# LlamATE: Automated Term Extraction Using Large-scale Generative Language Models

Tran Thi Hong Hanh, Carlos-Emiliano González-Gallardo, Antoine Doucet, Senja Pollak

# LlamATE: Automated Term Extraction Using Large-scale Generative Language Models

Hanh Thi Hong Tran[1][0000−0002−5993−1630], Carlos-Emiliano González-Gallardo[2][0000−0002−0787−2990], Antoine Doucet[3][0000−0001−6160−3356], and Senja Pollak[4,5][0000−0002−4380−0863]

[1] ARKHN, Paris, France
hanh.tran@arkhn.com
[2] University of Tours, LIFAT, Tours, France
gonzalezgallardo@univ-tours.fr
[3] University of La Rochelle, L3i, La Rochelle, France
antoine.doucet@univ-lr.fr
[4] Jožef Stefan International Postgraduate School, Ljubljana, Slovenia
[5] Jožef Stefan Institute, Ljubljana, Slovenia
senja.pollak@ijs.si

**Abstract.** Over the past decades, automatic term extraction (ATE), a natural language processing (NLP) task that aims to identify terms from specific domains by providing a list of candidate terms, has been challenging due to the strong influence of domain-specific differences on term definitions. Leveraging the advances of large-scale language models (LLMs), we propose *LlamATE*, a framework to verify the impact of domain specificity on ATE when using in-context learning prompts in open-sourced LLM-based chat models, namely *Llama-2-Chat*. We evaluate how well the LLM-based chat (e.g., using reinforcement learning with human feedback (RLHF)) models perform with different levels of domain-related information in the dominant language in NLP research (e.g., English) and other European languages (e.g., French, Slovene) from ACTER datasets, i.e., in-domain and cross-domain demonstrations with and without domain enunciation. Furthermore, we examine the potential of cross-lingual and cross-domain prompting to reduce the need for extensive data annotation of the target domain and language. The results demonstrate the potential of implicit in-domain learning where examples of the target domain are used as demonstrations for the prompts without specifying the domain of each example, and cross-lingual learning when knowledge is transferred from the dominant to lesser-represented European languages as for the data used to pre-train the LLMs. *LlamATE* also offers a valuable compromise by reducing the need for extensive data annotation, making it suitable for real-world applications where labeled corpora are scarce. The code is publicly available at https://github.com/honghanhh/terminology2024.

**Keywords:** Term extraction · LLMs · Prompt engineering · In-context learning · Llama-2-Chat · Cross-domain · Transfer learning · Self-verification

## 1 Introduction

Throughout history, term extraction has been a challenging task due to the strong influence of domain-specific differences on term definitions, which distinguishes it from general language. The International Organization for Standardization (ISO) considered a term as "*the designation*[6] *of a defined concept*[7] *in a special language by a linguistic expression.*" and defined the process of term extraction as: " *terminology work*[8] *that involves the identification and excerption of terminological data*[9] *by searching through a text corpus*[10]" (ISO:1087, 2019). Terms are beneficial not only for several terminographical tasks performed by linguists (e.g., glossary construction (Maldonado and Lewis, 2016) and specialized dictionary construction (Le Serrec et al., 2010)), but also for several complex downstream tasks (e.g., topic detection (El-Kishky et al., 2014), machine translation (Wolf et al., 2011), text summarization (Litvak and Last, 2008), information retrieval (Lingpeng et al., 2005), ontology engineering and learning (Biemann and Mehler, 2014), and sentiment analysis (Pavlopoulos and Androutsopoulos, 2014)). However, manual term extraction is often time- and labor-consuming. Therefore, several automatic term extraction (ATE) approaches have been proposed to identify the candidate terms from domain-specific corpora by providing a list of candidate terms (see examples of term extraction results in Figure 1).

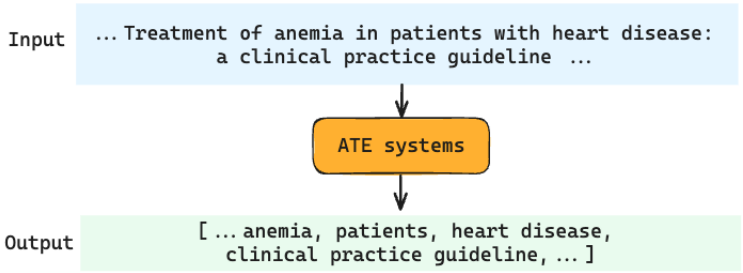


Fig. 1: An example of an ATE system output from a given input sentence.

*TermEval 2020: Shared Task on Automatic Term Extraction*, organized as part of the CompuTerm workshop (Rigouts Terryn et al., 2020a), presented an important step forward in systematic comparison among several ATE systems with the introduction of a new manually-annotated corpus, namely "*The Anno-*

[6] "Representation of a concept by a sign which denotes it in a domain or subject".
[7] "Unit of knowledge created by a unique combination of characteristics".
[8] "The work concerned with the systematic collection, description, processing and presentation of concepts".
[9] "The data related to concepts and their designations".
[10] "A collection of natural language".

*tated Corpora for Term Extraction Research*" (ACTER) corpora (Rigouts Terryn et al., 2020a). The corpora contain domain-specific texts from four fields in three languages (i.e., English, French, and Dutch) with two versions of gold standards (with and without named entities). This is also the corpora on which we conduct our experiments to verify our hypothesis on the impact of domain-specificity and the potential of cross-lingual transfer and cross-domain transfer in the era of large-scale language models (LLMs).

