news-based texts and explore other domains. In recent years, the growing interest in user-generated texts, from social media and website sources led to the creation of the WNUT dataset, which is particularly useful for identifying new types of named entities. Specifically, over time, additional entity types have been introduced to accomplish specific research needs, indeed, new entity types were continuously incorporated into NER tasks. For instance, IREX introduced the identification of the artifact (ARTIFACT) entity category, while CoNLL conferences advocated for the recognition of miscellaneous items (MISC), going beyond entities like PER, ORG, and LOC.

2. FROM RULE-BASED TO MACHINE LEARNING

2.1 Rule-based and knowledge-based approaches

Rule-based and knowledge-based methods established the fundamentals of named entity recognition approaches as for many other NLP tasks. Rule-based methods rely on manually crafted rules designed by linguistic experts, using domain-specific knowledge related to the domain and language. Knowledge-based methods benefit from dictionaries and lexical resources rather than annotated data. These kinds of systems may be accurate, but they have limitations due to the lack of flexibility and adaptability to other domains. The partial solution to this issue is to refine and extend those rules to enhance accuracy, gradually adding new concepts and relationships (Smith 1985).

Generally, rules are manually redacted, they collect meaningful sequences of words, and patterns, involving grammatical knowledge. In the named entity recognition task, lists of named entities are collected in gazetteers to confront candidate phrases with them. At a syntactical level, part of speech tagging is involved in disambiguating entities, while morphological rules are focused on prefixes, suffixes, compounding, etc. On the other side, orthographic pattern rules point out features such as capitalization patterns or punctuation, to detect potential entities. Lastly, context pattern rules select entities based on their surrounding word and phrases considered as their context.

Early experimental systems employed extensive dictionaries to identify personal titles and names from newspaper texts. Challenges mostly included overlaps between personal names and other lexical categories, misses, and false identifications, from words performing naming functions. Dictionaries and algorithms were adjusted with manual testing to obtain a more accurate system (Borkowski et al. 1966).

During the nineties, rule-based systems performed MUC evaluations, providing an overview of their limitations. Specifically, the NER task at MUC-6 evaluation had seen the system's application on the Wall Street Journal corpus. The requirement was to insert SGML tags into the text to mark and categorize names, temporal expressions, and numerical expressions. Fifteen teams participated, submitting a total of 20 system configurations. Half of the systems scored over a 90% F-measure overall, specifically, systems performed best in the identification of person names, followed by locations, having the most difficulties in detecting organizations' entities. Identifying dates was nearly perfect for most systems, while percentages presented occasional challenges. Furthermore, systems did better on the body text of articles than on the headlines. Plus, case sensitivity made the task easier, since the performance dropped significantly when dealing with upper-case-only versions.

The SRA (Krupka 1995) system achieved the highest score of 96.42% F-measure, making heavy use of manually compiled name lists and orthographic rules. This performance was ideal for this domain; however, the system lacked flexibility and adaptability beyond it, requiring manual extension of rules and resources. The system was unable to generalize when the case needed a deeper semantic analysis, for instance in distinguishing organization names from persons or locations, when there weren't case distinctions on the surface (Sunheim 1995).

In addition, the difficulties encountered by these methods were attested in the subsequent MUC-7 evaluation, in which the named entity task consisted of inserting SGML tags into the text to mark each string that represents a person, organization, location name, date, time stamp, currency or percentage figure. Many of the MUC-6 systems experienced a drop in named entity scores, for instance, the University of Sheffield system's FI score dropped from 89.06% in MUC-6 to 85.83 in MUC-7, highlighting a decrease of around 5%. One main reason was the shift to a new domain of newspaper articles about aviation accidents and air disasters in MUC-7, which exposed gaps in the lexical resources and rules that were tailored to the financial news domain of MUC-6. After adding relevant name entries and adjusting rules the named entity F1 score of the Sheffield system recovered to 90.41%. Errors regarded also the misclassification of certain entities as "person" instead of "organization" due to the inadequate disambiguation rules. Once again, the conclusion is that while adapting to new domains the systems require extended resources and updates (Gaizauskas et al. 1998).

The top-performing system for named entity recognition in the MUC-7 evaluation was The Language Technology Group (LTG) system. The main difference with other systems, which mostly involved gazetteers, was an approach that adopted a more context-driven perspective, with a pipelined architecture utilizing reusable text processing components that operated over XML/SGML annotations. These components included tokenizers, lemmatizes, taggers, and other tools that could exploit existing markup, remove annotations, or introduce new ones without altering the existing annotations. This enabled applying customized processing strategies and resources to different document sections via configuration. The system did not treat gazetteers as definitive sources but rather used them to identify "likely" named entity expressions. The real power came from analyzing the linguistic contexts to determine if a candidate was indeed a definite named entity instance or not.

At each stage, decisions were made by weighing the scores produced, while drawing evidence from other parts of the document's content and structure. A key challenge faced by the system was single capitalized words appearing only once or twice in the text without suggestive surrounding contexts, in such cases, the system's decisions were not reliable. However, the performance might be not the fastest one initially, but it demonstrated a hybrid approach, intersecting knowledge-based

symbolic pattern rules with statistical partial matching techniques in a document-centric manner, with a flexible, pipelined architecture for NER (Mikheev et al. 1998).

More recent approaches have attempted to combine other techniques to improve domain portability, for instance, drNER (Eftimov et al. 2017) is a rule-based method proposed for named entity recognition in the dietary domain. It extracts dietary information from evidence-based dietary recommendations, focused on entities like Food, Nutrient, and Quantity/Unity. The innovation is in the combination of terminological-driven NER, indeed, the system allows for the use of corpus-based dictionaries for some entities of interest, instead of using dictionaries that consist of concepts and synonyms. Moreover, using rules defined around entity characteristics can be time-consuming to develop, therefore, drNER employs just a small set of Boolean algebra rules to identify potential entity mention phrases. The system performs following a two-phase process of entity detection and then extraction based on syntactic analysis. It was evaluated on a test corpus of 100 documents from websites and publications, with reasonable performance, claimed to be the first focused NER method for the dietary domain not relying on an annotated corpus.

Additionally, the use of regular expressions is part of the rule-based strategies for NER. They can be employed to match complex strings using basic or complicated patterns. For instance, DrugNER (Korkontzelos et al. 2015) is a study aimed at providing the recognition of drug names in biomedical texts, the voting systems combine multiple heterogeneous models, which incorporate dictionary knowledge, a small gold-standard corpus, and automatically generated annotations. The system uses genetic programming to employ regular-expression patterns that recognize common drug name suffixes. For instance, the morphological similarities among drug names are fundamental in achieving high performance. The conclusion leads to the fact that high-quality gold-standard data is not strictly necessary for effective drug NER, combining models can give a classification performance comparable to the best models trained on extensive gold-standard data. The study suggests that effective drug NER can be achieved with reduced dependence on labor-intensive manual annotations by combining dictionary knowledge, small, annotated datasets, and advanced pattern recognition techniques.

As we've already seen, the main issue and challenge is the maintenance of rule-based systems. The advent of machine learning radically changed the scenario of natural language processing. In this case, using a machine learning model trained on annotation, from a rule-based system, allows to identify the areas where the system may need updates, by examining where the learned model disagrees with the rule-based system on new data. The idea is to use a rule-based NERC (Named Entity Recognition and Classification) system to annotate a large training corpus, by recognizing and classifying named entities in the text. The second phase involves training a separate system using a machine learning algorithm on the previously annotated data. In this way, no further manual

annotation is required. Both systems are then run on new data, the purpose is to examine the results in cases where they disagree, as this could indicate potential issues from the rule-based system which might require updating. This allows a language engineer to further identify and fix problems among the general rules. The method has been evaluated in Greek and French language corpora, and the evaluation shows that it can effectively assist in maintaining rule-based NERC systems by including machine learning techniques in this sense, without requiring manually annotated training data (Petasis et. Al 2001).

The evolution of these methods saw a growing tendency to develop hybrid systems including machine-learning components, that may be supervised, semi-supervised, or unsupervised. For instance, for Arabic named entity recognition, the combination of rule-based and ML methods allowed the optimization of the overall performance. A large dataset from multiple sources has been gathered, containing 23,928 annotated named entities across 11 different types, including lesser-studied categories like file names and phone numbers. The rule-based component employs 73 grammar rules and 90 gazetteers to handle the extraction, while the ML component comprehends different groups of features: rule-based, gazetteers, morphological information, part of speech tags, contextual information, etc. Three machine learning classifiers have been experimented with: decision trees, support vector machines (SVMs), and logistic regression. The results show that the hybrid approach outperformed the singular two approaches, with the decision tree at the top of the F1 scores (Oudah et al. 2012).

To conclude, rule-based methods are still employed, especially when high precision is required, for instance, NLP libraries like spaCy support these capabilities. Indeed, the library also allows the definition and the customization of rules, to extend their use and handle low-resource scenarios of specific domains.[19]

[19] https://spacy.io/usage/rule-based-matching

2.2 Overview of ML approaches: broad subdivision of machine learning approaches, workflow, and types of models.

Machine learning (ML), in the field of AI, started being recognized as a concept in the 1950s, while from the 1970s onwards, the term was coined to define algorithms able to learn from input data, that can be labeled or unlabeled, consequently making predictions or evaluations. These algorithms can generalize to unseen data and perform tasks without being explicitly programmed. The quality and quantity of the input data matter for the accuracy of the final model's performance. For clarity, a broad subdivision of ML methods includes supervised, semi-supervised, and unsupervised systems.

Supervised systems require data providing the correct solution to learn how to solve a problem, in simple terms, the algorithm tries to learn the model by copying the data. Once it can reproduce the examples, it is ready to annotate new data, hopefully with accuracy. In a supervised system, some points of the function are provided to let the system interpolate them, learn how to generalize from unseen data, and try to understand the best function through an iterative process, increasing precision. The training process can be seen as function approximation, where the goal is to induce an approximation f' of a target function f. The feasibility of obtaining a large enough number of instances, representative of the target function, determines whether supervised learning is appropriate for an application.

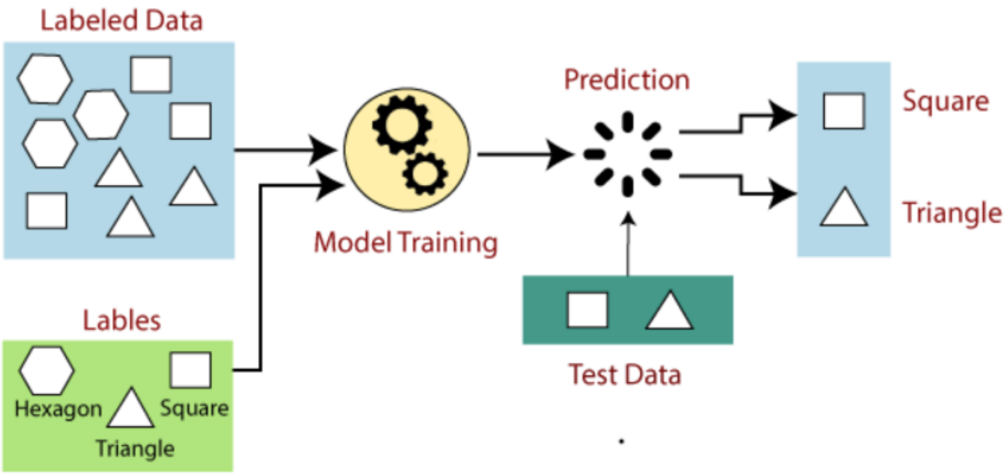


Figure 7: Supervised machine learning system's workflow.[20]

[20] https://www.javatpoint.com/supervised-machine-learning

The provided corpus is called a training set, and the values of instances are usually defined through manual annotation. Supervised learning is typically employed in classification tasks, where an instance is classified. Each instance is defined as a vector of linguistic features, formalized as a numerical vector. After some trial-and-error steps, the two classes are separated, and the system's classification performance on the training set is very high. The system is tested on a validation set to estimate performance on unseen data. If results are unsatisfactory, algorithm parameters must be changed. Once the performance is ideal, generalization abilities are estimated on a test set.

In supervised learning, data containing the correct solution must be provided for every task. On the other hand, unsupervised ML systems can look at data in raw form, without annotation, and still perform. These systems are generally clustering algorithms, capable of classifying similar and dissimilar instances, which are always represented as vectors compared by quantifying their similarity and introducing a distance measure between vectors. This allows the creation of groups of homogeneous elements.

There are two main kinds of clustering methods: hierarchical clustering and partitional clustering (Fig. 8).

In hierarchical clustering, given a set of instances, the algorithm proceeds by grouping the two most similar instances. Every time two instances are merged, they form a node in a tree, creating a cluster of instances. There are several ways of computing the centroid (middle point) of each group, necessary to compare groups. Following this procedure, a dendrogram tree is built, where the height of each branch is proportional to the similarity measure. The algorithm can group entities from the bottom-up or top-down, stopping once groups of very dissimilar instances are joined. Although inspecting the tree is complicated, measures like the silhouette measure can inspect it automatically.

Conversely, partitional clustering algorithms require the number of clusters as input. The algorithm then starts by randomly selecting centroids that represent the central point in the space of each class to be created. Being random, they must move and adapt to respect the data. Subsequently, the centroid is recomputed by averaging all instances in each group, and instances are reassigned to each centroid. Eventually, the algorithm finds the correct clustering in the space.

However, this was a main subdivision, but in Python packages such as scikit-learn[21], there are many other sophisticated clustering algorithms.

[21] https://scikit-learn.org/stable/

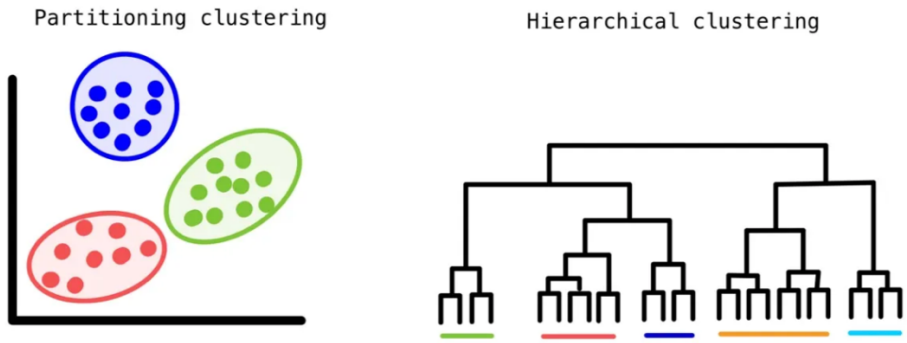


Figure 8: Partitional clustering and hierarchical clustering. [22]

2.3 Supervised Machine Learning Systems: Hidden Markov Models, Maximum Entropy Models, combined learning systems, and other techniques.

Early studies were based mostly on rule-based systems; for instance, five systems out of eight were rule-based in the MUC-7 competition. More recently, at the end of the nineties and early two-thousands, the machine learning component was incorporated to automatically induce the systems or sequence labeling algorithms to start from a collection of training examples. However, at the time, with the lack of training data, handcrafted rules still provided the best performance. Sixteen systems were presented at CoNLL-2003, where participants employed a wide variety of machine learning techniques, system combinations, and additional information beyond the available training data. This information included gazetteers, unannotated data, and outputs from externally trained named entity recognition systems (Sang et al. 2003).

The goal of supervised machine learning systems in the named entity recognition task is to indicate whether the data instances belong to a specific category (positive result) or not (negative result). Each named entity is tagged in the manually labeled documents to help the model learn the characteristics or attributes that differentiate positive examples from negative ones. The model then applies these learned features to unseen documents. However, these techniques require many annotated documents, making them resource-intensive and leading to exploring different solutions like semi-supervised and unsupervised machine learning systems (Nadeau et al. 2007).

One of the first SL techniques for NER is the use of Hidden Markov Models-based systems (HMM). These are statistical models used in NLP to model sequential data, where the observed data

[22] https://medium.com/@chyun55555/how-to-find-the-optimal-number-of-clusters-with-r-dbf84988388b

(tokens/words) are considered the visible states, while the underlying states are hidden (named entities tags). The sequence of observed data is generated by a Markov process, where the probability of transitioning to a particular state depends only on the current state and not on the previous states.

An HMM comprises three components learned from the training dataset: a matrix representing the transition probability from one state to another, another matrix representing the probability of emitting a particular output symbol from a given state (emission probability), and a vector stating the probability of starting in each state. These components are learned using algorithms like the Baum-Welch. An HMM is mathematically described by five parameters:

$$\text{HMM} = \{S, O, \pi, T_r, E_m\}$$

Where S refers to the number of hidden states (entity labels), 0 to the number of observations (tokens), π is the distribution of initial state probabilities. T_r is the transition probability matrix, and E_m is the emission probability matrix. The translation of the NER problem into an HMM problem is defined as:

$$P(S|O) = P(EN|T)$$

Given a sequence of observed tokens, the probability of identifying the sequence of EN (named entity tags) given T is equivalent to calculating the probability of obtaining the sequence of hidden states S (labels we want to infer), given the observations O (actual tokens we observe in the sequence T) (Keraghel et al. 2024).

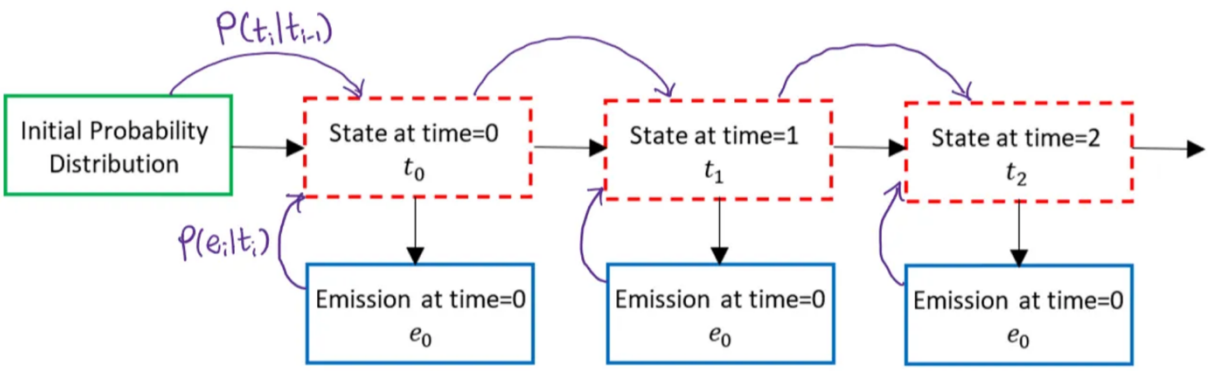


Figure 9: HMM process.[23]

[23] https://tanpengshi.medium.com/introduction-to-hidden-markov-model-hmm-with-simple-ner-d1353ff35842

As we can see in the provided diagram, the components of the HMM are the initial probabilities of starting at each hidden state (green box), the hidden states t_0, t_1, t_2 at each time step, which we aim to infer (red dashed boxes), the transition probabilities ($P(t_i|t_{i-1})P(t_i|t_{i-1})$) indicating the likelihood of moving from one state to another (arrows between the hidden states), lastly, the emissions (blue boxes) representing the observed tokens (e_0, e_1, e_2) at each time step, emitted from the hidden states with probabilities $P(e_i|t_i)P(e_i|t_i)$. Each state corresponds to a BIO tag, indicated in the BIO schema, previously discussed in the first chapter (e.g B-PER, I-LOC). Once trained, the HMM can predict named entity categories, from new unseen data using the Viterbi algorithm to find the most likely sequence of tags for a given sentence.

However, HMMs have limitations, such as the inability to model long sequences or complex patterns in text. The oldest HMM-based models for NER relied mostly on handcrafted gazetteers and simple characteristics like capitalization and digits to detect named entities. An HMM-based chunk tagger has been used to identify sequences of words as candidates for NE and then assign an NE type to each chunk (Zhou 2002).

A more recent HMM-based approach that is language-independent and domain-independent uses an annotated corpus in the target language and domain to estimate the HMM parameters in a completely data-driven manner. This approach allows for applicability across any natural language or domain by using annotated data for training and can be adapted to new languages and domains without additional linguistic resources or rules. However, the required annotations can be a high overhead (Morwal et al. 2012). Due to the limitations mentioned, these models have been surpassed by HMMs employed in unsupervised learning, which automatically learn word domain-specific patterns from large unlabeled text corpora. These patterns are integrated into the HMM framework using word similarity-based smoothing techniques to estimate probabilities better. (Zhao 2004).
Another statistical machine learning approach for NER tasks is the Maximum Entropy (ME) approach. At CoNLL-2003, ME was the most frequently applied technique, with five systems using it, either in isolation or in combination with other applications.

The maximum entropy approach for named entity recognition uses a probabilistic model to assign the most likely sequence of named entity tags to a given sequence of words. The procedure includes defining the probabilistic model, selecting features, training to maximize the probability of observed tag sequences, and employing the Generalized Iterative Scaling (GIS) algorithm. Adding a Gaussian prior is applied as a smoothing technique to prevent overfitting. Lastly, in the test phase, the Viterbi algorithm is used to find the tag sequence with the highest probability (Bender et al. 2003).

The ME model features are based on the surrounding context of the token, learning the weights associated with these features. The resulting probability distribution has the highest entropy and takes on the form:

$$p(o|h) = \frac{1}{Z(h)} \prod_{j=1}^{k} \alpha_j^{f_j(h,o)}$$

Where o refers to the outcome, h is the context, and $Z(h)$ is a normalization function. The constraints are usually derived from the training data, and the model aims to assign higher probabilities to outcomes that are more likely given the observed data (Keraghel et al. 2024).

However, some disadvantages related to ME models include the easy overfit of the training data, in the case of many features used, implying the necessity of regularization techniques that can complicate the training process, moreover, the quality of the features matters as well. Lastly, the optimization of multiple parameters can be computationally expensive (Bender et al. 2003).

At CoNLL-2003, the best performance for both English and German languages was obtained by a combined learning system that used Maximum Entropy Models, Transformation-Based Learning (TBL), Hidden Markov Models, and Robust Risk Minimization (RRM) (Florian et al. 2003).

TBL combines rule-based and statistical approaches to iteratively refine the output by applying transformation rules learned from a training corpus. This approach might be problematic if the number of rules increases, becoming very complex and difficult to interpret, making TBL less efficient for large datasets, since the iterative process of refining rules can be time-consuming. These systems may have difficulties in the capture of intricate patterns in data, compared to more sophisticated models. Additionally, RRM optimizes a risk function that measures the expected error of the model to prevent overfitting and improve generalization (Zhang et al. 2003).

Apart from the training data, the combined system also employed gazetteers and outputs from two externally trained named entity recognizers. It explored the performance of four distinct classifiers individually and in combination. These classifiers, including fnTBL, HMM, RRM, and MaxEnt, have different methodologies and characteristics. For example, fnTBL focuses on the most discriminative features, while the others consider all features of an example. The classifiers also handle data sparseness and sequence search differently, providing single classifications and probability distributions.

	English		German	
	(a)	(b)	(a)	(b)
HMM	82.0	74.6	-	-
TBL	88.1	81.2	69.5	68.6
MaxEnt	90.8	85.6	68.0	67.3
RRM	92.1	85.5	70.7	71.3

Figure 10: Individual classifier results on the two test sets (Florian et al. 2003).

The best results were achieved using the RRM combination approach, effectively integrating various information sources. Each classifier provides its classification for a given instance, and the final classification is determined by a majority vote and the probability distributions produced by each classifier. This framework demonstrates the effectiveness of classifier combination in handling NER tasks across different languages and conditions (Florian et al. 2003).

English devel.	Precision	Recall	$F_{\beta=1}$
LOC	96.59%	95.65%	96.12
MISC	90.77%	87.42%	89.06
ORG	90.85%	89.63%	90.24
PER	96.08%	97.12%	96.60
overall	94.26%	93.47%	93.87

English test	Precision	Recall	$F_{\beta=1}$
LOC	90.59%	91.73%	91.15
MISC	83.46%	77.64%	80.44
ORG	85.93%	83.44%	84.67
PER	92.49%	95.24%	93.85
overall	88.99%	88.54%	88.76

German devel.	Precision	Recall	$F_{\beta=1}$
LOC	83.19%	72.90%	77.71
MISC	83.20%	42.18%	55.98
ORG	83.64%	61.80%	71.08
PER	87.43%	67.02%	75.88
overall	84.60%	61.93%	71.51

German test	Precision	Recall	$F_{\beta=1}$
LOC	80.19%	71.59%	75.65
MISC	77.87%	41.49%	54.14
ORG	79.43%	54.46%	64.62
PER	91.93%	75.31%	82.80
overall	83.87%	63.71%	72.41

Figure 11: Results on the development and tests set in English and German (Florian et al. 2003).

Other supervised machine learning techniques for NER include Support Vector Machines (SVMs), Decision Trees (DT), and Conditional Random Fields (CRF).

Before Deep Neural Networks, SVMs were among the most successful ML systems due to their flexibility, quick training, and reliable results. SVMs can find the optimal linear classifier between two classes and use a specialized kernel to project the space into a higher-dimensional space. In NER, features such as part of speech tags, context, capitalization, and prefixes are used for training SVMs. SVMs showed strong performance metrics at CoNLL-03, demonstrating the ability to

generalize well from training data to unseen data. As with other SL methods, SVMs can be computationally expensive to train with large datasets, additionally, another issue might occur if the classification involves many classes (Mayfield et al. 2003).

Decision Trees, though less commonly used for NER, are a tree-like model of decisions and possible consequences. The decision tree is built by recursively splitting the training data based on features that best separate the classes. For classifying a new token, the feature values are used to traverse the tree from the root to a leaf node, determining the named entity class of the token. DT are vulnerable to overfitting, especially when the tree becomes very deep, this might lead to an incorrect generalization of unseen data, plus, small changes in the training data can lead to different tree structures, making them less stable compared to other techniques. Furthermore, this technique makes decisions based on individual features and does not capture dependencies between them (Sekine et al. 1998).

Conditional Random Fields (CRF) are probabilistic undirected graphical models used to predict sequences of labels for sequence data. They predict entity labels by considering the labels of neighboring tokens, reducing labeling ambiguity. A CRF model is defined as follows:

$$P(Y|X) = \frac{1}{Z_0} \exp\left\{ \sum_{t=1}^{T} \sum_{k=1}^{K} \lambda_k f_k(y_{t-1}, y_t, x, t) \right\}$$

Where Z_0 is the normalization factor for all possible sequences of states (labels), f_k are feature functions, each representing the occurrence of a specific combination of observations and associated labels, y_{t-1} is the label of the previous word, y_t is the label of the current word, x_t is the word at position $_t$ in the observed sequence. λ_k are the model parameters and can be interpreted as the importance of the reliability of the information provided by the binary function. Conditional random fields can handle the NER problem where the observations are the processed strings, and the labels are the named entities. Thus, finding the best sequence of named entities corresponds to a sequence of tokens in NER, which is equivalent to finding the best sequence of labels (Keraghel et al. 2024).

The model introduced at CoNLL-03 used a method called WebListing to augment lexicons with relevant terms, improving the performance of NER tasks. In general, CRF's training is computationally expensive on many parameters. Plus, the implementation and tuning of CRFs can be very complex (McCallum et al. 2003).

In supervised learning, different classifiers may vary across different output classes, specifically, some classifiers can better detect person names while others may be more accurate for location names. As already mentioned, the classifier ensemble technique has been considered to increase the performance of the system; to make this work, it is necessary to determine whether a

classifier should be allowed to vote for a particular class and quantify the amount of voting for each class by each classifier. This technique is called MultiObjective Optimization (MOO) and the results demonstrated that it enhances the performance of NER systems rather than the single performance of all individual classifiers. These methodologies are particularly effective for resource-constrained languages (Saha et al. 2013).

To conclude, we can affirm that supervised learning has been successfully applied to the named entity recognition issues, the techniques mentioned above can learn patterns from annotated training data and subsequently classify them. However, all the explained approaches cannot obtain an acceptable performance on large corpora, since the annotation process always requires manual intervention from experts, especially in the domain's specificity. Again, they cannot detect complex patterns and long-range dependencies and are particularly vulnerable to overfitting. Lastly, the combination of classifiers shows more accurate performances than the system used individually.

2.4 Semi-supervised Learning: key techniques, further innovations, advanced techniques.

Semi-supervised learning (SSL) can be considered as an interlude between supervised and unsupervised machine learning systems. The purpose of these systems is to use, in addition to labeled data, information available in unlabeled data, to reduce the reliance on manually annotated datasets and improve performance in low-resource cases. However, challenges include the potential propagation of errors from the initial data and the need for handling them on unlabeled data.

The key technique for SSL is “bootstrapping”, in which the user must provide to the system a set small number of examples, like personal names. Consequently, the system looks for sentences that might contain these names and examines the contextual features around them, which are common among the provided examples. Then, the system tries to find other instances of personal names that appear in similar contexts. The learning process is then reapplied to new examples, to discover new relevant contexts. Thanks to this process, many personal names and features describing the context are gathered (Collins et al. 1999).

Additionally, a very influential mutual bootstrapping technique is centered on iteratively expanding a set of entities and contexts, beginning with seeds of entities and the patterns found around them, which are then ranked to discover new entities. This approach shows that the algorithm’s performance can decline with the introduction of incorrect or irrelevant data (Riloff et al. 1999).
Later SSL exploited existing NER systems for initial examples rather than manually redacted seeds, employing syntactic relations such as subject-object pairs to make the surrounding entities more evident (Cucchiarelli et al. 2001).

Further innovation has been done by integrating distributional similarity methods to generate synonyms and enable pattern generalization (Pasca et al. 2006).

Moreover, co-training (Kozareva et al. 2005) involves training multiple classifiers on different data views and iteratively labeling unlabeled instances to build a more solid model, indeed, the gazetteer lists extracted from unlabeled data are automatically used as features. The performance is then compared by using only labeled data and using bootstrapping.

The combination of pre-training and self-training techniques shows improvements in the performance of biomedical NER systems, outperforming standard supervised training methods across different datasets such as NCBI-disease and many others, that cover different types of biomedical entities. In the pre-training phase, models are trained on large-scale unlabeled data to learn general language representations. Consequently, in the self-training phase, the model iteratively labels new instances, retraining itself on rediscovering its predictions (Gao et al. 2021)

2.5 Unsupervised Machine Learning Systems: early approaches, statistical methods, recent developments, and evaluation challenges.

Unsupervised learning does not require annotation beforehand and is particularly useful for languages and domains that lack annotated data. Approaches use lexical resources, statistical patterns, and clustering algorithms based on similarities and shared properties in the data, allowing systems to discover new instances autonomously. However, unsupervised learning is not a very popular approach for NER, as systems are not usually completely unsupervised and often need to be integrated with other methods. These approaches can be flexible across different domains or languages (Mansouri et al. 2008).

The first unsupervised systems for NER typically used lexical resources and statistical approaches. WordNet (Fellbaum 1998) is a large lexical database of English in which parts of speech are collected into sets of cognitive synonyms, each of them expressing distinct concepts. These sets are interlinked using conceptual-semantic and lexical relations. It has been a reference lexical database for labeling input words with named entity types based on co-occurrence patterns in large corpora, without labeling training data (Alfonseca et al. 2002)

Another technique extends the use of lexical resources by finding potential hypernyms (broader terms). The recognition task is performed by identifying normally capitalized phrases in a document and then submitting queries to a search engine to find potential hypernyms of the capitalized sequences. The hypernyms are then clustered to derive a typology of named entities for the document (Evans 2003)

The Pointwise Mutual Information (PMI) statistical method has been introduced to measure the association strength between terms and their surrounding context, providing metrics for entity classification. The KnowItAll system (Etzioni et al. 2005) operates by extracting a large collection of names from the Web in an unsupervised and domain-independent manner using PMI. PMI validates candidates' named entities and their association with specific contexts, indicative of the entity's class.

Named entities tend to appear synchronously in multiple news sources (Shinyama et al. 2004). Leveraging this observation, rare named entities might be identified effectively using clustering and context similarity methods, exploiting and comparing the time series distributions of words in different newspapers. Unsupervised methods face issues such as the ambiguity of entity mentions, which is addressed by combining knowledge-based techniques with statistical methods and using contextual information from uniquely matched entities. Statistical measures like TF-IDF (Term Frequency-Inverse Document Frequency) and RTF (Relative Term Frequency) are used to select discriminative words that are useful in resolving ambiguities (Nadeau et al. 2006).

For instance, unsupervised learning in social media analysis involves a two-stage process. It starts with pre-labeling using a linear conditional random fields model, followed by fine-tuning with an improved model based on tweet clustering. The second step groups tweets into clusters and refines the entity recognition process by exploiting the redundancy and context similarities within the clusters, reflecting the redundant nature of social media data and learning how to handle it (Liu et al. 2013).

More recent studies on unsupervised NER address challenges such as domain distribution shift and noise in entity ranking. Innovative techniques include adversarial training, entity-aware attention, and KDE-centered ranking algorithms (Peng et al. 2021; Dutta et al. 2022). The outputs from unsupervised learning systems do not have labels, unlike supervised learning systems, which are evaluated on the "gold standard," which employs a labeled data set. This makes the use of traditional evaluation metrics a challenge.

3. DEEP LEARNING AND FINE-TUNING TRANSFORMERS

3.1 Overview

The evolution of deep learning profoundly changed named entity recognition approaches in the architecture of new systems, as for many other NLP tasks. However, the basics for neural networks were developed in the 1980s, with the idea of emulating the animal nervous system, first recreating a model of a biological neuron. In simple terms, neurons are composed of one cell receiving many connections called dendrites and some input modulated by synapses by other neurons. Consequently, if the soma cell body receives enough signal by accumulating all the input signals, then the neuron sends the signal to the axon which, in the end, is connected through the axon terminals to the dendrites of other neurons.

To emulate the behavior of a biological neuron, we introduce the structure of an artificial one. A biological neuron collects inputs from other neurons, which are modulated by synapses, similarly, we can modulate these inputs by weighting them. All data are accumulated, and all the weighted contributions from the network input are summed. After, the neuron must decide: if this activation is above a given threshold, the neuron fires and activates; if it is below the threshold, it does not activate and produces no output.

The general activation function should be logistic or hyperbolic, classical replacements for binary functions, which are not ideal for the learning algorithm. In a binary function, if the input is below the threshold, the output is 0; otherwise, it is 1. This function is neither continuous nor differentiable at 0. So, it would be much better to approximate the step with other similar functions that exhibit continuous behavior (and therefore are differentiable at every point).

Nowadays, some scholars have introduced different kinds of activation functions, which may be better in some cases: the ReLU (Rectified Linear Unit) and the GELU (Gaussian Error Linear Unit).

Given that real problems can be solved by a network of neurons (as it happens in our brain), the question is: how do we combine artificial neurons to obtain models able to solve real-world problems? In the past, millions of architectures have been proposed, we can describe three main kinds of architectures that form the backbone of any other more complex deep neural networks: Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNN), and Recurrent Neural Networks (RNN).