After the evolution of transformer-based token classifiers toward term extraction (e.g., BERT (Lang et al., 2021) and XLMR (Tran et al., 2022c,a,b,c)), recent years have witnessed the blossoming of LLMs and their reinforcement learning with human feedback (RLHF) versions exemplified by both close-sourced (e.g., ChatGPT[11]) and open-sourced (e.g., *Llama-2-Chat*[12]) models. Although these models have undergone extensive training using diverse datasets from various domains and have showcased remarkable competitiveness across several complex downstream NLP tasks (Kocoń et al., 2023; Guo et al., 2023) regarding both prompt engineering (Radford et al., 2019) and fine-tuning (Dettmers et al., 2024), no application has been found on term extraction except our initial work (Tran et al., 2024a) yet, leaving a gap in the terminology research.

The main contribution of our work is four-fold:

1. We present and compare the most recent advances in prompting techniques when applied to ATE in English, French, and Dutch. To the best of our knowledge, this is one of the first works focusing on ATE prompt engineering for languages other than English.
2. We investigate the potential of in-context learning (ICL) or few-shot demonstrations with prompt engineering in LLM-based chat models, namely *Llama-2-Chat* (i.e., *Llama-2-Chat-70b-chat-hf*) for ATE in several languages.
3. We empirically evaluate how the domain impacts the performance of ATE given the same few-shot ($k$-shot) examples to LLM-based chat models using four distinct prompting approaches: (1) explicit in-domain $k$-shot demonstration; (2) explicit cross-domain $k$-shot demonstration; (3) implicit in-domain $k$-shot demonstration; and (4) implicit cross-domain $k$-shot demonstration.
4. We conduct an extensive cross-lingual study of term extraction in the Indo-European languages to assess the potential of prompting when the annotated data is scarce.

This paper is organized as follows: Section 2 presents the related work, while Section 3 describes our chosen datasets. Section 4 introduces the overall architecture of *LlamATE* and discusses the individual components before describing the experimental setup and evaluation metrics. The results with the error analysis are shown in Section 5, followed by the discussion on the error analysis, practical use of *LlamATE* and limitations in Section 6, before concluding with future work in Section 8.

[11] https://platform.openai.com/docs/models
[12] https://Llama.meta.com/Llama2/

## 2 Related work

In early research on term extraction, Kageura and Umino (1996) identified two main approaches: linguistic and statistical. A more comprehensive survey by da Silva Conrado et al. (2014) categorized ATE methods into three feature-based sub-categories: statistical, linguistic, and hybrid. Meanwhile, Astrakhantsev et al. (2015) focused on the dominant languages (i.e., English) and provided a detailed overview of term definitions, linguistic features, and numerous extraction methods with a generalized four-stage pipeline: preprocessing, term candidate collecting, term candidate scoring, and term candidate ranking. With the advent of deep learning, Tran et al. (2023) thoroughly summarized recent advances in term extraction with respect to three aspects: well-annotated corpora, approaches, and evaluation metrics. The key findings were that neural models generally outperformed feature-based machine learning (ML) models by a wide margin, and that transformer-based cross-domain and cross-lingual models generally performed well and set new benchmarks.

### 2.1 Machine learning approaches

Due to their relatively low accuracy, these first ML approaches were mainly used to complement approaches based on hand-crafted rules. This, along with the success of traditional systems (e.g., TermoStat (Drouin, 2003)), which relied on several linguistic and statistical features, led to the idea of combining different types of information. While several NLP tools (e.g., tokenization, lemmatization, stemming, chunking, and PoS tagging) were employed in this approach to obtain linguistic profiles of term candidates, numerous statistical measures were also applied, including termhood (Vintar, 2010), unithood (Daille et al., 1994), C-value (Frantzi et al., 1998). Regarding ML algorithms, the most popular algorithms used for ATE included AdaBoost (Castellví et al., 2001), ROGER evolutionary algorithm (Azé et al., 2005), RIPPER rule induction (Foo and Merkel, 2010), CRF++ (Judea et al., 2014), K-nearest neighbors (Qasemizadeh and Handschuh, 2014), Logistic Regression (Bolshakova et al., 2013; Fedorenko et al., 2014), Decision Trees (DTs) (Karan et al., 2012), and Support Vector Machines (SVM) (Ljubešić et al., 2018).

An example of an ML approach employing extensive feature engineering and several classifiers was given in Conrado et al. (2013) where the authors proposed to select statistical and linguistic features and feed them into different ML classifiers (e.g., JRip, Naive Bayes, J48, or SMO from WEKA). Yuan et al. (2017) instead used common features (e.g., term frequency, c-value, weirdness) extracted from the token n-grams ($n = \{1, 2, 3, 4, 5\}$) excluding all the stopwords and fed them into different classifiers: Random Forest (RF), Linear SVM, Multinomial Naive Bayes, Logistic Regression, and SGD classifiers. Another example was given in Hazem et al. (2020), which used the combination of various types of meaningful information—such as linguistic, stylistic, statistical, and distributional descriptors—to generate the feature vectors. It then used the XGBoost

model to learn several classifiers, which were weighted according to their performance and iteratively aggregated. HAMLET (Rigouts Terryn et al., 2021), on the other hand, included a relatively wide range of supervised ML methods (e.g., Multi-layer Perceptron, and Logistic Regression) and relied on a total of 152 features from six different feature groups: morphological, frequency-based, statistical, relational and linguistic, and corpus-based. The Binary RF classifier performed best of all the classifiers tested. In addition, Nugumanova et al. (2022) combined probabilistic topic modeling (PTM) and non-negative matrix factorization (NMF). They compared five different NMF algorithms and four different NMF initializations and found optimal NMF combinations for comparison with the extraction baseline (e.g., TF-IDF, RAKE, YAKE, and TextRank).