The MLP architecture has an input layer receiving as many inputs as the dimensions of the instance, formalized as a vector, with the correct output for classifying the instance, since DNNs are supervised ML systems. In the output layer, there are corresponding classes. Essentially, in the input, one neuron exists for each feature, while in the output, there is one neuron for each class in which the input values must be classified (Fig. 12). It is organized in layers: each neuron of a given layer connects to all the neurons of the previous layer and the following layer, without exceptions. If a connection's weight becomes 0, they are no longer connected. Every connection has a weight that simulates the synapse, modulating the output of one neuron before it arrives at the input of the following neuron.

A given neuron collects all the outputs from every neuron in the previous layer, sums them by weighting them with their corresponding weight, and if this sum is above a given threshold, sends the result forward. All neurons behave this way, except for the input neurons, which collect and visualize the inputs.

In defining the architecture, all the weights and thresholds of the neurons forming the MLP must be defined as parameters of the models. These values must be properly set during the training process, during which the neural network adapts its behavior to match the correct output. The weights are initialized randomly, and the system requires a set of correctly classified instances. As in other supervised ML systems, the output is compared with the correct one, and the backpropagation algorithm backpropagates the error, adapting each weight to minimize the error. The training proceeds forward and backward millions of times, and step by step, the weights are adapted to minimize the error. Many hidden layers can be inserted into the network.

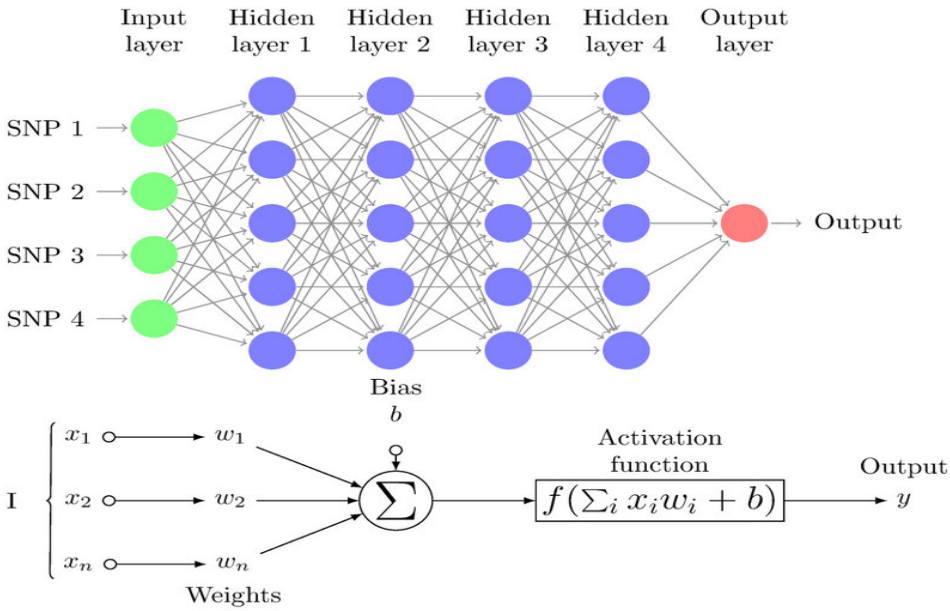


Figure 12: MLP architecture.[24]

Today, many different architectures, typically much deeper than MLP, have been devised. Computing the backpropagation algorithm for such complicated models was challenging, but computational graphs now allow this. Any kind of mathematical expression can be seen as a DAG (Directed Acyclic Graph), where nodes represent mathematical operations and edges are labeled with variables or values. The output/result (equivalent to the error) can be computed by flowing through the graph, and a procedure can backpropagate and minimize the error.

Similarly, the idea behind Convolutional Neural Networks (CNNs) came from neural systems of animal eyes that process information, making them interesting for processing images. These images can be represented as matrices of values, and a convolution operator (a very small kernel or filter) is applied to the entire image. The process is initialized by applying the kernel in a specific place, where each cell of the kernel is multiplied by the corresponding cell in the image, and the results are summed. This mathematical operation produces a single value, consequently, the kernel is shifted to a new portion of the original matrix, and the same operation is repeated until the filter has covered the entire image. At the end of the procedure, the matrix is transformed into a feature map.

Since multiple filters are applied in the same convolutional operation, the values of the filters are the parameters of the convolutional neural network, which are set during the training

[24]https://www.researchgate.net/publication/334609713_A_Guide_for_Using_Deep_Learning_for_Complex_Trait_Genomic_Prediction/figures?lo=1

process to adapt their behavior, for instance, taking a group of cells and replacing it with the maximum value. In this way, the feature map can be subsampled and reduced in its size.

Another set of convolutions may then be applied to define another set of filters. CNNs alternate between sets of convolutions (given by certain sets of filters) and subsampling repeatedly. The number of cycles depends on the task and the complexity of the network being built. At the end of these convolutions, the final set of subsampled feature maps is taken as features for a fully connected MLP or another classifier to make the final decision. CNNs do not make decisions; they transform the information received into other sets of information (Fig. 13).

Starting from an image, the first layer of convolution extracts low-level features, such as lines with different colors, borders between colors, small circles, and simple geometrical configurations. The second layer of convolution, applied to the first one, extracts mid-level features, identifying more complex features and shapes. Each subsequent convolutional layer abstracts further from the original information, creating a more abstract representation of the image and recognizing different complexities. This process serves as a feature processing, creating different abstraction layers. At the end of this process, a trainable classifier must be added to solve a specific task.

Usually, an MLP classifier is used, but depending on the task, a more complex classifier could be employed. The learning mechanism remains the same: an image is presented, the network propagates the output, and the last layer classifies the image into one of the n-possible classes. Initially, the network makes many errors, so the error is backpropagated. The weights in this case are the values of the kernels, which move around on the image. Each transformation yields a more abstract representation of the original input, and each kernel at the end of the training specializes in recognizing specific configurations in the input. For visual inputs, this means recognizing geometrical figures; for textual inputs, it means recognizing correlations between linguistic phenomena.

CNNs can also be applied to texts by transforming words into embeddings/vectors. By associating a vector of features with each word and stacking these vectors for all the words in a sentence, a matrix (an image) is created. The same structure can be applied, with kernels moving only vertically, which is more useful for textual data.

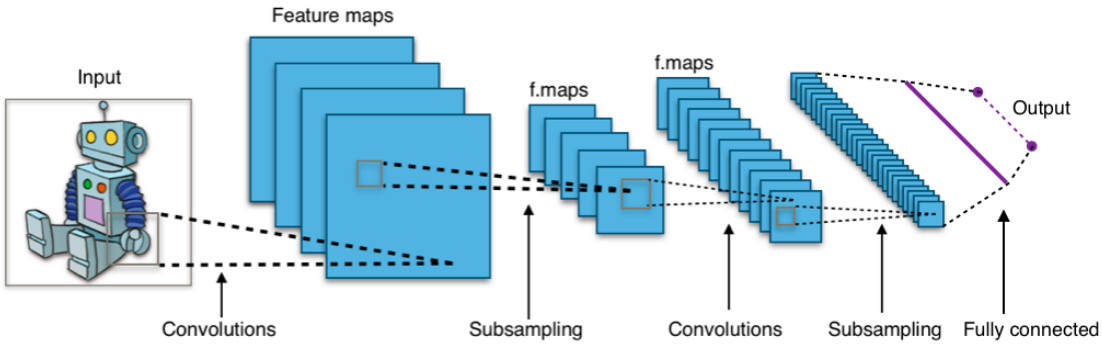


Figure 13: Convolutional Neural Network.[25]

Returning to the MLP architecture, the number of input neurons equals the number of features defining the instance of the problem. This number of cases is defined by the user.

A property of the language is linear, this means that each event in the linear chain is connected to all other events before or after it. So, to process a sequence of tokens using an MLP, the horizon must be fixed, considering the surrounding words.

More context can be added, but the horizon remains fixed because the number of input neurons is fixed. The MLP architecture cannot dynamically adapt to a sequence of events that can change its length. The same limitation applies to CNNs: the input matrix is fixed, and its dimensions cannot be changed. Therefore, a specific architecture, designed to handle sequences of events, is needed: the Recurrent Neural Network (RNN) architecture.

Recurrent Neural Networks are mostly appropriate for handling language phenomena due to their ability to manage data sequences. Their structure is simple but powerful, consisting of three elements: an input layer, an output layer, and one or more hidden layers. A loop-back connection represents the output of the hidden layer together with the input for the next step, pointing out the importance of timing in RNNs as they work step-by-step. At any given point in processing a linear sequence, the hidden layer receives two types of information: the current input in the sequence and the output of the hidden layer from the previous step. (Fig. 14).

This architecture can also be represented by unfolding it through time. Starting from the second step, each step is processed, and the corresponding output is produced using the sequence input and the hidden layer's previous output. Training remains consistent: each sequence of data, once entirely processed, produces an output compared to the correct one, followed by backpropagation throughout the entire network. The task at hand determines what is optimized.

Early RNNs had simple hidden layers like those in MLP, where their outputs became an additional input. However, this setup was suboptimal as the backpropagation of the gradient vanished

[25] https://www.shiksha.com/online-courses/articles/a-comprehensive-guide-to-convolutional-neural-networks/

almost immediately for long sequences, making it impossible to optimize the weights for the initial elements. Modern RNNs address this with hidden layers based on Long Short-Term Memory (LSTM) units and Gated Recurrent Units (GRU). Adding gates and memory within the hidden layer allows these models to remember past information, permitting more efficient backpropagation throughout the entire sequence.

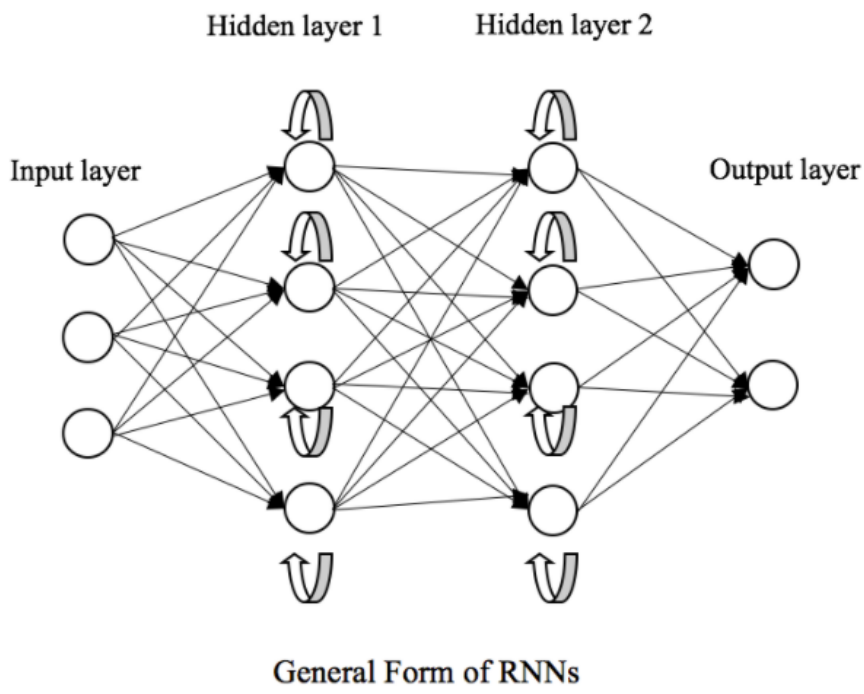


Figure 14: RNNs schema.[26]

By propagating the hidden layer throughout the sequence, decisions at time t can be influenced by any previous input and decision. Thus, RNNs are strong, as they can theoretically consider all prior information when making a specific decision. However, the actual storage capacity of the hidden layer is limited, so it is unclear how much information they can retain.

A limitation arises when future information is needed for making a correct decision at time t, such as in part-of-speech tagging where future words influence the tag for a specific word. Bidirectional RNNs solve this by using two RNNs: one follows the sequence, producing outputs by looking at the past, and the other processes the reversed sequence, essentially looking at the future. The output combination of both RNNs provides all the information required for the problem. Indeed, bidirectional RNNs are capable of processing entire sequences and applying the decision process by considering all the relevant data.

[26] https://opendatascience.com/understanding-the-mechanism-and-types-of-recurring-neural-networks/

Therefore, RNNs can be applied to many sequence-to-sequence problems, depending on the specific tasks. Tasks such as machine translation and text summarization require combining two RNNs, an encoder and a decoder. The encoder processes the input sequence into a single vector, and the decoder uses this vector to generate the output sequence. However, the encoder-decoder architecture presents issues, since the bottleneck between the encoder and decoder requires compressing all input sequence information into one vector, and enlarging this vector increases training time. The compromise between quick training phases with limited memory and long training times with better memory capacity affects the learning of long-term dependencies.

The encoder-decoder setup loses information due to the single-vector bottleneck, causing errors in the de-codification. Allowing the decoder to access all hidden states of the encoder could provide a more comprehensive view of the input information. The concept of attention in RNNs addresses this by exposing all encoder hidden states to the decoder, enabling it to consider multiple context factors rather than relying on a single context vector.

This approach was first proposed by Bahdanau et al. in 2014, introducing the idea of attention in RNNs. Attention is a concept applied within the RNN with an encoder-decoder architecture and Bahdanau examined this in the context of machine translation. When a decoder generates an output, it can utilize input information and access all the hidden states of the encoder. However, these states need to be combined effectively. In Bahdanau's proposal, each encoder state and the decoder state are combined using a specific function that computes a score indicating the relevance of each encoder hidden state to the decoder state being examined. This is achieved through a scoring function that measures the similarity between the two states, essentially determining how connected they are. The goal is to assign greater weight to the hidden encoder states that are more like the decoder state in the weighted combination. These weights are known as attention weights. They examine the input sentence to identify which parts of the input sequence are more relevant for deciding a specific point in the output sequence. Attention can be understood as key retrieval in a database, where different values are connected to different keys. When a query is made, the objective is to retrieve all values corresponding to the key that matches the query. A similarity function compares the query to the key, providing a similarity score. This score (attention weight) is then used to weigh the contribution of each value in computing the final attention.

Self-attention is a variation of attention computed from a single set of data. This is the core mechanism within transformers. In this process, a sequence of input vectors or feature vectors is given. From each element of the sequence, query, key, and value vectors are derived. This is termed self-attention because these values are derived by examining other elements in the sequence. This mechanism allows each element in the sequence to consider its affinity and

correlation with all other elements. Attention is then computed in the same way as before, but instead of comparing the encoder's hidden states to the decoder's hidden states, one set of hidden states is compared to all others within the same sequence.

Transformers, introduced by a Google team, are among the most significant developments in DNN. They address the slow training times and gradient propagation issues of RNNs, which struggle to remember past information in long sequences. In 2017**,** Vaswani et al. proposed transformers, organized as an encoder-decoder architecture but without using RNNs. The transformer architecture (Fig. 15) fixes the maximum number of elements to a large number, leveraging Graphic Processing Units (GPUs) for large input dimensions and fast calculations. The encoder is a stack of simple blocks consisting of multi-head attention and a feed-forward network (MLP).

Self-attention is computed by matrix multiplication, the fastest operation on GPUs, over the entire sequence. This allows different kinds of attention to be computed in parallel over the entire sequence, with many multi-head attentions, processing the input sequence simultaneously. A feed-forward network then consolidates these attentions' outputs into a fixed representation.

The encoder block transforms the input representation into another representation, hence the name "transformers." The final hidden state of the encoder and all other hidden states are passed to the decoder, which has a similar structure. The decoder employs masked multi-head attention, allowing it to see only the past sequence, ensuring a partial view as it generates the sequence step-by-step. Attention is computed only on past states.

Transformers can process an input sequence and produce an output sequence without involving any RNNs. Although transformers fix the input size, this limitation is manageable. The attention values for irrelevant parts become zero, excluding them from calculations.

Since MLPs do not inherently recognize word positioning in sequences, a position vector is added to the input features to introduce positional information. This position vector ensures that the network understands the order of words in the sequence.

Transformers essentially consist of a stack of encoders and decoders, transforming input into output. The original proposal by Vaswani et al. in 2017 has led to many variations, most implementations use only the encoder stack or the decoder stack. Notable architectures derived from this proposal include Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. 2018) and Generative Pre-trained Transformers (GPT) (Radford et al. 2018).

BERT has been introduced as a novel language representation model. Despite previous models like ELMo or OpenAI’s GPT that are unidirectional, meaning that they process text either from left to right or right to left), BERT is designed to pre-train deep bidirectional representations by conditioning on both left and right contexts simultaneously. In this way, the model can capture

the context of a word in a sentence enhancing its language understanding skills. Indeed, the pre-training of BERT consists of two main objectives, which are the "masked language model" (MLM) approach and the "next sentence prediction" (NSP) task. Random tokens in the input text are "masked" (hidden) and the model learns to predict them based on their surrounding context and not just sequentially. The NPS task is important to further enhance BERT's understanding of relationships between sentences, it involves predicting whether a given sentence follows the other one in the original text. The pre-trained model is then fine-tuned on a smaller dataset to adjust it to the specific task.

BERT utilizes only the encoder tack to create abstract input representation, while GPT uses only the decoder stack for text generation, producing texts based on input after training on large text datasets in a supervised manner. It is unidirectional, reads the text from left to right, and is more prominent for generation tasks since each generated word is based only on the previous words. For these reasons, BERT it is mainly used for language understanding tasks like NER. Otherwise, GPT is trained on the "Language Modeling" task to predict the next word, making it excellent for completion, writing, and conversational tasks.[27]

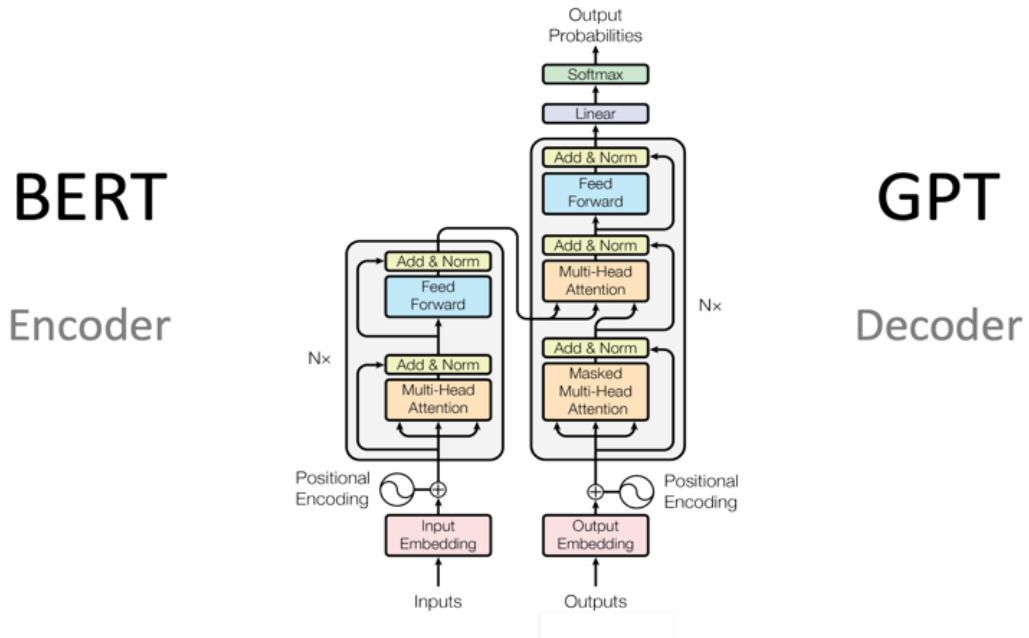


Figure 15: Transformers architecture.[28]

[27] https://blog.invgate.com/gpt-3-vs-bert
[28] https://heidloff.net/article/foundation-models-transformers-bert-and-gpt/

3.2 Deep learning and transformers for NER: reasons for applying deep learning in NER tasks, deep learning steps and popular approaches.

There are mainly three reasons for applying deep learning techniques for NER. Compared with linear models, like HMM and linear chain CRF, deep learning-based models can learn complex and intricate features from data. In addition, deep learning saves efforts in the design of NER features, since feature-based approaches require engineering skills and domain expertise. Conversely, DL-based models effectively learn useful representations and underlying factors from raw data. The last reason is the end-to-end training paradigm, by gradient descent, a property that enables the design of possibly complex NER systems. (Li et al. 2020).

This architecture is made up of three steps, namely data representation, context encoding, and entity decoding. Deep learning architectures are often combined with supervised learning models like SVMs and CRFs. The data representation, the first step in deep learning, is aimed at representing words with vectors that serve as an input layer for these kinds of models.

Three types of distributed representations have been used in NER models: word-level, character-level, and hybrid representations. Word-level representations are typically pre-trained over large collections of text through unsupervised algorithms such as Continues Bag of Words (CBOW) and Skip-Gram models. Using as the input, the pre-trained word embeddings can be fixed or further fine-tuned during NER model training. Popular choices among word embeddings for named entity recognition are One-Hot encoding which represents each word as a sparse binary vector. It is very easy to understand and use, but it cannot prioritize the words based on importance and can result in the formation of a sparse matrix that is computationally and memory expensive. (Jehangir et al. 2023).

The Term Frequency (TF) technique is employed as well to attest to the importance of a word in a document relative to a corpus, measuring its appearance across a specific document. In contrast, the Inverse Document Frequency (IDF) component measures the rarity of a word in the corpus. Other employed unsupervised techniques are Word2Vec, GloVe, and fastText aimed to capture semantic similarities and syntactic relationships.

Character embeddings focus on capturing the internal structure of words at a character level, exploiting explicit sub-word-level information such as prefixes and suffixes. This kind of representation has the advantage of naturally handling out-of-vocabulary words, inferring representations for unseen words, and sharing information on morpheme-level regularities.

Character embeddings are generated by using various methods: they extend Word2Vec, predicting embeddings considering neighboring characters. CNN-based and RNN-based models are often used for extracting character-level representations, where the former applies convolutional operations to capture local character features, and the latter uses sequential operations to capture

character dependencies. Moreover, character embeddings use stacked LSTM networks to detect contextual information from character sequences, this is particularly useful to eliminate the need for feature engineering, language-specific syntactic and morphological analyzer, and language-specific lexical or named-entity resources. The LSTM model can operate on sequential data of arbitrary length and encode observed patterns in its memory at different scales. A Viterbi decoder is then used to convert the character-level tag probabilities into consistent word-level tags. In essence, character-based models have been introduced to address the data sparsity problem inherent in word-level inputs. (Kuro et al. 2016).

Beyond word-level and character-level representations, the representation might be combined with a feature-based approach in a hybrid manner, by incorporating gazetteers, lexical similarity, linguistic dependency, and visual features into the final representations of words before feeding into context encoding layers.

As the second step, after data representation, in deep learning systems for NER, context encoding utilizes feature vectors obtained through textual embeddings, including words, characters, or their combination. Context encoding techniques are meant to consider dependencies and relationships between neighboring words. Recent models like BERT and ELMo use pre-trained language models to provide contextualized word embeddings, capturing richer semantic and syntactic information.

This step presents classic context encoder architectures such as CNNs and RNNs. For example, hybrid models like BiLSTM-CRF and CNN-BiLSTM integrate character and word-level features with additional information to enhance NER performance. For instance, using convolutional neural networks, after each word in the input sequence is embedded in a vector at the input state representation, a convolutional layer is used to produce local features around each word. The global feature vector is constructed by combining local feature vectors extracted by the convolutional layers, with a fixed dimension regardless of sentence length. Finally, these global features are fed into the tag decoder to compute distribution scores for all possible tags for the words in the network input.

Recurrent neural networks with variants like Gated Recurrent Unit (GRU) and Long-Short-Term Memory (LSTM) have demonstrated remarkable achievements in modeling sequential data. In the context of NER, each word is fed into the RNN one at a time, the system then maintains the hidden state which captures information from the previous words. Particularly, bidirectional RNNs use past information (via forward states) and future information (via backward states) for a specific time frame. Tokens encoded by a bidirectional RNN will contain evidence from the whole input sentences. Bidirectional RNNs are a standard for composing deep context-dependent representations of text. The key challenge with RNNs is their limited ability to capture long-term dependencies, this is the

reason to use LSTM and GRU, which use gating mechanisms that help them retain and update information over long sequences.

The last step for deep neural network architectures for NER is entity decoding, which is aimed at predicting entity labels for each word in each sequence. These architectures typically involve CRF, MLP, and Pointer Networks. (Keraghel et al. 2024), (Chen et al. 2018).

The first ever work on named entity recognition using deep learning, which was also designed to handle multiple natural language processing tasks simultaneously, including NER, was done by Collobert et al. in 2008. This single model, trained with multitask learning, is an architecture based on a convolutional neural network trained on unlabeled text from Wikipedia, in which deep layers automatically learn and extract features relevant to the tasks, sharing parameters across them, to understand commonalities and reduce overfitting. The input sentences are processed through feature extraction layers, and automatically trained to be relevant to the task. First, the initial layer maps words to real-valued vectors using a lookup table, transforming the input sentence into a sequence of vectors. The CNN architecture includes a Time-Delay Neural Network (TDNN), a layer to handle sequences of different lengths, the output from this layer is then passed through a "Max" layer, that understands the most relevant features over the entire sentences so that the network focuses on the most informative parts based on the task.

The integration with other NLP tasks, like part-of-speech tagging, chunking, and semantic role labeling improves the generalization and the performance, by training these tasks jointly in a multitask learning framework. The parameters are then shared in the deep layers of the network to enhance the learning of the relevant features for each specific task, including NER. The NER task is trained using labeled data; however, the architecture incorporates a language model trained unsupervised using the Wikipedia dataset. This unsupervised training provides additional context and helps improve performance through semi-supervised learning. The NER task at the time got significant improvements (Collobert et al. 2008).

RNNs are frequently used by academics for NER because of their capacity to handle dynamic input sequences. Later, RNNs were eventually replaced by the classical long short-term memory (LSTM), which uses a gating mechanism to achieve long-distance transmission of information.

Another popular approach in deep learning for NER, involves LSTM networks and Bidirectional LSTM, in combination with other models like conditional random fields. This way demonstrated improvements in sequence tagging tasks, the most used approach before transformers. In this perspective, one of the most promising approaches involves using the BI-LSTM-CRF model (Huang et al. 2015), which uses past and future input features via bidirectional LSTM components and leverages sentence-level tag information through the CRF layer. This one predicts optimal tag sequences by considering the entire sentence, rather than individual positions, showing high accuracy

in the results. The training procedure involves a generic Stochastic Gradient Descent (SGD) method, illustrated using the Bi-LSTM-CRF model. The training data is divided into batches, and the model parameters are updated through forward and backward passes of the LSTM and CRF components. (Fig. 16). The model has been tested on three NLP tagging tasks: POS tagging (Penn TreeBank), chunking (CoNLL 2000), and NER (CoNLL 2003). The features extracted included spelling and context features, plus word embedding was used to improve performance. Although is less dependent on word embeddings, this model needed more training.

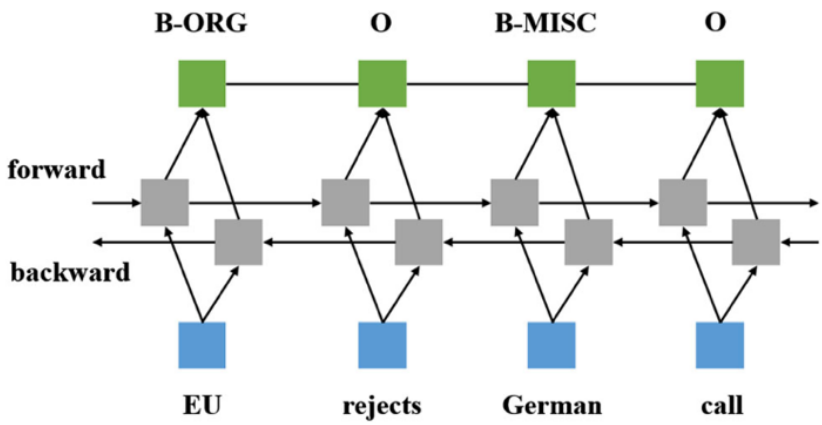


Figure 16: Bi-LSTM-CRF model (Huang et al. 2015; Hu et al. 2024).

Similarly, with a focus on the NER task specifically, the use of character-level embeddings created by running a bidirectional LSTM over the characters of each word and combining them with word-level embeddings demonstrated advancements in the performance for multiple languages such as English, Dutch, German and Spanish (Lample et al. 2016).

Ma et al. used CNNs to derive character-level word representations, and the word-level and character-level coding vectors from the CNNs are cascaded and fed into the BiLSTM-CRF. The labels of the whole sentence are jointly decoded using continuous CRFs. The model achieves the best NER performance that is available at the time without relying on task-specific resources, feature engineering, or data pretreatment. However, the model is composed of two different architectures, which can easily cause error propagation if the predecessor model makes mistakes (Ma et al. 2016; Hu et al. 2024).

Most recent biomedical and medical approaches evaluated on popular datasets like NCBI-Disease combine CNNs and BiLSTM networks with a CRF layer (Cho et al. 2020). The reason is that biomedical NER models often fail to handle new and unseen entity types. The deep learning approach

to address this problem consists of utilizing pre-trained word vectors from large corpora to detect semantic information of words. Then, a CNN is used to extract local patterns and character-level features, while the BiLSTM focuses on capturing long-term dependencies and contextual information, processing the input sentence in both forward and backward directions. The CRF is used for decoding, considering the dependencies among adjacent tags. Adding an attention mechanism is applied to improve the recognition of entities by focusing on relevant tokens.

As we've already seen, the transformer architecture (Vaswani et al. 2017) is suitable for sequence-to-sequence problems and designed to handle long-distance dependencies. It utilizes stacked self-attention and point-wise layers to build basic blocks for the encoder and decoder. Studies show that transformers are superior in quality while requiring less time to train.

BERT (Devlin et al. 2018) is distinctive in its application of named entity recognition, being a language representation model designed to pretrain deep bidirectional representations from unlabeled text. Its power lies in that it conditions both left and right contexts simultaneously, in all layers, a method that differs from prior models. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create SOTA models for multiple tasks. Transformer-based encoder models are used directly for NER as context encoders or to provide inputs to other models like CRF or LSTM. Indeed, other authors (Labusch et al. 2019) demonstrated that using BERT as a classifier outperforms other Bi-LSTM-CRF-based methods. BERT-derived models, for example, trained on Italian texts, improve performance across key NLP tasks, including NER, establishing new SOTA benchmarks. The simplicity and effectiveness of fine-tuning BERT-derived models without extensive parameter tuning suggest further potential performance gains with optimized hyperparameters (Tamburini 2020).

Other models like RoBERTa (Liu et al 2019) work well on NER, utilizing a larger pre-training corpus and employing a novel training approach, the masked language model, enabling the detection of deeper contextual embeddings and representations. Based on RoBERTa, a monolingual transformer-based model named CamemBERT (Martin et al. 2020), has been developed for the French language. One of the motivations that led to this work is the limited availability of pre-trained language models for languages other than English. This model has been trained using the RoBERTa architecture on a different dataset, focusing mostly on web-crawled data from a corpus. The model shows its effective performance on web-crawled data, along with the feasibility of creating high-performing language models for languages with smaller datasets than the ones used for English. This poses a challenge for the idea that massive datasets are always necessary for training effective language models, discussing the implications for under-resourced languages, and suggesting that similar models may be developed using available web-crawled data. The study shows that using these data is preferable to using Wikipedia data and that the results on a relatively small dataset are as good

as those obtaining larger datasets. CamemBERT is an example of creating high-quality and language-specific models using public corpora, underscoring the importance of data diversity and quality over size.

In conclusion, BiLSTM-CRF is the most common architecture for NER using deep learning, however, the transformer encoder is more effective than LSTM when it is pre-trained on huge corpora. Generally, transformers fail on NER tasks if they are not pre-trained and when the trained data is limited. On the other hand, it is faster than recursive layers when the length of the sequence is smaller than the dimensionality of the representation (Li et al., 2020).

Many variations of BERT have appeared because of the researchers' extensive investigation of it. The pre-trained language models have also been open-sourced by researchers, which considerably expedites future research and encourages the creation of pre-trained language models (Hu et al. 2024).

4. LARGE LANGUAGE MODELS, PROMPTING AND EMERGING PERSPECTIVES.

4.1 Introduction

In the first decade of the 2000s, natural language processing was a completely supervised approach, not grounded on DNN with feature engineering. An important enhancement in the field started in 2010, with GPUs and Representation Learning as the main two progresses to develop new NLP operations. From 2010 to 2018 the paradigm was still supervised but using neural networks and word embeddings, with no longer the need to select features, the engineering was concentrated on contriving an applicable architecture for dealing with specific problems.

Starting from 2017, first with the preface of the transformer model and in 2019 with the debut of BERT, the paradigm radically changed, with transformers-based pre-trained language models (PTLMs), to break specific tasks. On the engineering side, the focus is on optimizing the pre-training phase. The language modeling answered by pre-training of transformers becomes the core of the solution by using language models, with minimum supervision, and by fine-tuning them, it's possible to handle either classification or sequence tagging problems.

The current paradigm employs such models after only fine-tuning them to a specific task, adding some more layers only after it. This means adding some further layers, after their outputs, and training them alone or jointly with the PTLM, to solve the task using appropriate training datasets developed exactly for this purpose. When the PTLM becomes large and trained with such a huge dataset containing a large variety of text types, it is possible to solve many downstream tasks by simply posing the right questions to the language model and letting it answer them.

Increasing the dimension of a PTLM makes it a good few-shot learner, meaning that with very few examples of correct task resolution, as prompts, it can solve the task exhibiting better and better performance, without the need for a specific dataset. In this way, a generic PTLM is driven to solve the task by emulating the behavior of the prompts example. Prompt engineering offers the advantage of not needing labeled data, which is costly and often presents legal or privacy issues. The developing of a model per task is expensive as well, while with one model you can potentially solve a lot of different tasks, reducing the development work. Plus, these models can learn high-level concepts, presenting better generalization abilities.

On the contrary, the new paradigm is computationally very expensive, comprehending many parameters. When a model needs to manage 300 million parameters for both training and interference it is indeed computationally cheaper. However, the dimensions of pre-trained language models progressed through the years. The first contextualized language model, ELMo, comprised 94 million

parameters, while BERT was 340 million in 2018. GPT-2 had 1.5 billion parameters, GPT-3 had 175 billion parameters, and GPT-4 had 100.000 billion.

At the end of 2021, Megatron-Turing, a joint force between NVIDIA and Microsoft, composed of more than 500 billion parameters, trained with 870 TB of plain text and tried to pre-train a language model going over 1000 billion parameters. The training and the inference with models with more than 500 parameters need computing infrastructures requiring serious economic investments which must cover thousands of state-of-the-art GPUs, huge amounts of terabytes of memory, an appropriate networking infrastructure, and large, curated datasets. Such kind of investments are not sustainable by scholars and research teams. On the contrary, nowadays, this responsibility falls primarily on big tech companies like Google, Microsoft, Meta, Amazon, and Apple, as well as dedicated AI research labs like OpenAI (Tamburini 2022).