### 2.2 Neural approaches

The use of neural networks, especially language models, to solve term extraction tasks has become increasingly important in recent years. They have been used to represent the information in the text with word embeddings or to apply a deep architecture as an end-to-end classifier. Traditional automatic term extractors relied heavily on term characteristics (Repar et al., 2019) or neural non-contextual (e.g., GloVe[13] (Kucza et al., 2018; Le and Sadat, 2021), Word2Vec (Zhang et al., 2018; Bay et al., 2021), and FastText (Terryn et al., 2022)) and contextual (e.g., Flair + BERT (Andrius, 2020)) embeddings to represent the term information and adapt the ML classifier to extract the candidate terms (Rigouts Terryn et al., 2021). With the advent of neural language models (LMs) and LLMs, different mechanisms have been proposed to solve this task more efficiently.

*Tagging-based Models* : Initially, ATE was formulated as a sequence classification task, which assigns binary label $y \in Y = \{is\_a\_term, not\_a\_term\}$ to each sequence $s$ in a given sentence $S = \{s_1, ..., s_n\}$, where $n$ denotes the number of sequences generated from all possible n-grams of a fixed length of a given sentence. Different BERT-based variants have been tested including RoBERTa, CamemBERT, and XLMR (Hazem et al., 2020; Lang et al., 2021). However, generating all possible n-grams from each sentence across all documents posed a computational and storage challenge. As a result, token classification was proposed as an alternative strategy.

The token classifier assigns a label $y \in Y = \{B, I, O\}$ to each word $x$ in a given sentence $X = \{x_1, ..., x_n\}$, where $Y$ denotes the set of labels in the BIO annotation regime, and $n$ denotes the length of the given sentence. Several language models were applied, including both non-transformers (e.g., RNN (Kucza et al., 2018), LSTM-CNN (Wang et al., 2016), CNN-BiLSTM-CRF (Han et al., 2018), and (Bi)LSTM-CRF (Andrius, 2020; Rigouts Terryn et al., 2022; Hazem et al., 2022)) and transformers models (e.g., RoBERTa (Tran et al., 2022b; Delaunay et al., 2024), CamemBERT (Tran et al., 2022b), RobBERT (Tran et al., 2022b), SloBERTa (Tran et al., 2022b), and XLMR (Hazem et al.,

[13] https://nlp.stanford.edu/projects/glove/

2022; Tran et al., 2022a,b, 2024b)). Besides the fully supervised classifier, cross-domain (Lang et al., 2021; Tran et al., 2022c) and cross-lingual (Tran et al., 2022a; Hazem et al., 2022; Tran et al., 2024b) learning were also applied to enhance the extraction performance given the lack of available annotated data in the target domain and language, respectively. The results suggested that the models were still domain-sensitive and reconfirmed the potential of knowledge transfer from the dominant to lesser-represented languages.

*Span-based Models* : Given an input sentence $X = \{x_1, ..., x_n\}$ where $x_i$ denotes the $i^{th}$ word, the span-based term extractor aims to predict the start and end position of each term span in $X$. Wang et al. (2023a) used XLMR as an encoder and operated the system in a two-step procedure: (1) training a span-based extractor to extract candidate terms, and (2) adjusting boundaries of candidate terms.

*Generative-based Models* : Despite the widespread use of Seq2Seq models for NLP tasks, the adoption of self-supervised pre-training approaches for ATE tasks has just recently gained traction (Lang et al., 2021). Given each sentence from the document $D = \{sent_1, ..., sent_n\}$ where $n$ is the number of sentences, the Seq2seq model (e.g., mBART) aims to transform each sentence into another sequence of elements $L = \{term_1, term_2, ..., term_M\}$ where $M$ is the number of candidate terms found in the input sentence.

*Prompt Engineering* : The emergence of LLMs has improved performance across several downstream NLP tasks with huge traction on prompt engineering with in-context learning (ICL) that leverages the LLM ability to generate texts with only a few task-specific examples as demonstrations (or few-shot demonstrations). Despite their attraction, none have been used to solve the term extraction task yet except our initial work (Tran et al., 2024a), leaving a gap in the terminology research.

## 3 Datasets

The experiments were conducted on version 1.5 of the ACTER[14] (Rigouts Terryn et al., 2020a) covering texts from diverse languages and domains. The ACTER dataset is a collection of 12 corpora covering four domains (*corruption* (corp), *equitation* (equi), *wind energy* (wind), and *heart failure* (htfl)) in three languages (English (en), French (fr) and Dutch (nl)). The dataset has two types of gold standard annotations: one containing both terms and named entities (NES); and the other one containing only terms (ANN). The dataset details have already been elaborately described in Rigouts Terryn et al. (2020b), and we refer interested readers to this work for further information.

[14] `https://github.com/AylaRT/ACTER`

## 4 Methodology

The general workflow of our proposed prompting term extractor or so-called *LlamATE* is visualized in Figure 2.

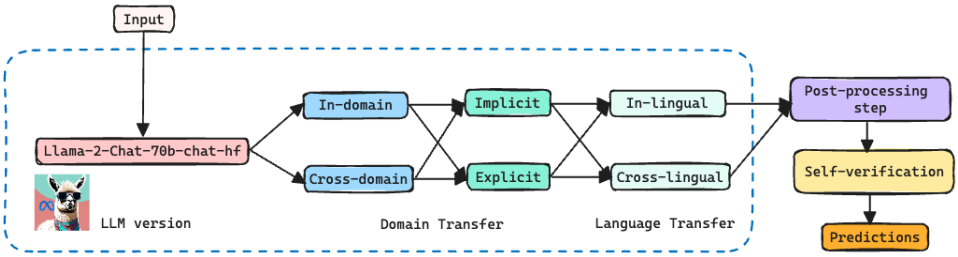


Fig. 2: Our general *LlamATE* workflow for few-shot prompting term extractor with three optional settings: [1] whether *LlamATE* shows in-domain or cross-domain demonstration, and whether *LlamATE* defines the domain of the demonstrations explicitly or implicitly (domain transfer); [2] whether *LlamATE* uses in-lingual or cross-lingual examples (language transfer); and [3] whether *LlamATE* applies self-verification with/without explanation or not (self-verification).