The basis of LLMs is in the transformer decoder model, where multiple attention layers are stacked to create a complex neural network. Large language models resemble smaller language models in the architectures and pre-training; however, they are much more increased in terms of the size of the model and training data (Keraghel et al. 2024). Unlike the BERT model, which focuses on improving the accuracy and depth of language understanding, GPT performs better in generating tasks. The model uses a one-way (forward) attention mechanism, compared to BERT's two-way attention mechanisms, which makes the model architecture relatively simple. The challenge for GPT comes in terms of high computational resource requirements when processing long texts, poor model interpretability, and large fine-tuning data requirements. GPT-3 can process longer text sequences, helping the model to understand more complex structure texts. However, with its 175 billion parameters, the deployment may include difficulties, for this reason, the researchers explore ways to improve the efficiency of the model, for example, by optimizing its architecture, reducing the number of parameters, and adopting more efficient training techniques (Yang et al. 2024).

4.2 The behavior of large language models and prompting on the NER task

Large language models were originally designed for text generation, so despite their nice performance in NLP tasks, the models show important limitations when applied to NER because NER is a sequence labeling task and not a text generation task. In the last few years, attempts with LLMs have been made for NER. The table below (Fig. 17) shows a summary of some of them, together with their outcomes:

Study	Approach	Context	Outcome
GPT-NER [Wang et al., 2023]	Transforms sequence labeling to text generation	General NER tasks	Comparable to fully supervised baselines, better in low-resource and few-shot setups
PromptNER [Ashok and Lipton, 2023]	Uses entity type definitions for few-shot learning	Few-shot learning in NER	State-of-the-art performance on few-shot NER, significant improvements on various datasets
ChatGPT Evaluation [Laskar et al., 2023]	Evaluates ChatGPT on various NER tasks	Various NER tasks	Impressive in several tasks, but far from solving many challenging tasks
Injecting comparison skills in TOD Systems [Kim et al., 2023]	Compares properties of multiple entities	Task-Oriented Dialogue Systems	Effectively addresses ambiguity handling in database search results
Zero-Shot on historical texts with T0 [De Toni et al., 2022]	Explores zero-shot abilities for NER	Historical texts in multiple languages	Shows potential for historical languages lacking labeled datasets, error-prone in naive approach
Resolving ECCNPs [Kammer et al., 2023]	Proposes a generative encoder-decoder Transformer	German medical texts	Outperforms rule-based baseline
Large code generation models [Li et al., 2023]	Utilizes generative LLMs of code for Information Extraction tasks	Information extraction tasks	Consistently outperforms fine-tuning moderate-size models and prompting NL-LLMs in few-shot settings

Figure 17: Summary of studies on LLMs in NER from Keraghel's review (Keraghel et al. 2024).

Improvements have been made with the GPT-NER model (Wang et al. 2023), transforming the NER task from a sequence labeling task (where each word in a sentence is tagged with a label) into a text generation task. The authors introduce a method in which specific entities in the text are marked with special tokens during generation, allowing the model to more effectively identify and label entities. Instead of tagging each token in a sequence, the model generates annotated text with special tokens around recognized entities (e.g., "<entity_type> ... </entity_type>"). Prompts, carefully designed, guide the model to generate the desired output, including instructions that show how entities must be annotated (Fig.18). GPT-NER is optimized for few-shot learning, making it

adaptable to different domains without requiring extensive retraining. Moreover, the authors discuss a self-verification strategy to reduce errors, such as hallucinations, typical for this kind of model, in which the model incorrectly labels non-entities as entities. After the initial entity annotation, the model evaluates again its output, and corrections are made to minimize the issues. The results show that GPT-NER achieves performance comparable to traditional supervised methods, excelling in low-resource settings where limited training data is available, making it a competitive approach (Wang et al. 2023).

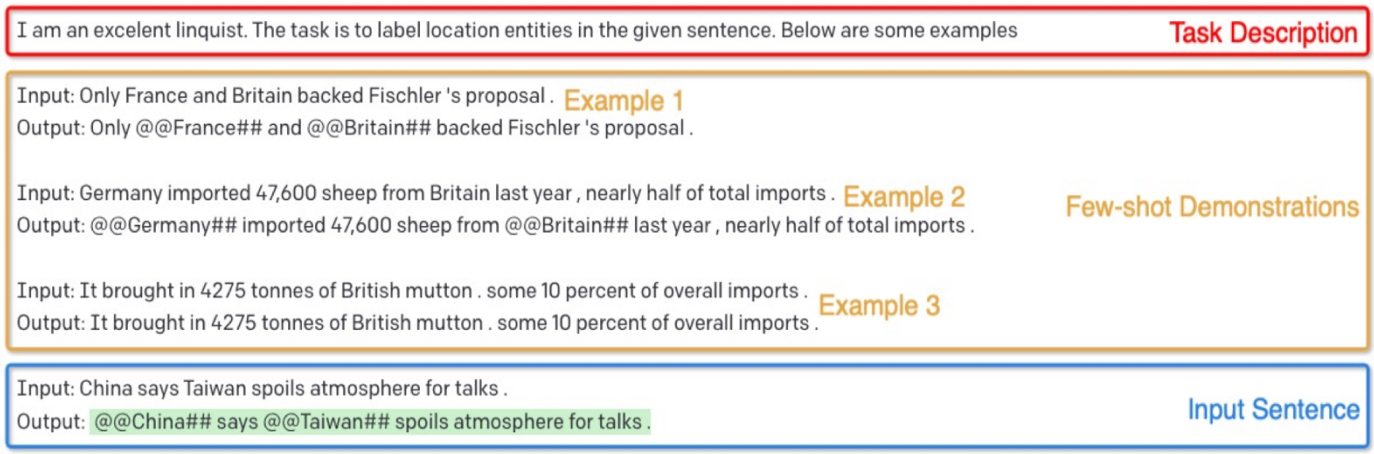


Figure 18: Example of GPT-NER's prompt in which the objective is to identify locations in the sentence.

The prompt is made up of:

1. **Task description (red rectangle): specify the goal to pursue for the GPT-3 model.**
2. **Few shots demonstration (yellow rectangle): comprehends few-shot examples for reference for the model.**
3. **Input sentence and output sentence (blue rectangle).**
4.

To improve the results on cross-domain NER, in which the model is trained on data from a specific domain and then is supposed to perform on new data from a different domain, PromptNER (Ashok et al. 2023) is a new algorithm that requires minimal training data and can adapt to other domains using definitions and few-shot examples. Indeed, the method prompts an LLM to generate a list of potential entities from a given text, supported by explanations for why each word or phrase qualifies as an entity, increasing flexibility across different domains (Fig. 19). The model achieves state-of-the-art results in few-shot NER settings, improving performance on datasets like CoNLL, GENIA, and FewNERD[29], and excels in cross-domain scenarios.

[29] https://ningding97.github.io/fewnerd/.

Input

Defn: An entity is a person(person), university(university), scientist(scientist), ,,,,event(event), award(award), or theory(theory). Abstract scientific concepts can be entities if they have a name associated with them. If an entity does not fit the types above it is (misc). Dates, times, adjectives and verbs are not entities

Q: Given the paragraph below, identify a list of possible entities and for each entry explain why it either is or is not an entity:

Paragraph: He attended the U.S. Air Force Institute of Technology for a year , earning a bachelor 's degree in aeromechanics , and received his test pilot training at Edwards Air Force Base in California before his assignment as a test pilot at Wright-Patterson Air Force Base in Ohio .

Answer:
1. U.S. Air Force Institute of Technology | True | as he attended this institute is likely a university (university)
2. bachelor 's degree | False | as it is not a university, award or any other entity type
3. aeromechanics | True | as it is a scientific discipline (discipline)
4. Edwards Air Force Base | True | as an Air Force Base is an organised unit (organisation)
5. California | True | as in this case California refers to the state of California itself (location)
6. Wright-Patterson Air Force Base | True | as an Air Force Base is an organisation (organisation)
7. Ohio | True | as it is a state (location)

Figure 19: Example of prompt to pre-trained language model. The definition is in blue, the question and task in red, and the example answer and chain of thought format in green (Ashok et al. 2023).

Moving forward, an extensive evaluation of ChatGPT (OpenAI 2022) comprehending multiple NLP tasks, covering 140 tasks, and analyzing 255,000 responses by the model shows the strengths of weaknesses of this one. The results are that ChatGPT generally performs well, in a pair or better than existing models. However, the performance is still less accurate than other task-specific and fine-tuned models. ChatGPT struggles with reasoning tasks and simple logical problems, and it is less effective in multilingual tasks, with less known languages than English. An interesting finding is that the model is uniquely capable of handling multiple arbitrary queries in a single prompt, making it advantageous in resource-constrained applications. The reliability of the model is in doubt as well, since it can produce hallucinated responses, so human intervention is often required (Laskar et al. 2023).

To evaluate and improve the GPT models' performance, specifically GPT-3 and GPT-4, Hu et al. (2023) developed a task-specific prompt framework for clinical NER. The materials used for this research include two datasets: the MT Samples corpus[30] for general medical issues and the Vaccine Adverse Event Reporting System (VAERS)[31]. The framework includes baseline prompts with task description and format specification, annotation guideline-based prompts (including specific definitions and rules), error analysis-based instructions (to address common mistakes made by the models), and annotated samples for few-shot learning (to guide the model through examples).

The study shows a quick and easy path to build more generalizable clinical NER systems using LLMs, indicating the possibility of building one large model for multiple information extraction tasks in the medical domain, such as medical problems, treatments, and tests from clinical notes. As we already said, traditionally, to build a machine learning or deep learning-based NER system for specific types of clinical entities, it is necessary to have an annotated corpus of clinical documents, which is time-consuming, costly, and requires domain expertise. The research explains that LLMs, without any further model training or fine-tuning, exhibit a performance, close to the fine-tuned models that require much more than a few samples.

In conclusion, the evaluation of two clinical NER tasks shows that the GPT-4 model with the proposed prompts achieved close performance as the state-of-the-art Bio-ClinicalBERT model (Alsentzer et al. 2019), fine-tuned specifically for bio-medical text (Fig. 20). The performance is impressive, considering that almost no training data was used in GPT, however, there is a need for further refinement before LLMs can effectively outperform BERT-derived models in this domain.
Despite these benefits, LLMs still require qualitative and precise annotated examples for few-shot learning, so the role of subject matter experts remains fundamental. Moreover, the current evaluation schema for NER may not be ideal since the boundaries of the identified entities need to match exactly with those in the annotated gold standard, this may be problematic for the ways that meaningful information presents itself in medical texts. GPT models are generative models and so they may interpret and generate text in a way that sometimes goes beyond the strict boundaries of predefined entity classes (Hu et al. 2023).

[30] https://mtsamples.com
[31] https://vaers.hhs.gov/index.html

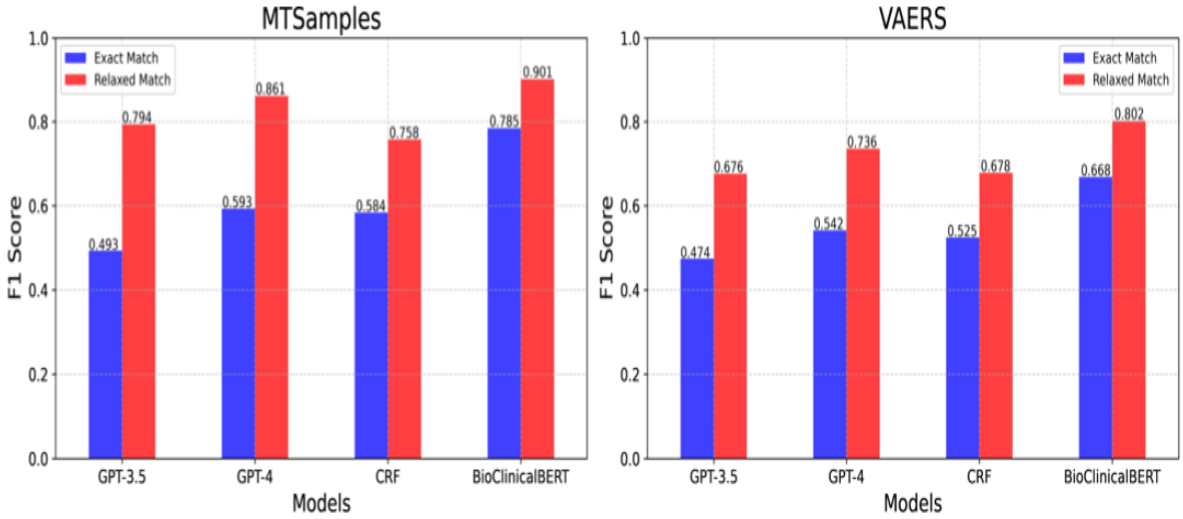


Figure 20: Comparison of F1 scores in recognizing clinical entities.

Overall, large language models have demonstrated generalizability in the understanding of arbitrary entities. However, they generally require prohibitive costs in training and inference. Instruction tuning, a technique for fine-tuning large language models on a labeled dataset of instructional prompts and corresponding outputs, gained popularity in distilling LLMs into more cost-efficient models. It is based on following instructions in general, helping adapt pre-trained models for practical use. Yet, "student" models still trail their original "teacher" LLM by large margins in downstream applications.

UniversalNER is a model that employs a targeted distillation approach for training "student models" through mission-focused instruction tuning for broad application classes like open information extraction, considering NER as a case study. The model can recognize many entity types coming from different text domains, outperforming existing instruction-tuned models of the same size, and surpassing ChatGPT as well. This last one was prompted to generate the instruction-following dataset, which comprehends 45,899 input-output pairs that include 240 725 entities and 13,020 entity types from different domains and subsets. The system uses a conversation-style tuning format where the LM is provided with a passage, and each entity type is queried using natural language processing questions. The outputs are generated as JSON lists for entities of the query type in the passage. During tuning, negative entity types are sampled from the collection of all entity types that do not appear in the passage as queries, and the expected outputs as empty JSON lists are set.

UniversalNER has been benchmarked on 43 NER datasets across nine domains (general, biomedical, clinical, STEM, programming, social media, law, finance, and transportation), making it the largest NER benchmark to date. Moreover, it surpasses existing instruction-tuned models at the same size, and it especially outperforms ChatGPT in terms of average F1, this demonstrates the efficiency of target distillation (Fig. 21) (Zhang et al. 2024)

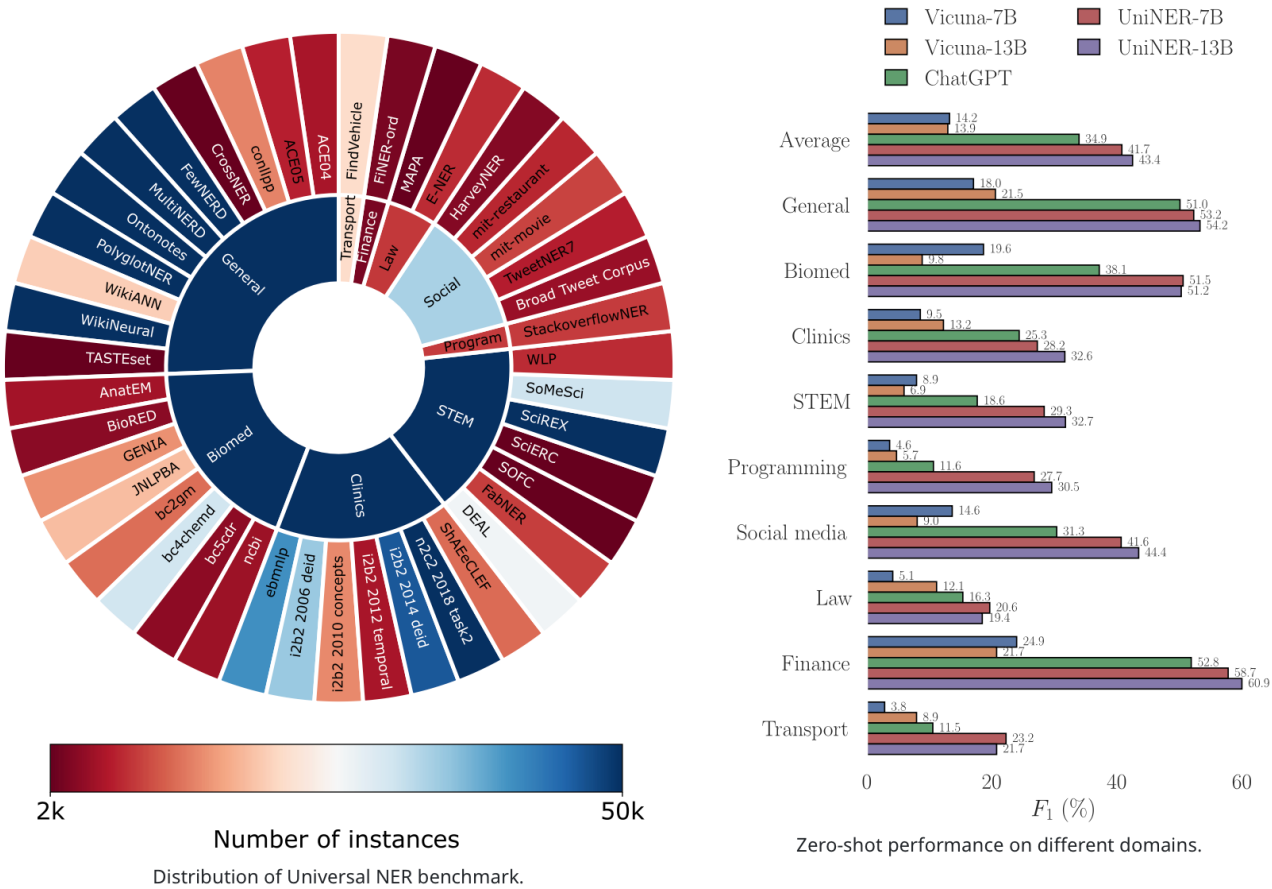


Figure 21: UniversalNER performance.[32]

We've seen that traditional NER models are indeed effective but limited to a set of predefined entity types. At the same time, LLMs can extract arbitrary entities through natural language

[32] https://universal-ner.github.io

instruction, being, in this sense, more flexible even if impracticable due to their cost in a resource-limited scenario.