### 4.1 Large language models

The emergence of LLMs has improved performance across several downstream tasks with two different strategies: fine-tuning and ICL. While the fine-tuning strategy involves initializing a pre-trained model and conducting additional training epochs on task-specific supervised data, ICL leverages the LLM's ability to generate texts with only a few task-specific examples as demonstrations. On the one hand, training or fine-tuning LLMs specifically tailored to a particular task (e.g., term extraction) poses challenges due to the substantial computational resources required (Wang et al., 2023b). On the other hand, we believe that the vast world knowledge of LLMs offers promising insights for improving the handling of such extraction tasks.

As a result, in our research, we applied the ICL strategy to *Llama-2-Chat*, the chat version of the open-sourced auto-regressive LLM released by Meta AI, tuned to align with human preferences for helpfulness and safety. We used the largest version of *Llama-2-Chat*, namely *Llama-2-Chat-70b-chat-hf*[15] trained between January 2023 and July 2023 (see our series of complementary experiments in Section 7 to support this optimal choice).

[15] https://huggingface.co/meta-Llama/Llama-2-70b-chat-hf

### 4.2 Architecture Design

The quality of the input prompt when querying an LLM system is essential to the quality of the response. The popularization of LLMs was followed by the development of prompt engineering, the aim of which is to design prompts that best fit the language model to maximize the quality of the generated text. As a rule of thumb, the more explicit the prompt, the better[16]. Some examples of the task can be included to show the language model the sort of answers that are expected; this is known as a few-shot demonstration. Moreover, specifying certain criteria like the field or discipline that is relevant to the task narrows down the language model into a fixed text generation style. We have therefore designed our prompt as follows (see example in Figure 3).

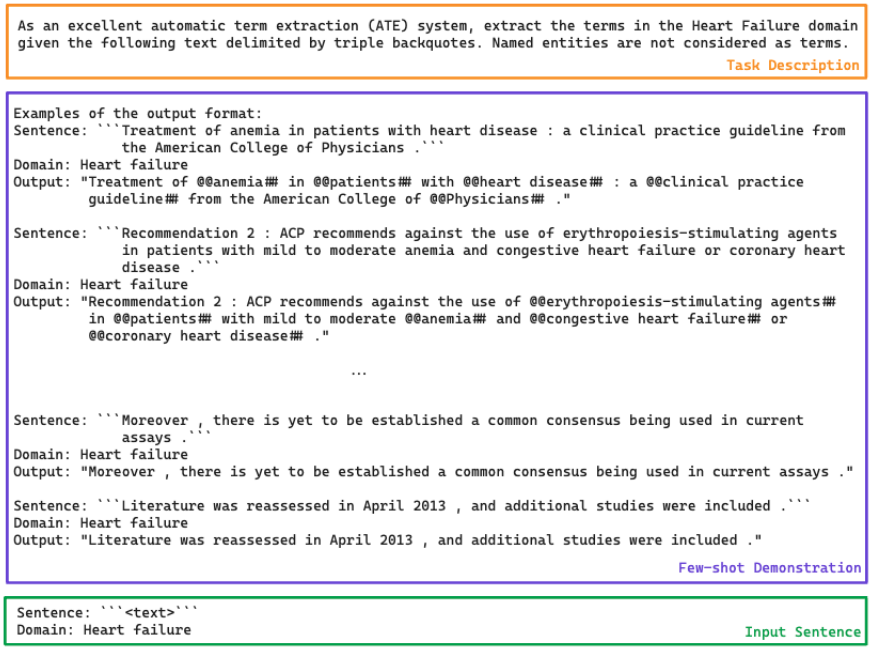


Fig. 3: An example of our prompt to extract the candidate terms for the ANN version of the ACTER dataset in the explicit in-domain in-lingual settings.

[16] https://platform.openai.com/docs/guides/prompt-engineering

*1. Prompt language*: By default, we use prompt instruction in English for all scenarios, even for term extraction in another language (e.g., French), as it is the most ubiquitous language in all their training corpora (i.e., 89.70% of training data in *Llama-2-Chat* is in English).

*2. Task Description*: We define the role that the LLM needs to follow and specify the task the LLM has to perform: *"As an excellent automatic term extraction (ATE) system, extract the terms in the Heart Failure domain given the following text delimited by triple backquotes."*. We provide an additional sentence to clarify the scope of the candidate terms. In the ANN version, we mention *"Named entities are not considered as terms."* and in the NES version, we mention *"Named entities are considered as terms."*.

*3. Few-shot Demonstration*: In the context of the ATE task, a standard few-shot or k-shot sample refers to obtaining exactly a few ($k$) examples where an example is a sentence in the corpora. However, meeting this strict requirement can be challenging, especially when dealing with data that exhibits an imbalanced distribution between sentences containing terms and not containing terms, in in-domain or cross-domain settings. In-domain means that we use examples from one domain (e.g., *heart failure*) for few-shot demonstrations and ask *LlamATE* to extract the candidate terms from another sentence in the same domain (e.g., *heart failure*). Cross-domain means that we use examples from one or more domains (e.g., *corruption*, *equitation*, *wind*) for demonstration and ask *LlamATE* to extract the candidate terms from another sentence in the new, unseen domain (e.g., *heart failure*). To address this challenge, we draw inspiration from the work of Ding et al. (2021) and introduce a relaxation to the criterion. We make an empirical study to find the optimal number of positive and negative examples to demonstrate LLMs as in Section 7.2 and use relaxation methods to permit a maximum of $1.2 \times k$ examples in cross-domain k-shot scenarios. We provide examples that are appended to the task description phase to regulate the format of outputs for each test input. The demonstration sequentially packs a list of examples, each consisting of the input (sentence and/or domain) and output sequences. The output sequences use the generative output format where we use unique tokens "@@" and "##" to encapsulate the candidate terms (see the example in Figure 3 and how we find the optimal output format in Section 7.2).

Note that for the demonstration, we opt to extract the candidate term, regardless of the nature of the term. Thus, we have not discriminated between the ACTER term annotations, where specific terms, general terms, out-of-domain (OOD) terms, and named entities are distinguished.

### 4.3 Domain Transfer

Domain greatly influences the selection of terms in a collection of texts. However, only keeping those terms specific to the domain can be restrictive and limit the vocabulary diversity, leading to certain drawbacks in further tasks like cross-domain alignment. The concept of *termhood* was introduced by Kageura and Umino (1996) to indicate the relation degree between a lexical unit and specific concepts inside a domain. Rigouts Terryn et al. (2021) address this concern

by defining *termhood* as a function of lexicon- and domain-specificity in their annotation guidelines. On the one hand, lexicon-specificity denotes if a lexical unit is only known by specialists or if it is part of a common language. On the other hand, domain-specificity denotes whether a lexical unit is relevant or unrelated to the researched domain.

We find two analogies between *termhood* and LLMs with ICL for the ATE tasks. First, domain-specificity can be induced into the prompt by explicitly declaring the domain of interest or letting the language model infer it from the few-shot demonstrations. Second, lexicon-specificity can be represented with few-shot demonstrations that illustrate the input sentences and the desired output regardless of the domain. These two analogies result in four different prompt scenarios as below:

- *Explicit in-domain*: In-domain demonstrations with explicit domain.
- *Implicit in-domain*: In-domain demonstrations with implicit domain.
- *Explicit cross-domain*: Cross-domain demonstrations with explicit domain.
- *Implicit cross-domain*: Cross-domain demonstrations with implicit domain.

With these scenarios, we verify the impact of the domain on the predictive performance of *LlamATE* when designing prompts with few demonstrations to help LLMs capture the candidate terms in the correct formats.

### 4.4 Language Transfer

We evaluate the capability of *LlamATE* to apply the knowledge learned from the over-represented language (i.e., English, which accounts for 89.70% of *Llama-2-Chat*'s training data) in the term extraction task to another unseen under-represented language in the *Llama* model (i.e., French and Dutch). Therefore, for both French and Dutch, we use the English prompts where the demonstrations are the examples extracted from the English corpus and combine this setting with the above-mentioned domain transfers for both ANN and NES versions. In this scenario, we examine how well *LlamATE* performs without the language-specific examples and how good the knowledge transfer between different languages is.

### 4.5 Postprocessing Steps

More than 95% of the predictions returned by *LlamATE* correspond to the original sentences with the candidate terms encapsulated by the symbols "@@" and "##". However, despite not asking for an explanation or any further information, there exists the case where *LlamATE* returns the encapsulated sentence and (1) further explanation; (2) a summary of the candidate list in the form of a bulleted list with or without encapsulated symbols, or (3) a small note for a conclusion. We thus add an automatic post-processing step that uses regex to normalize the predicted outputs with the following steps. First, we skip the unasked details provided by *LlamATE* to avoid potential noise (as the additional information may cover hallucinated candidate terms) and keep only the answered sentence. We then filter the candidate terms using regex and formulate them as a list of candidate terms for each sentence.

### 4.6 Self-verification

LLMs suffer from *hallucination* or overprediction issues, even with demonstrations. To address this issue, we have developed a self-verification strategy where the model re-evaluates or checks its own response against the original prompt or the extracted information. The primary purpose is to ensure that the response with the extracted candidate terms provided is accurate and relevant to the prompt. Given the candidate terms extracted by *LlamATE*, we ask the LLMs to further verify whether the extracted term is correct by responding with *YES* or *NO* with and without *Explanation*.

*YES/NO Verification* : We ask the model to self-verify if the list of extracted candidate terms it returns is correct given the sentence and each term in the list. In this way, the step helps the model to identify and correct potential mistakes or inconsistencies in its initial output and filter out the false positives, improving the performance and robustness. Sharing the same mechanism as the main prompt, the format of our self-verification prompt is exemplified in Figure 4.

```
As an excellent automatic term extraction (ATE) system, verify whether a given phrase is a term in
the Heart Failure domain in the given sentence by answering YES or NO.
Named entities are not considered as terms.                                          Task Description

Examples of the output format:
Sentence: ```Treatment of anemia in patients with heart disease : a clinical practice guideline from
            the American College of Physicians .```
Phrase: anemia
Output: YES

Sentence: ```Recommendation 2 : ACP recommends against the use of erythropoiesis-stimulating agents
            in patients with mild to moderate anemia and congestive heart failure or coronary heart
            disease .```
Phrase: erythropoiesis-stimulating agents
Output: YES

                                          ...


Sentence: ```Treatment of anemia in patients with heart disease : a clinical practice guideline from
            the American College of Physicians .```
Phrase: Treatment
Output: NO

Sentence: ```Recommendation 2 : ACP recommends against the use of erythropoiesis-stimulating agents
            in patients with mild to moderate anemia and congestive heart failure or coronary heart
            disease .```
Phrase: Recommendation
Output: NO

                                                                              Few-shot Demonstration

Sentence: ```<text>```
Phrase: <candidate terms>                                                        Input Sentence
```

Fig. 4: An example of the YES/NO verification for ACTER's ANN version.