An innovative open-source model called GLiNER (Generalist Model for Named Entity Recognition using Bidirectional Transformer) has been introduced recently. The models' architecture employs a bidirectional transformer encoder, facilitating parallel entity extraction, which represents an advantage over the sequential token generation of LLMs. The real innovation of the architecture is that the model treats the NER task as a matching problem in a latent space. Entity embeddings are matched to textual spans rather than generating entities token-by-token, making the process more efficient. GLiNER outperforms ChatGPT and fine-tuned LLMs in zero-shot evaluation on NER benchmarks. It is a hybrid approach since it uses a smaller scale BiLM, like BERT or DeBerta (He et al. 2021), and takes as input entity type prompts and text.

GLiNER can process text and extract named entities in a zero-shot setting, meaning it can recognize entities without requiring prior training in a specific dataset, making it adaptable to new domains and languages without any further training. Plus, GLiNER employs less parameters than, for instance, GPT-4. It comes with a user-friendly Python library[33], allowing for quick integration and seamless annotation of named entities in which users can define the entity types of interest and adjust a threshold value to control the number of predicted entities (Fig. 22). Users can define the entity types of interest and adjust a threshold value to control the number of predicted entities. The model processing capabilities enable the annotation of large datasets efficiently, reducing time and cost in production environments. It is efficient for multilingual support since it performs well across different languages (Zaratiana et al. 2023).

These above are some innovative adaptations to address the NER task, the prompting approaches are just some of the many others evolving in this field.

[33] https://github.com/urchade/GLiNER

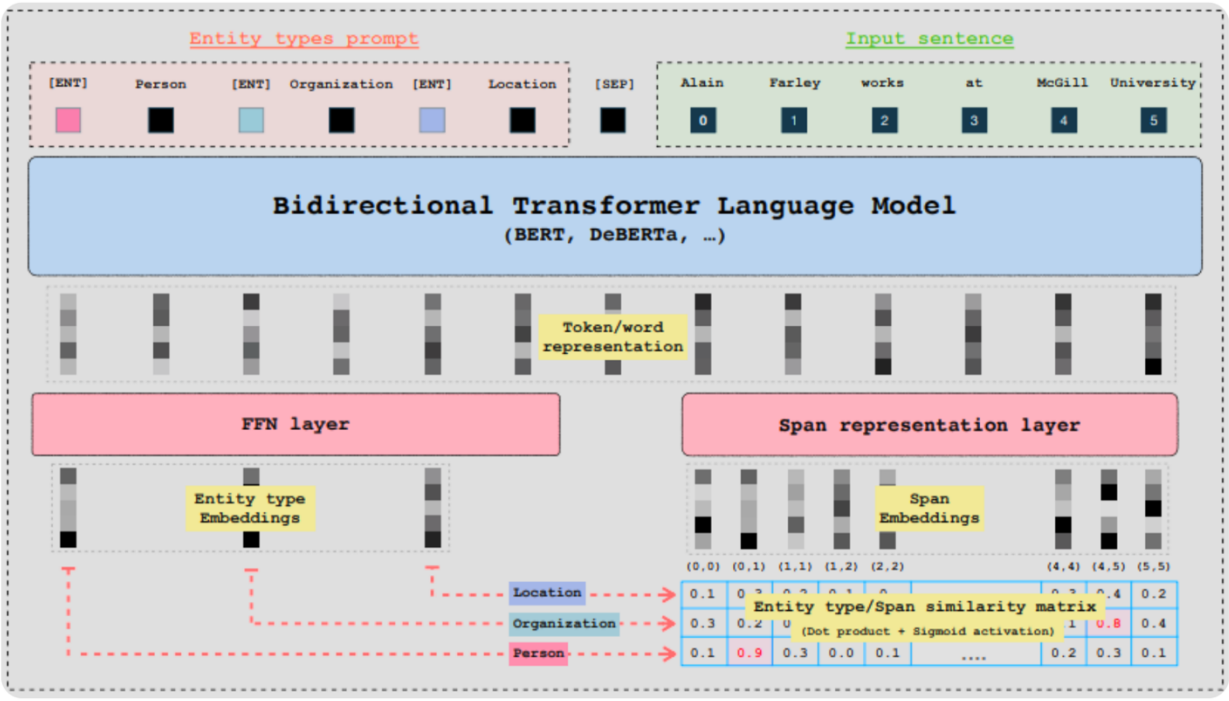


Figure 22: GLiNER's model architecture (Zaratiana et al. 2023)

🛠️ Installation & Usage

Installation

```
!pip install gliner
```

Usage

After the installation of the GLiNER library, import the `GLiNER` class. Following this, you can load your chosen model with `GLiNER.from_pretrained` and utilize `predict_entities` to discern entities within your text.

```
from gliner import GLiNER

# Initialize GLiNER with the base model
model = GLiNER.from_pretrained("urchade/gliner_mediumv2.1")

# Sample text for entity prediction
text = """
Cristiano Ronaldo dos Santos Aveiro (Portuguese pronunciation: [kɾiʃ
"""

# Labels for entity prediction
# Most GLiNER models should work best when entity types are in lowe
labels = ["Person", "Award", "Date", "Competitions", "Teams"]

# Perform entity prediction
entities = model.predict_entities(text, labels, threshold=0.5)

# Display predicted entities and their labels
for entity in entities:
    print(entity["text"], "=>", entity["label"])
```

Expected Output

```
Cristiano Ronaldo dos Santos Aveiro => person
5 February 1985 => date
Al Nassr => teams
Portugal national team => teams
Ballon d'Or => award
UEFA Men's Player of the Year Awards => award
European Golden Shoes => award
UEFA Champions Leagues => competitions
UEFA European Championship => competitions
UEFA Nations League => competitions
European Championship => competitions
```

Figure 23: Installation, code, and expected output.[34]

[34] https://github.com/urchade/GLiNER

4.3 Ethical considerations: data protection, privacy risks, and biases.

Data mining technologies such as NER imply risk factors that can be identified and understood, through legal instruments such as the GDPR[35], EUDPR[36], and EU Charter guidelines[37].

Using a NER system to make decisions or limit individuals' rights such as their access to services, financial products, or contracts can be problematic, especially if those decisions are automated without human intervention. For example, this scenario is risky when systems are used in the health sector: extracting entities from medical records and using that output to make decisions about individuals' health and lives, by large volumes of vulnerable, sensitive data related to healthcare. This is considered a high-level risk factor since it may lead to the violation of patient privacy due to data breaches. Another example may be in the recruitment process for a job position, where NER systems filter curriculum vitae, or in legal and social media contexts, where the system's output influences court filings or content moderation decisions. These technologies always handle highly sensitive information, such as health records, financial details, criminal convictions, behavioral data, and unique identifiers like location data. The more data involved, the greater the impact if there is a breach or mishandling, particularly when the data is sensitive. The processing of data from vulnerable individuals, like children, the elderly, people with mental illness, or those accessing social services requires special protection. Processing their data without the proper safeguards could violate their fundamental rights.

Low data quality represents an issue as well. If the input or training data used in NER systems is of poor quality, it can lead to inaccurate results, such as incorrect identification of named entities. This can have serious consequences, depending on the use case. NER systems are not always 100% accurate and there may be limited high-quality datasets available for all languages and domains.

Generally, if security measures are not sufficient the risks apply to any NER system, especially since most of them are provided as SaaS (Software as a Service) solutions, which may involve transferring data to third-party processors or entities in different jurisdictions, increasing the risks of inadequate data protection.

To mitigate the risks, it is fundamental to ensure that APIs are verified to secure implementation and that data are ad adequately encrypted. Access to the system, data, and storage must be protected and strong user authentication methods must be implemented to prevent unwanted access.

[35] https://www.garanteprivacy.it/regolamentoue
[36] https://eur-lex.europa.eu/eli/reg/2018/1725/oj
[37] https://commission.europa.eu/aid-development-cooperation-fundamental-rights/your-rights-eu/eu-charter-fundamental-rights_en

It is also important to accommodate possible data drift, use reference keys for systems or user profiles, validate data freshness, and periodically monitor system values. Making necessary adjustments and correcting errors is vital to maintain the integrity and accuracy of the data.

NER, context ambiguity is a primary risk that leads to incorrect output. To mitigate this, training sets should include more examples of problematic named entities, and a human review process should be implemented to verify the correctness of sensitive data. Additionally, manual verification should be provided with an overview of the accuracy levels, alongside a redress mechanism for data correction. Managing excessive storage of personal data and ensuring strict adherence to data minimization principles are necessary to comply with GDPR and protect against unlawful transfer of personal data (Barberà 2023).

It is important to be aware of biases in the named entity recognition models, as in many other NLP tasks and AI systems. Biases may be about gender or race, according to a recent analysis (Mehrabi et al. 2019). NER systems tend to misclassify female names as "non-PERSON entities" (Fig. 24), on the contrary, male names are correctly classified. This depends on the training datasets, that may not contain inclusive and representative data.

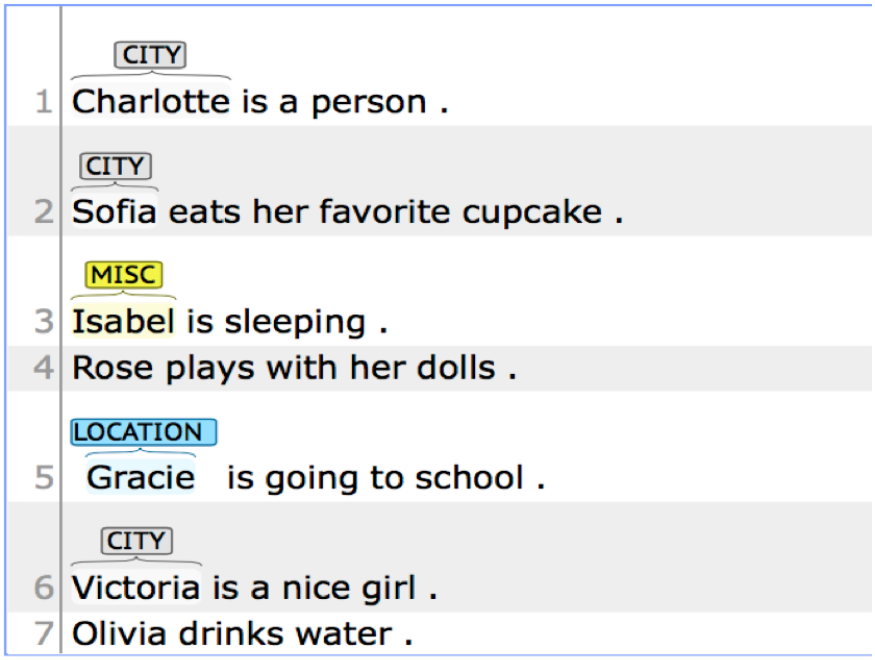


Figure 24: In this example, common female names in the U.S (the authors use a dataset comprising 139 years of U.S. census baby names) are tagged as LOCATION, CITY, and MISC, instead of PERSON. The author shows that the state-of-the-art models at that time often committed this mistake (Mehrabi et al. 2019).

Common NER training datasets like CoNLL-2003 and OntoNotes-5 contain more male names than female names, not reflecting real-world demographics, being biased in this sense.

The evaluation regards five NER models: three versions of spaCy (small, medium, large), Flair, and CoreNLP, which performed on the new benchmark specially created for the evaluation of gender bias. All the models show a higher error rate for female names, and updates sometimes worsen gender bias (Mehrabi et al. 2019). Moreover, some names commonly associated with certain demographic categories like genders or ethnicities are easier to recognize by the system than others. The ELMo model shows performance with fewer biases than the other models, even if it does not eliminate bias. For instance, the accuracy on white names, either male or female, is the highest for most models, while for the ELMo character-base model the accuracy is highest for Muslim male names. As we've already seen, the CoNLL-03 dataset contains many more male names than female names, in the same way, it contains more white names than non-white names (Mishra et al. 2020).

Gender disparities are showed also by biomedical NER models, used to extract chemical mentions from the text for tasks like pharmacoepidemiology and drug discovery. Models like Word2Vec, Flair, BERT/BiOBERT have been tested to evaluate gender bias. The results attest that female-related names, especially names that may be also brand names, are frequently misclassified as chemicals. Furthermore, chemical NER systems often fail to detect some of the drug categories, compared to other drugs, contraceptives, that are commonly associated with female users, are less frequently detected. This is valid also in texts concerning female health, in which hormonal treatments and antidepressants are less frequently identified, on the contrary, in texts related to male issues the error rate is lower in detecting medications and treatments. The objective of future studies is to mitigate these biases, including the investigation of prompting-based NER solutions (Zhao et al. 2022).

.

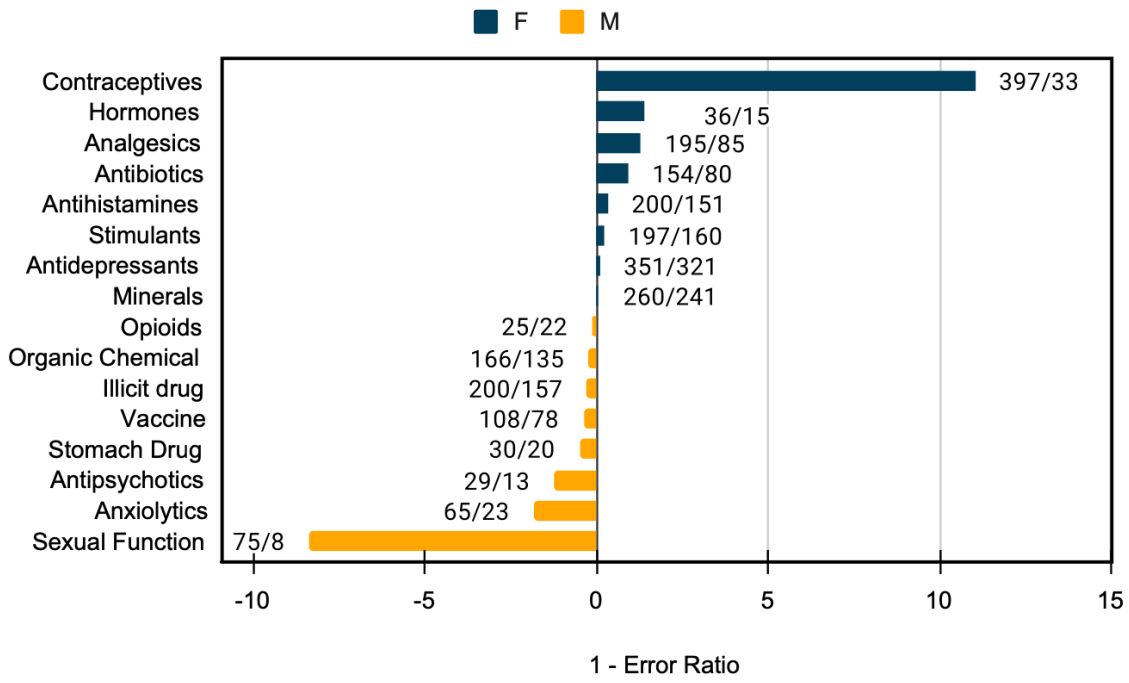


Figure 25: This bar chart shows error ratios among drug recognition performed by a chemical NER model. As we can see, the count of errors is the highest for "Contraceptives" in female-related data. While in categories such as "Sextual Function" and "Anxiolytics" the errors are more frequent for male-associated data (Zhao et al. 2022).

4.4 The process of research through a bibliometric analysis: research hotspots, and low resource NER.

Acknowledging the overall trend of NER research, comprehending the hotspots and their evolution, involved in the process of knowledge, sharing, and interaction in the research community, can be useful to draw the basis for further research.

Before 2000 the number of research papers related to NER was relatively small. In 2005, the publication numbers reached a temporary brief peak, mostly associated with the ACE Conference 2004. Due to the deep learning technologies advancements, specifically, from 2018 onwards the number of publications in the field has increased exponentially, underlying a period of rapid development. From a bibliometric perspective, the NER's most related disciplines, with many papers are related to computer science, artificial intelligence, medicine, and biochemistry. Moreover, most of the papers are published in China and the United States, followed by England, Germany, India, and Japan.

The largest impact in terms of the paper's citations in the field was made by Devlin et al. (2019), which demonstrates the strengths of pre-trained models in NLP research, starting with BERT, determining an important advancement, quickly becoming a trend. The combination of the bidirectional LSTM combined with the CRF for capturing text's long-term dependencies using character level information in the NER task for the first time, presented by Lample et. al. (2016) follows Devlin's paper in terms of popularity. Another fundamental paper is the one published by Vaswani et al. in 2017, which presented the Transformer model and the self-attention and multi-head attention mechanism. This became an undeniable turning point for NLP in general. The Transformer model is for sure more efficient compared to traditional CNNs and RNNs, being able to capture long-distance dependencies by focusing on different parts of the text. More recently, Lee et al. (2020) is the most cited paper, it is the one introducing the BioBERT model (Bidirectional Encoder Representations from Transformers for Biomedical Text Mining). This research shows how the BERT model can be improved and adapted to different domains through pre-training on a large amount of text, in this case in the biomedical domain.