*YES/NO Verification with Explanation* : Similar to *YES/NO verification*, we ask the model to verify the list of extracted candidate terms but elaborate on

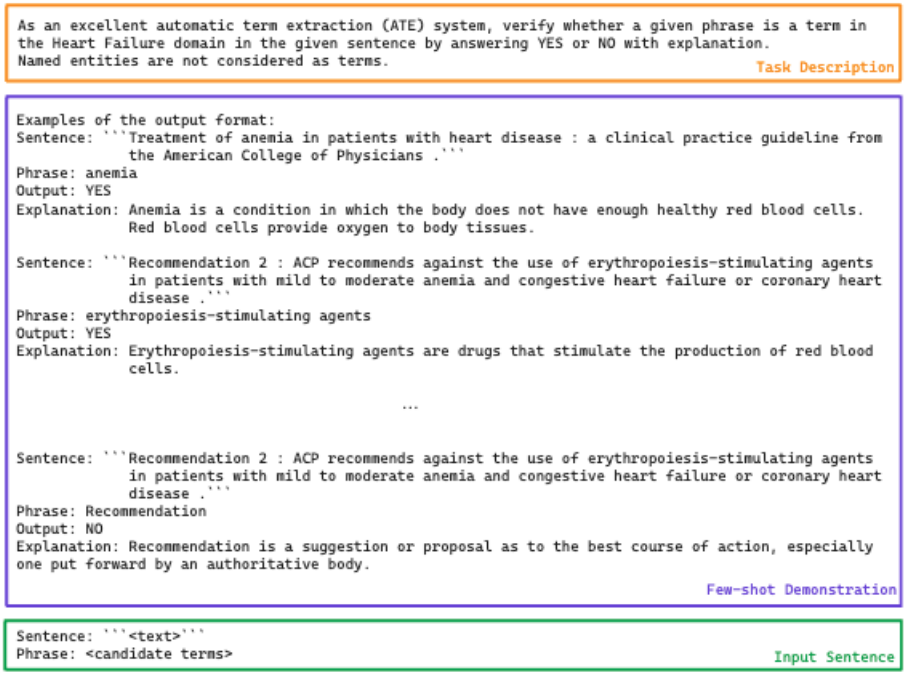


Fig. 5: An example of the YES/NO verification with explanation for ACTER's ANN version.

its answer by asking to provide useful evidence for humans to verify the answer. We have an assumption that the additional explanation will make the model's decision-making process becomes more transparent, which can be valuable in understanding how the output is derived. The justification from the model can also increase our trust in the model's output. This self-verification with explanation prompt is depicted in Figure 5.

In both cases, we pack multiple demonstrations with the same number of positive and negative examples in the verification prompt. Demonstrations are followed by the test example, and fed to the LLMs to obtain the output.

### 4.7 Experiment Settings

We use the largest version (70b) of *Llama-2-Chat*, developed by MetaAI, the most powerful model of the *Llama* sub-series as of mid-April 2024 (see the comparison in Table 2) via the HuggingFace's API endpoint. In our experiments, the *Llama-2-Chat* is used only as a predictor, we only query the model, and no additional fine-tuning steps have been realized. Based on the sentence length in the corpora, we set the output length of the model to 1,024 tokens. With regard to experimental reproducibility, we set the temperature parameter to 0.1 (close to zero, but not zero). This parameter must be a strictly positive floating point

number because if the temperature is exactly zero, this would effectively mean dividing by zero or multiplying by infinity, which would lead to invalid probabilities. This makes the sampling of the model almost deterministic and selects the most likely token with very high probability. All the experiments were requested from the HuggingFace's API endpoint and set up on CPUs in a Macbook Pro Ventura 13.6.5, 2.9 Quad-Score Intel i7, 16 GB memory.

### 4.8 Evaluation Metrics

We assess the performance of *LlamATE* using the following metrics:

- *Evaluation metrics*: The performance of each term extractor at the level of terms is measured by strictly comparing the aggregated list of candidate terms identified across the entire test set against the manually designated gold standard list of terms, using precision (P), recall (R), and F1-score (F1). These evaluation metrics were the same as for the experiments from the *TermEval 2020* competition (Rigouts Terryn et al., 2020a) and other related works (Tran et al., 2024a,b).
- *Environmental metrics*: We use the version 2.2. of the Green Algorithms[17] to estimate the carbon footprint of each experiment, based on factors such as runtime, computing hardware, and location where electricity used by our computer facility was produced.

## 5 Results

Table 1 presents the results of our main experiments which are divided into eight categories: [1] baselines; [2] k-shot baseline; [3] monolingual domain transfer; [4] cross-lingual transfer; [5] monolingual domain transfer with self-verification; [6] cross-lingual transfer with self-verification; [7] monolingual domain transfer with self-verification and explanation; and [8] cross-lingual transfer with self-verification and explanation. For a fair comparison, we only compare *LlamATE* with the benchmarks covering the performance of both versions in ACTER datasets. They include the winners of *TermEval 2020* shared task (*TALN-LS2N* (Hazem et al., 2020) for English and French, and *NLPLab UQAM* (Le and Sadat, 2021) for Dutch), the best ML classifier (HAMLET (Rigouts Terryn et al., 2021)), and the best token classifier for all three languages in ATE tasks above all popular transformer-based models, namely *XLMR* (Tran et al., 2022a, 2024b) with two different annotation regimes (Tran et al., 2024b). We also compare our prompting with the token classifier where we only fed the same number of demonstrations we used in the *LlamATE* as training examples (*10-shot XLMR*) to verify the impact of prompting given the lack of annotated data.

### 5.1 General Observation

The results showed that *LlamATE* significantly outperformed the benchmark token classifier (e.g., *10-shot XLMR*) when the number of training examples was

[17] http://calculator.green-algorithms.org/

| Settings | ANN version | | | | | | | | | NES version | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | English | | | French | | | Dutch | | | English | | | French | | | Dutch | | |
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| *(Group 1) Baselines* | | | | | | | | | | | | | | | | | | |
| *TALN-LS2N* | 32.6 | *72.7* | 45.0 | 41.9 | 50.9 | 45.9 | - | - | - | 34.8 | *70.9* | 46.7 | 45.2 | 51.6 | 48.2 | - | - | - |
| *NLPLab UQAM* | 20.1 | 16.0 | 17.8 | 15.1 | 11.2 | 12.9 | 18.1 | 19.3 | 18.6 | 21.5 | 15.6 | 18.1 | 16.1 | 11.2 | 13.2 | 18.9 | 19.3 | 18.6 |
| *HAMLET* | 45.6 | 66.9 | 54.2 | 50.7 | *74.3* | 60.2 | 55.5 | *81.8* | 66.1 | 47.1 | 67.3 | 55.4 | 51.4 | *74.3* | 60.8 | 55.8 | *80.9* | 66.0 |
| *XLMR$_{BIO}$* | *55.6* | 56.4 | 56.0 | *64.9* | 58.2 | 61.4 | *69.2* | 69.0 | 69.1 | *58.4* | 61.1 | *59.7* | *67.4* | 57.5 | 62.1 | *69.9* | 67.7 | 69.8 |
| *XLMR$_{NOBI}$* | 51.1 | 67.2 | *58.0* | 63.2 | 60.5 | *61.8* | 64.5 | 78.1 | *70.6* | 53.1 | 67.3 | 59.3 | 63.1 | 62.7 | *62.9* | 69.2 | 73.2 | *71.1* |
| *(Group 2) k-shot baselines* | | | | | | | | | | | | | | | | | | |
| *10-shot XLMR* | 70.8 | 2.7 | 5.2 | 66.7 | 0.1 | 0.2 | 87.1 | 1.3 | 2.6 | 64.4 | 10.0 | 17.3 | 81.8 | 0.4 | 0.8 | 81.3 | 12.5 | 21.7 |
| *(Group 3) Monolingual Domain Transfer* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | 51.7 | **50.6** | 51.1 | 41.1 | 34.7 | 37.6 | 52.5 | 56.9 | 54.6 | 53.6 | **59.0** | 56.2 | 41.4 | 37.2 | 39.2 | 49.0 | 49.2 | 49.1 |
| *Implicit in-domain* | 48.4 | 49.4 | 48.9 | 38.2 | 32.7 | 35.2 | 50.4 | **58.0** | 53.9 | 50.3 | 57.4 | 53.6 | 38.6 | 31.3 | 34.6 | 50.5 | 54.1 | 52.2 |
| *Explicit cross-domain* | 51.5 | 45.6 | 48.4 | 58.1 | 35.5 | 44.1 | 54.4 | 37.4 | 44.3 | 52.6 | 54.4 | 53.5 | 52.2 | 33.8 | 41.0 | 35.4 | 49.1 | 41.1 |
| *Implicit cross-domain* | 49.6 | 44.3 | 46.8 | 50.3 | 32.0 | 39.1 | 51.9 | 43.7 | 47.4 | 50.4 | 49.5 | 49.9 | 45.3 | 37.2 | 40.9 | 36.1 | 50.2 | 42.0 |
| *(Group 4) Cross-lingual Transfer* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | - | - | - | 48.2 | 44.8 | 46.4 | 47.0 | 52.8 | 49.7 | - | - | - | 46.1 | 50.9 | 48.4 | 47.4 | 58.2 | 52.2 |
| *Implicit in-domain* | - | - | - | 44.9 | **48.4** | 46.6 | 45.8 | 56.8 | 50.7 | - | - | - | 43.3 | **54.9** | 48.4 | 43.4 | **63.5** | 51.6 |
| *Explicit cross-domain* | - | - | - | 47.4 | 42.3 | 44.7 | 47.8 | 48.8 | 48.3 | - | - | - | 49.0 | 44.6 | 46.7 | 48.4 | 51.3 | 49.8 |
| *Implicit cross-domain* | - | - | - | 45.8 | 45.7 | 45.7 | 46.6 | 52.9 | 49.6 | - | - | - | 45.1 | 47.6 | 46.3 | 46.2 | 54.1 | 49.8 |
| *(Group 5) Monolingual Domain Transfer with Self-verification* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | 54.0 | 49.2 | **51.5** | 47.3 | 34.1 | 39.6 | 55.2 | 54.1 | 54.6 | **55.5** | 57.6 | **56.5** | 46.1 | 36.3 | 40.6 | 53.7 | 47.1 | 50.2 |
| *Implicit in-domain* | 51.9 | 47.4 | 49.5 | 45.7 | 32.1 | 37.7 | 54.2 | 55.6 | **54.9** | 53.7 | 55.3 | 54.5 | 45.2 | 30.7 | 36.6 | **55.4** | 52.0 | 53.6 |
| *Explicit cross-domain* | 53.1 | 44.6 | 48.5 | **60.8** | 35.0 | 44.4 | **57.5** | 35.7 | 44.1 | 54.2 | 52.7 | 53.4 | **56.4** | 33.2 | 41.8 | 47.0 | 47.2 | 47.1 |
| *Implicit cross-domain* | 52.0 | 42.7 | 46.9 | 54.2 | 31.3 | 39.7 | 56.0 | 41.5 | 47.7 | 54.2 | 48.1 | 51.0 | 49.7 | 36.4 | 42.0 | 48.0 | 47.9 | 47.9 |
| *(Group 6) Cross-lingual Transfer with Self-verification* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | - | - | - | 50.9 | 44.0 | 47.2 | 50.3 | 51.2 | 50.7 | - | - | - | 48.1 | 49.8 | 48.9 | 51.6 | 56.5 | 53.9 |
| *Implicit in-domain* | - | - | - | 48.3 | 47.7 | **48.0** | 50.3 | 54.6 | 52.4 | - | - | - | 45.7 | 53.5 | **49.3** | 48.7 | 61.4 | **54.3** |
| *Explicit cross-domain* | - | - | - | 50.3 | 41.7 | 45.6 | 51.3 | 46.7 | 48.9 | - | - | - | 51.0 | 43.4 | 46.9 | 52.6 | 49.6 | 51.1 |
| *Implicit cross-domain* | - | - | - | 48.9 | 44.9 | 46.8 | 51.4 | 50.8 | 51.1 | - | - | - | 48.3 | 46.8 | 47.5 | 51.4 | 52.2 | 51.8 |
| *(Group 7) Monolingual Domain Transfer with Self-verification and Explanation* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | **54.4** | 47.6 | 50.8 | 47.1 | 34.3 | 39.7 | 54.6 | 54.2 | 54.4 | 55.3 | 52.2 | 53.7 | 45.2 | 36.2 | 40.2 | 52.5 | 47.4 | 49.8 |
| *Implicit in-domain* | 51.8 | 45.6 | 48.5 | 45.0 | 32.2 | 37.5 | 52.9 | 55.6 | 54.2 | 53.6 | 50.0 | 51.7 | 43.7 | 30.5 | 35.9 | 53.7 | 51.9 | 52.8 |
| *Explicit cross-domain* | 53.2 | 42.9 | 47.5 | **60.8** | 34.9 | 44.3 | 56.6 | 35.9 | 43.9 | 54.1 | 47.7 | 50.7 | 55.3 | 33.1 | 41.4 | 43.2 | 47.3 | 45.2 |
| *Implicit cross-domain* | 52.1 | 41.6 | 46.3 | 53.8 | 31.5 | 39.7 | 54.6 | 41.9 | 47.4 | 54.2 | 44.2 | 48.7 | 48.7 | 36.6 | 41.8 | 44.4 | 48.2 | 46.2 |
| *(Group 8) Cross-lingual Transfer without Self-verification and Explanation* | | | | | | | | | | | | | | | | | | |
| *Explicit in-domain* | - | - | - | 51.0 | 44.0 | 47.2 | 49.2 | 50.9 | 50.0 | - | - | - | 47.7 | 49.2 | 48.4 | 49.7 | 56.1 | 52.7 |
| *Implicit in-domain* | - | - | - | 48.1 | 47.7 | 47.9 | 48.9 | 54.7 | 51.6 | - | - | - | 45.4 | 52.9 | 48.9 | 46.1 | 60.9 | 52.5 |
| *Explicit cross-domain* | - | - | - | 50.2 | 41.7 | 45.6 | 50.2 | 46.8 | 48.4 | - | - | - | 50.4 | 42.5 | 46.1 | 50.8 | 49.4 | 50.1 |
| *Implicit cross-domain* | - | - | - | 48.9 | 45.0 | 46.9 | 50.1 | 51.0 | 50.5 | - | - | - | 47.8 | 46.5 | 47.1 | 49.3 | 52.0 | 50.6 |