A deeper cluster analysis concludes that the future research directions or named entity recognition mostly include the Pre-training of Large Language Models. The focus is also on progressing with cross-language and cross-domain NER technologies, enabling the transfer and application of knowledge to other languages and domains. These goals are particularly challenging, dealing with linguistic differences and new types of entities that can be found in very specific domains. Another salient research spot is Nested NER. Nested named entities are those entities that contain other entities like "University of Bologna" which is an organization entity that contains "Bologna" which is also a geographical entity (Parkhale et al. 2023). These complications require more sophisticated NER architectures to better understand these context ambiguities. On the other hand, Fine-grained NER is the branch that focuses on classifying entities going beyond the basic types, requiring more sophisticated categorization abilities, often implying the use of multimodal attention. Moreover, Multimodal NER is dedicated to combining text with images and videos, especially in complex and ambiguous social media scenarios (Yang et al. 2024).

In the case of low resources and data scarcity scenarios, Transfer Learning, Data Augmentation, Active Learning, Few-shot learning, and Zero-shot learning are the most renowned techniques that are currently driving the research community.

Transfer learning is a technique involving the pretraining of a model on an extensive corpus and then fine-tuning it on a smaller and more task-specific dataset. In this way, the model learns more context in the first place and improves the performance in specific tasks and domain applications.

Data augmentation is also a method that is focused on generating more data to address the lack of them. This is done using synonyms, paraphrasing, swapping, insertion, deletions, and lexical

substitutions. However, these techniques are more frequent in other NLP tasks rather than NER, since NER makes predictions about words and not sentences, so the transformations of words may change their labels. Moreover, Active Learning is a semi-supervised learning approach in which the model understands which instances are the most uncertain, so that in the process the selected samples are then labeled by humans and the model is then re-trained on the updated dataset. In this way, the smallest amount of data is required.

Recently, few-shot learning experienced exponential growth, the number of research papers increased (Fig. 26), and the interest in low-resource scenarios is growing, especially in domains where collecting and annotating data is problematic, and resources are scarce as well. Most of the publicly available datasets are available just in English, and real-world applications require a long time-consuming process of data annotations, which are vulnerable to errors, and often domain expertise is needed for quality, especially in domains such as healthcare and finance (Moscato et al. 2023).

Few-shot learning involves identifying a small number of entities and annotating a small dataset specific to the domain, covering the contexts in which they are present, and providing a few examples to the pre-trained model. The model learns the prototype and subsequently how to generalize. LLMs advanced few-shot learning in named entity recognition with prompt-based methods.

Furthermore, another important technique is Zero-shot learning, a method in which a pre-trained model learns how to classify a named entity even if it hasn't seen it before in training data. The concept is to provide the model with descriptions or associated words so that the system can detect the entities that correspond to that description and classify them under the correct category (Keraghel et al. 2024).

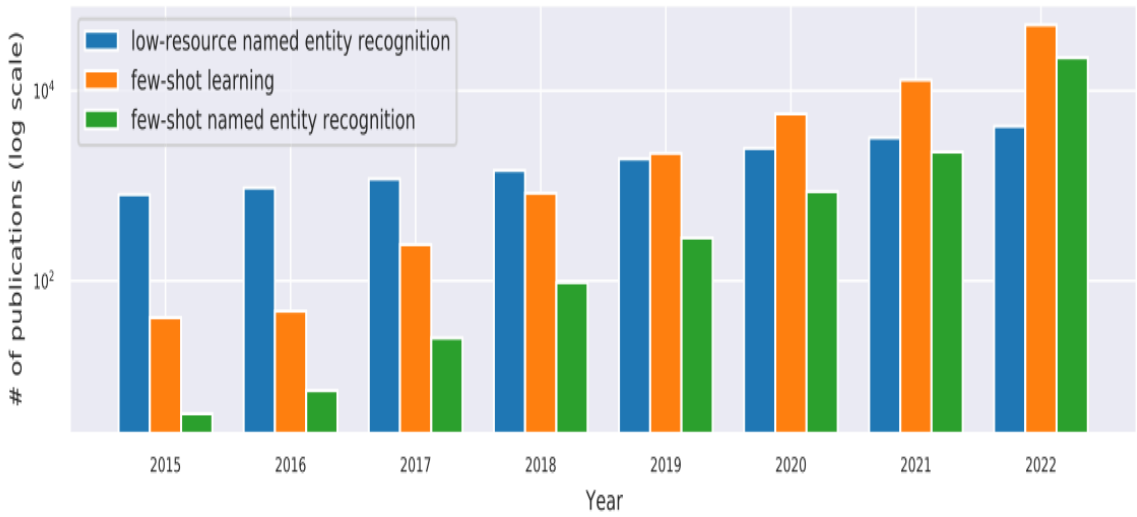


Figure 26: The diagram shows the growth of Google Scholar's publications regarding low-resource NER, few-shot learning, and few-shot NER (Moscato et al. 2023).

CONCLUSION

This thesis describes the evolution of named entity recognition from its roots to its current paradigm, underlying the fundamental characteristics and complexities of the task. First, the systems used for NER were limited to rule-based approaches, rigid on specific domains, being incapable of extending knowledge beyond that. The advent of machine learning advances the progress allowing the NER systems to learn from labeled and unlabeled data, formulating more sophisticated systems. The deep learning evolution shifted the scenario with the implementation of BERT and GPT, strong models able to provide a more accurate performance. In conclusion, the paper explores recent experiments with large language models and prompting, which was one of the goals for this research, since these kinds of approaches are recent attempts, less treated in NER surveys. Ethical considerations about privacy, risks, and biases are treated as well, to spread awareness of the current challenges.

The methods employed for this work involved deep research on the topic, trying to understand its origins and its path across the years. Most of NER's datasets have been collected from popular conferences like MUC[38], CoNLL[39], and ACE[40], so a careful consult of their websites and proceedings has been made to reconstruct the historical evolution of the task. Most of the bibliography has been collected from those websites and ACL Anthology[41], Google Scholar[42], Papers with Code[43], ResearchGate[44], IEEE Xplore[45], ScienceDirect[46], and arXive[47]. Special credits are addressed to recent surveys (Keraghel et. al. 2024), (Jehangir et al. 2023), (Li et al. 2022) carefully ordered and organized, precious for understanding and reconstructing the evolution of named entity recognition. The bibliography has been consulted in chronological order, to understand the workflow and reasons for changing and developing new systems.

Pre-trained language models represent the actual state-of-the-art, however, the challenges aim to provide unbiased and reliable performances. Low-resource NER remains a main challenge as well, experiencing exponential growth in the research community, as well as cross-language and cross-domain NER, to facilitate flexibility and inclusivity.

[38] https://dl.acm.org/conference/muc
[39] https://conll.org
[40] https://www.ldc.upenn.edu/collaborations/past-projects/ace
[41] https://aclanthology.org
[42] https://scholar.google.com
[43] https://paperswithcode.com/sota
[44] https://www.researchgate.net
[45] https://ieeexplore.ieee.org/Xplore/home.jsp
[46] https://www.sciencedirect.com
[47] https://arxiv.org

To offer the most updated research, among all the periods of work on this thesis, a daily consult of the new articles has been done to ensure the inclusion of the most recent papers about LLMs and prompting. This thesis aims to be a resource for those interested in approaching the knowledge of the task and understanding its evolution and its current paradigm.

BIBLIOGRAPHY

Alfonseca, E., and Manandhar, S. 2002. “An Unsupervised Method for Named Entity Recognition Using WordNet.” In *Proceedings of the ACL-02 Workshop on Building and Using Semantic Networks*, 1-7.

Alsentzer, E., Murphy, J., Boag, W., Weng, W., Jindi, D., Naumann, Tristan., McDermott, M. 2019. “Publicly Available Clinical BERT Embeddings.” In *Proceedings of the 2nd Clinical Natural Language Processing Workshop*, pages 72–78.

Ashok, D., and Lipton Z. C. 2023. “Prompt-ner: Prompting for named entity recognition.” *arXiv preprint arXiv:2305.15444*.

Bahdanau, D., Cho, K., and Bengio, Y. 2015. “Neural Machine Translation by Jointly Learning to Align and Translate.” In *International Conference on Learning Representations (ICLR)*.

Barberá, Isabel. 2023 “AI Possible Risks & Mitigations - Named Entity Recognition (NER).” Support Pool of Experts Programmed, European Data Protection Board.

Barskar, Raju, Gulfishan Firdose Ahmed, and Nepal Barskar. 2012. “An Approach for Extracting Exact Answers to Question Answering (QA) System for English Sentences.” *Procedia Engineering* 30: 1187-1194. ISSN 1877-7058. https://doi.org/10.1016/j.proeng.2012.01.979.

Bendel, O., Och, F. J., and Ney, H. 2003. "Maximum Entropy Models for Named Entity Recognition." In *Proceedings of CoNLL-2003*.

Borkowski, Casimir George. 1966. “A System for Automatic Recognition of Names of Persons in Newspaper Texts.”

Borthwick, A. 1999. “A Maximum Entropy Approach to Named Entity Recognition.” New York University.

Chen, T., Moreau, T., Jiang, Z., Zheng, L., Yan, E., Shen, H., and Cowan, M. 2018. “TVM: An Automated End-to-End Optimizing Compiler for Deep Learning.”

Cho, M., Ha, J., Park, C., and Park, S. 2020. “Combinatorial feature embeddings based on CNN and LSTM for biomedical named entity recognition.” In *Journal of Biomedical Informatics*.

Collins, M., and Singer, Y. 1999. "Unsupervised Models for Named Entity Classification." In *Proceedings of the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. 2008. "Natural Language Processing (Almost) from Scratch."

Cucchiarelli, A., and Velardi, P. 2001. "Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence." *Computational Linguistics*.

Derczynski, L., Nichols, E., van Erp, M., Limsopatham, N., and Szymanski, J. 2017. "Results of the WNUT2017 Shared Task on Novel and Emerging Entity Recognition." In *Proceedings of the 3rd Workshop on Noisy User-generated Text (W-NUT)*.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, edited by Jill Burstein, Christy Doran, and Thamar Solorio, 4171-4186. Minneapolis, Minnesota: Association for Computational Linguistics. doi:10.18653/v1/N19-1423.

De Vries, A.P., Vercoustre, A.M., Thom, J.A., Craswell, N., and Lalmas, M. 2008. "Overview of the INEX 2007 Entity Ranking Track." In *Focused Access to XML Documents: INEX 2007*, edited by N. Fuhr, J. Kamps, M. Lalmas, and A. Trotman, 246-251. Lecture Notes in Computer Science, vol. 4862. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-540-85902-4_22.

Dutta, H., and Gupta, A. 2022. "Unsupervised ranking of person name entities from noisy OCR text." In *Decision Support Systems*. https://doi.org/10.1016/j.dss.2021.113662.

Eftimov, T., Koroušić Seljak, B., and Korošec, P. 2017. "A rule-based named-entity recognition method for knowledge extraction of evidence-based dietary recommendations." https://doi.org/10.1371/journal.pone.0179488.

Etzioni, O., Cafarella, M., Downey, D., Popescu, A., Shaked, T., Soderland, S., Weld, D., and Yates, A. 2005. "Unsupervised named-entity extraction from the Web: An experimental study." In *Artificial Intelligence*, Volume 165, Issue 1, 91-134. https://doi.org/10.1016/j.artint.2005.03.001.

Evans, R. 2003. “A framework for named entity recognition in the open domain.” In *Recent Advances in Natural Language Processing*.

Fellbaum, C. 1998. *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press.

Fu, Jinlan, Pengfei Liu, and Graham Neubig. 2020. “Interpretable Multi-dataset Evaluation for Named Entity Recognition.” In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, edited by Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, 6058-6069. Online: Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.489.

Gaizauskas, Azzam, Huyck, Humphreys, Cunningham, and Wilks. 1998. “University of Sheffield: Description of the System as Used for MUC-7.” Department of Computer Science, University of Sheffield.

Gao, S., Kotevska, O., Sorokine, A., and Christian, J. B. 2021. “A pre-training and self-training approach to biomedical named entity recognition.” *PLOS ONE*. https://doi.org/10.1371/journal.pone.0246310.

GDPR – Regolamento 2016/679, “Garante per la protezione dei dati persone”. (retrieved 25, 2024) https://www.garanteprivacy.it/regolamentoue

Google Scholar. “Google Scholar Profile for Ralph Grishman.” Google Scholar Profile for Ralph Grishman. (retrieved March 13, 2024). https://scholar.google.com/citations?user=blwKAkUAAAAJ&hl=en.

Goyal, A. C., and Manish, V. K. 2019. “Analysis of Different Supervised Techniques for Named Entity Recognition.”

Grishman, Ralph, and Beth Sundheim. 1996. “Message Understanding Conference-6: A Brief History.” In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*.

Harris, Z. S. 1954. “Distributional Structure.” In *WORD*, 10(2-3), 146-162.

He, P., Liu, X., Gao, J., Chen, W. 2020. “Deberta: Decoding-enhanced bert with disentangled attention.” arXiv preprint arXiv:2006.03654.

Hu, Y., Ameer, I., Zou, X., Peng, X., Zhou, Y., Li, Z., Li, Y., Li, J., Jiang, X., Xu, H. 2023. "Zero-shot clinical entity recognition using chatgpt." arXiv preprint arXiv:2303.16416

Hu, Z., Hou, W., and Liu, X. "Deep learning for named entity recognition: A survey."

Huang, Z., Wei, X., and Yu, K. 2015. "Bidirectional LSTM-CRF Models for Sequence Tagging." https://arxiv.org/pdf/1508.01991.

Iovine, A., Fang, A., Fetahu, B., Rokhlenko, O., and Malmasi, S. 2022. "CycleNER: An Unsupervised Training Approach for Named Entity Recognition." In *Proceedings of the ACM Web Conference 2022*.

Jehangir, Basra, Saravanan Radhakrishnan, and Rahul Agarwal. 2023. "A survey on Named Entity Recognition — datasets, tools, and methodologies." *Natural Language Processing Journal* 3: 100017. ISSN 2949-7191. https://doi.org/10.1016/j.nlp.2023.100017.

Keraghel, Imed Stanislas Morbieu, and Mohamed Nadif. 2024. "A survey on recent advances in named entity recognition.". https://arxiv.org/abs/2401.10825

Korkontzelos, I., Piliouras, D., Dowsey, A. W., and Ananiadou, S. 2015. "Boosting Drug Named Entity Recognition Using an Aggregate Classifier." *Artificial Intelligence in Medicine* 65(2): 145–153. https://doi.org/10.1016/j.artmed.2015.05.007.

Krupka, George R. 1995. "SRA: Description of the SRA System as Used for MUC-6." In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Kuru, O., Can, O., and Yuret, D. 2016. "CharNER: Character-Level Named Entity Recognition." In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 911-921.

Labusch, K., Kulturbesitz, C., and Zellhofer, D. 2019. "BERT for Named Entity Recognition in Contemporary and Historical German." In *Proceedings of the 15th Conference on Natural Language Processing*, 8-11.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. 2016. "Neural Architectures for Named Entity Recognition." In *Proceedings of the 2016 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, 260-270.

Landauer, T. K., and Dumais, S. T. 1997. “A Solution to Plato’s Problem: The Latent Semantic Analysis Theory of Acquisition, Induction, and Representation of Knowledge.” In *Psychological Review*, 104(2), 211-240.

Li, J., Sun, A., Han, J., and Li, C. 2020. “A Survey on Deep Learning for Named Entity Recognition.” In *IEEE Transactions on Knowledge and Data Engineering*, 1149-1165.

Li, J., Sun, Y., Johnson, R. J., Sciaky, D., Wei, C. H., Leaman, R., ... Lu, Z. 2016. “BioCreative V CDR task corpus: a resource for chemical disease relation extraction.” *Database (Oxford)*. doi: 10.1093/database/baw068. PMID: 27161011; PMCID: PMC4860626.

Liu, X., and Zhou, M. 2013. “Two-stage NER for tweets with clustering.” In *Information Processing and Management*, 264-273. https://doi.org/10.1016/j.ipm.2012.05.006.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. 2019. “RoBERTa: A Robustly Optimized BERT Pretraining Approach.” *CoRR*.

Liu, Zihan. "CrossNER: Evaluating Cross-Domain Named Entity Recognition." Medium. (retrieved April 29, 2024). https://zihanliu1026.medium.com/crossner-evaluating-cross-domain-named-entity-recognition-1a3ee2c1c42b.

Ma, R., Zhou, X., Gui, T., Tan, Y., Li, L., Zhang, Q., andHuang, X. 2022. “Template-free prompt tuning for few-shot NER.” In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5721–5732.

Mansouri, A., Affendey, L., and Mamat, A. 2008. “Named Entity Recognition Approaches.”

Marrero, Mónica, Julián Urbano, Sonia Sánchez-Cuadrado, Jorge Morato, and Juan Miguel Gómez-Berbís. 2013. “Named Entity Recognition: Fallacies, challenges, and opportunities.” *Computer Standards & Interfaces* 35, no. 5: 482-489. ISSN 0920-5489. https://doi.org/10.1016/j.csi.2012.09.004.

Mayfield, J., McNamee, P., and Piatko, C. 2003. “Named Entity Recognition using Hundreds of Thousands of Features.” In *Proceedings of the Seventh Conference of Natural Language Learning at HTL-NAACL 2003*, 184-187.

Martin, L., Muller, B., Suarez, P., Dupont, Y., Romary, L., de la Clergerie, E., Seddah, D., and Sagot, B. 2020. “CamemBERT: A Tasty French Language Model.” In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 7203-7219.

McCallum, A., and Li, W. 2003. “Early results for Named Entity Recognition with Conditional Random Fields, Feature Induction and Web-Enhanced Lexicons.” In *Proceedings of CoNLL-2003*.

Mehrabi, N., Thamme, G., Fred, M., Nanyun P., Aram G. 2019. "Man is to Person as Woman is to Location: Measuring Gender Bias in Named Entity Recognition." https://arxiv.org/abs/1910.10872.

Mikheev, Andrei, Grover, Claire, and Moens, Marc. 1998. “Description of the LTG System Used for MUC-7.” HCRC Language Technology Group, University of Edinburgh.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. 2013. “Efficient Estimation of Word Representations in Vector Space.”

Mishra, S., He, S., and Belli, L. 2020. “Assessing Demographic Bias in Named Entity Recognition.” *ArXiv, abs/2008.03415*.

Morwal, S. 2012. "Named Entity Recognition using Hidden Markov Model (HMM)." *International Journal on Natural Language Computing*.

Mosallam, Y., Abi-Haidar, A., and Ganascia, J.-G. 2014. “Unsupervised Named Entity Recognition and Disambiguation: An Application to Old French Journals.” In *Advances in Data Mining: Applications and Theoretical Aspects*.

Vincenzo Moscato, Marco Postiglione, and Giancarlo Sperlí. 2023. “Few-shot Named Entity Recognition: Definition, Taxonomy and Research Directions.” ACM Trans. Intell. Syst. Technol. https://doi.org/10.1145/3609483

Oudah, M., and Shaalan, K. 2012. “A Pipeline Arabic Named Entity Recognition Using a Hybrid Approach.” In *Proceedings of the 24th International Conference on Computational Linguistics: Technical Papers (COLING 2012)*.

Palshikar, Girish. 2012. Techniques for Named Entity Recognition: A Survey.

Palmer, David, and David Day. 2002. “A Statistical Profile of the Named Entity Task.” In *Proceedings of the 6th Conference on Computational Natural Language Learning (CoNLL-2002).* doi:10.3115/974557.974585.

Pakhale, Kalyani. 2023 “Comprehensive Overview of Named Entity Recognition: Models, Domain-Specific Applications and Challenges.” *ArXiv, abs/2309.14084*.

Pasca, M., Bigham, J., Lifchits, A., and Jain, A. 2006. “Organizing and Searching the World Wide Web of Facts – Step One: The One-Million Fact Extraction Challenge.” In *Proceedings of the National Conference on Artificial Intelligence*.

Peng, Q., Zheng, C., Cai, Y., Wang, T., Xie, H., and Li, Q. 2021. “Unsupervised cross-domain named entity recognition using entity-aware adversarial training.” In *Neural Networks*, 68-77. https://doi.org/10.1016/j.neunet.2020.12.027.

Pennington, J., Socher, R., and Manning, C. 2014. “GloVe: Global Vectors for Word Representation.”

Petasis, G., Vichot, F., Wolinski, F., Paliouras, G., Spyropoulos, C., and Karkaletsis, V. 2001. “Using Machine Learning to Maintain Rule-based Named-Entity Recognition and Classification Systems.” In *Proceedings of the Conference*, 418-425. doi: 10.3115/1073012.1073067.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. "Deep contextualized word representations." arXiv:1802.05365 [cs.CL], version 2. https://doi.org/10.48550/arXiv.1802.05365.

Petkova, Desislava, and W. Bruce Croft. 2007. “Proximity-based document representation for named entity retrieval.” In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management (CIKM '07)*, edited by Association for Computing Machinery, 731-740. New York, NY, USA: Association for Computing Machinery.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. 2018. “Improving Language Understanding by Generative Pre-Training.” OpenAI

Rahman, F., and Bowles, J. 2021. “Semantic annotations in clinical guidelines.” In *From Data to Models and Back. 9th International Symposium. DataMod 2020, Virtual Event, October 20, 2020. Revised Selected Papers*, edited by Springer International Publishing, 190-205. Springer International Publishing.

Ramshaw, A. L., and Mitchell, P. M. 1995. "Text Chunking Using Transformation-Based Learning." In *Proceedings of the Third ACL Workshop on Very Large Corpora*, 82-94.

Ramshaw, Lance A., and Mitchell P. Marcus. 1999. “Text chunking using transformation-based learning.” In *Natural Language Processing Using Very Large Corpora*, edited by Springer Netherlands, 157-176. Dordrecht: Springer Netherlands.

Rau, L. F. 1991. “Extracting Company Names from Text.” In *7th Conference on Artificial Intelligence Applications*, 29-32. doi:10.1016/0004-3702(93)90016-5.

Riloff, E., and Jones, R. 1999. “Learning Dictionaries for Information Extraction using Multi-level Bootstrapping.” In *Proceedings of the National Conference on Artificial Intelligence*.

Saha, S., and Ekbal, A. 2013. “Combining multiple classifiers using vote-based classifier ensemble technique for named entity recognition.” *Data & Knowledge Engineering* 15-39. https://doi.org/10.1016/j.datak.2012.06.003.

Sang, Erik F. Tjong Kim, and Fien De Meulder. 2003. “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition.” *Proceedings of the Seventh Conference on Natural Language Learning*, 142-147.

SemEval-2022 Task 1: Open-Domain Semantic Parsing for Text-to-SQL. 2022. In *Proceedings of the Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*, 196-209. Association for Computational Linguistics.

Sekin, S., Grishman, R., and Shinnou, H. 1998. “A Decision Tree Method for Finding and Classifying Names in Japanese Texts.” In *Proceedings of the Sixth Workshop on Very Large Corpora*.

Shinyama, Y., and Sekine, S. 2004. “Named Entity Discovery Using Comparable News Articles.” In *COLING 2004: Proceedings of the 20th International Conference on Computational Linguistics*, 848-853.

Smith, Reid. 1985. *Knowledge-Based Systems Concepts, Techniques, Examples*. Schlumberger-Doll Research. (retrieved November 9, 2013). reidgsmith.com.

Sunheim, Beth M. "Overview of Results of MUC-6 Evaluation." Naval Command, Control and Ocean Surveillance Center RDT and E Division (NRaD) Information Access Technology Project Team.

Tamburini, Fabio. 2020. “How ‘BERTology’ Changed the State-of-the-Art Also for Italian NLP.” In *Proceedings of the Workshop at Aachen: CEUR-WS.*

Tamburini, Fabio. 2022. “Neural Models for the Automatic Processing of Italian.” Bologna: Pàtron.

Tjong Kim Sang, Erik F. 2002. “Introduction to the CoNLL-2002 Shared Task: Language-Independent Named Entity Recognition.” In *COLING-02: The 6th Conference on Natural Language Learning 2002*.

Università degli Studi di Bologna. 2014. "Named Entity Extraction: la piattaforma Gate/Annie." Laurea Triennale thesis, Scuola di Scienze, Corso di Laurea in Ingegneria e Scienze Informatiche. Advisor: Prof. Dario Maio. Candidate: Luca Campana.

Vaswani, A., Shazeer, N., Parmar, N., Usszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. 2017. “Attention is all you need.” In *31st Conference on Neural Information Processing Systems (NIPS 2017)*.

Wang, Shuhe, Xiaofei Sun, Xiaoya Li, Rongbin Ouyang, Fei Wu, Tianwei Zhang, Jiwei Li and Guoyin Wang. 2023. “GPT-NER: Named Entity Recognition via Large Language Models.” *ArXiv* abs/2304.10428

Warto, Supriadi Rustad, Guruh Fajar Shidik, Edi Noersasongko, Purwanto, Muljono, and De Rosal Ignatius Moses Setiadi. 2024. “Systematic Literature Review on Named Entity Recognition: Approach, Method, and Application”. *Statistics, Optimization & Information Computing* 12 (4), 907-42. https://doi.org/10.19139/soic-2310-5070-1631.

Wilks, Yorick. 1997. “Information Extraction as a core language technology: What is IE?”. https://api.semanticscholar.org/CorpusID:10237124.

Yang, Jun, Taihua Zhang, Chieh-Yuan Tsai, Yao Lu, and Liguo Yao. 2024. "Evolution and Emerging Trends of Named Entity Recognition: Bibliometric Analysis from 2000 to 2023." In *Heliyon* https://doi.org/10.1016/j.heliyon.2024.e30053.

Zaratiana, U., Tomeh, N., Holat, P., Charnois, T. 2023. "GLiNER: Generalist Model for Named Entity Recognition using Bidirectional Transformer." https://arxiv.org/abs/2311.08526.

Zhang, S., and Elhadad, N. 2013. “Unsupervised biomedical named entity recognition: Experiments with clinical and biological texts.” In *Journal of Biomedical Informatics*, 1088-1098. https://doi.org/10.1016/j.jbi.2013.08.004.

Zhang, S., L. Dong, X. Li, S. Zhang, X. Sun, S. Wang, J. Li, R. Hu, T. Zhang, F. Wu. 2024 "Instruction Tuning for Large Language Models: A Survey.” arXiv:2308.10792, 2023.

Zhang, T., and Johnson, D. 2003. "A Robust Risk Minimization based Named Entity Recognition System." In *Proceedings of CoNLL-2003*.

Zhao, S. 2004. “Named Entity Recognition in Biomedical Texts using an HMM Model.” In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and Its Applications*, 87-90.

Zhao, X., Niazi, A., Rios, A. 2022. “A Comprehensive Study of Gender Bias in Chemical Named Entity Recognition Models.”

Zhou, G., and Jian, S. 2002. “Named Entity Recognition using an HMM-based Chunk Tagger.” In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 473-480.

Zhou, W., Zhang, Zhang S., Gu, Yu., Chen, M., Poon, H. 2024. “UniversalNER: Targeted Distillation from Large Language Models for Open Named Entity Recognition.” In *Proceedings of the International Conference on Learning Representations (ICLR) 2024*. https://universal-ner.github.io/

SITOGRAPHY

ACM. Date. “Multimodal Understanding of Command Input”. ACM Digital Library. (retrieved March 13, 2024). https://dl.acm.org/conference/muc.

Apache OpenNLP. “Home”. Apache OpenNLP. (retrieved March 20, 2024). https://opennlp.apache.org

Bergmann, Dave. “What is instruction tuning?” IBM, April 5 2024. (retrieved September 5, 2024). https://www.ibm.com/topics/instruction-tuning.

Chynun Y. “How to Find the Optimal Number of Clusters with R.” *Medium* (retrieved June 15, 2024). https://medium.com/@chyun55555/how-to-find-the-optimal-number-of-clusters-with-r-dbf84988388bù

DARPA. “Defense Advanced Research Project Agency.” Defence Advanced Research Project Agency. (retrieved March 13, 2024). https://www.darpa.mil.

Ding, Ning. 2021. "Few-NERD: A Few-shot Named Entity Recognition Dataset." Accessed (retrieved September 25, 2024). https://ningding97.github.io/fewnerd/.

European Commission "EU Charter of Fundamental Rights," (retrieved September 25, 2024). https://commission.europa.eu/aid-development-cooperation-fundamental-rights/your-rights-eu/eu-charter-fundamental-rights_en.

EUR-Lex “Regulation (EU) 2018/1725 of the European Parliament and of the Council of 23 October 2018," EUR-Lex, (retrieved September 25, 2024) https://eur-lex.europa.eu/eli/reg/2018/1725/oj.

EVALITA. "EVALITA 2009." EVALITA. (retrieved March 18, 2024). https://www.evalita.it/campaigns/evalita-2009/

Github. “urchade/GliNER. Github. (Retrieved September 5, 2024). https://github.com/urchade/GLiNER.

Grishman, R. 1996. “Message Understanding Conference – MUC-6.” NYU Computer Science. (retrieved March 11, 2024). https://cs.nyu.edu/~grishman/muc6.html.

Google Scholar. “Google Scholar Profile for Ralph Grishman.” Google Scholar Profile for Ralph Grishman. (retrieved March 13, 2024). https://scholar.google.com/citations?user=blwKAkUAAAAJ&hl=en.

Hugging Face. 2024. "bionlp2004 Dataset." Hugging Face Datasets. Retrieved from https://huggingface.co/datasets/tner/bionlp2004.

Hugging Face. “Home.” Hugging Face (retrieved March 20, 2024). https://huggingface.co

Hugging Face. 2022. “MIT Restaurant.” Hugging Face Datasets. (retrieved March 18, 2024). https://huggingface.co/datasets/tner/mit_resturant.

Hugging Face. “TNER – Hugging Face”. Hugging Face (retrieved March 20, 2024). https://huggingface.co/tner.

Invgate. "GPT-3 vs BERT: What Are They and What Are Their Differences?" Invgate Blog. (Retrieved September 2, 2024) https://blog.invgate.com/gpt-3-vs-bert.

JavaTpoint. “Supervised machine learning” (retrieved June 15, 2024). https://www.javatpoint.com/supervised-machine-learning.

Linguistic Data Consortium. "LDC." (retrieved March 18, 2024). https://catalog.ldc.upenn.edu/LDC2013T19

Manetti. 2024. "WikiGold Dataset." Manetti. (retrieved March 18, 2024). https://www.manetti.com/wikigold/.

Manuela Speranza. 2007. “EVALITA 2007: IL TASK RICONOSCIMENTO DI ENTITA’ NOMINATE”. EVALITA. (retrieved March 17, 2024). https://www.evalita.it/wp-content/uploads/2021/11/29-IA-IV-2-ner-task.pdf

MTSamples. “Transcribed Medical Transcription Sample Reports and Examples”. MTSamples. (retrieved September 25, 2024). https://mtsamples.com

National Center for Biotechnology Information (NCBI). Unknown. "NCBI BioNLP Disease Corpus." (retrieved March 18, 2024). https://www.ncbi.nlm.nih.gov/research/bionlp/Data/disease/

NLTK. “Documentation”. NLTK. (retrieved March 20, 2024). https://www.nltk.org

OpenAI. “Home”. OpenAI. (retrieved March 19, 2024). https//openai.com

OpenAI. “ChatGPT”. https://openai.com/chatgpt

Papers with Code. 2024. "fin Dataset." Papers with Code. (retrieved March 18, 2024) https://paperswithcode.com/dataset/fin.

Papers with Code. ““https://paperswithcode.com/dataset/few-nerd

Polyglot Edu. “Home.” Polyglot Edu. (retrieved March 20, 2024). https://polyglot-edu-eu/en.

RALI. 2024. "WikiCoref: An English Coreference-annotated Corpus of Wikipedia Articles" RALI. (retrieved March 18, 2024) http://rali.iro.umontreal.ca/rali/?q=en/wikicoref.

Rau, L. “L. Rau’s publications”. Semantic Scholar. (retrieved March 13, 2024), https://dl.acm.org/do/10.1145/contrib-99659073662/full/.

R Project. “Home.” R Project. (retrieved March 20, 2024). https://www.r-project.org.

Shaip. “Clinical Named Entity Recognition (NER).” Shaip. (retrieved September 26, 2024). https://it.shaip.com/solutions/clinical-named-entity-recognition-ner/.

Scikit-learn. “2.3 Clustering – scikit-learn 1.5.0 documentation”. (retrieved May 28, 2024). https://scikit-learn.org/stable/modules/clustering.html

Scikit-learn. “scikit-learn: machine learning in Python”. (retrieved May 28, 2024). https://scikit-learn.org/stable/.

Singla, Niraj. "Using Named Entity Recognition to Improve Machine Translation" Stanford University. (retrieved April 16, 2024). https://nlp.stanford.edu/courses/cs224n/2010/reports/singla-nirajuec.pdf

Spacy. “Home”. Spacy, (retrieved March 19, 2024). https://spacy.io.

Spacy. “Rule-base matching.” (retrieved May 26, 2024). https://spacy.io/usage/rule-based-matching

Stanford CoreNLP. “Home”. Github. (retrieved March 20, 2024). https://github.com/stanfordnlp/CoreNLP.

Tan P.S. 2021."Introduction to Hidden Markov Model (HMM) with Simple NER." *Medium*, (retrieved June 15, 2024) https://tanpengshi.medium.com/introduction-to-hidden-markov-model-hmm-with-simple-ner-d1353ff35842.

TechTarget. “Named entity recognition (NER).” (retrieved March 21, 2024). https://www.techtarget.com/whatis/definition/named-entity-recognition-NER.

TEI Consortium. “Text Encoding Initiative Consortium.” Text Encoding Initiative Consortium. (retrieved March 13, 2024). https://tei-c.org

TREC. National Institute of Standards and Technology. https://trec.nist.gov. (retrieved March 19, 2024).

UniversalNER. 2023. “UniversalNER: Targeted Distillation from Large Language Models for Open Named Entity Recognition.” (retrieved September 5, 2024). https://universal-ner.github.io.

VAERS "Vaccine Adverse Event Reporting System," U.S. Department of Health & Human Services, (retrieved September 25, 2024). https://vaers.hhs.gov/index.html.

Vitali, Fabio. “SGML (Standard Generalized Markup Language) Introduction.” University of Bologna. http://www.cs.unibo.it/~fabio/corsi/sdc/SGMLintro.pdf.

WordNet. Prienceton University. 2023. “Wordnet: a Lexical Database for English” (retrieved June 12, 2024). https://wordnet.princeton.edu