Table 1: The evaluation of *LlamATE* with different settings. The best results for models using full training data are underlined while the best results of *LlamATE* regarding precision, recall, and F1-score, respectively are in bold. Grey lines indicate the best performance based on F1-score.

identical to the number of demonstrations we fed into the instructional prompts for all prompt designs (see Groups 3 against Group 2). At the same time, our classifier showed competitive performance compared to the fully supervised token classifier benchmark (see Groups 3 against Group 1).

### 5.2 Verification Strategies Comparison

We compared the performance of naïve instructional prompts without using self-verification (Groups 3 and 4), with self-verification (Groups 5 and 6), and with verification and explanation (Groups 7 and 8). The results indicated that self-verification without explanation improved prompting performance. On the one hand, self-verification helped *LlamATE* recognize and correct its own errors (Groups 3 and 4). On the other hand, by checking whether the extracted terms matched the sentence and each other, the model was able to detect inconsistencies that it might otherwise have missed. In addition, the verification process allowed the model to assess its confidence in the answer and prioritize the correct core answer before focusing on the explanation.

However, the addition of an explanation in the self-verification step (Groups 7 and 8) was helpful for humans who needed to understand the reasoning behind the model's response. This transparency allowed for a better evaluation of the model's thought process and could reveal potential biases or limitations. Although this setting performed better than the one without self-verification, it could not outperform self-verification due to the possibility of a misleading explanation. *LlamATE* struggled to provide clear and accurate explanations, resulting in users being misled despite an explanation.

### 5.3 Monolingual vs. Cross-lingual Transfer Comparison

We compared the performance of the monolingual domain transfer (Group 3) against the cross-lingual transfer (Group 4), the monolingual domain transfer with self-verification (Group 5) against the cross-lingual domain transfer with self-verification (Group 6), and the monolingual domain transfer with self-verification with explanation (Group 7) against cross-lingual domain transfer with self-verification (Group 8) given the prompts and examples were in the dominant language (English) and the test sets were French and Dutch, respectively. The cross-lingual settings achieved comparable results (if a self-verification step was not added or added with additional explanation) and even better than the monolingual ones (if a self-verification step was added), which confirmed hypothesis that *"In a cross-lingual setting, a token classifier achieves comparable results to monolingual training in a target language"* as proved in the work of Tran et al. (2024b). This efficient performance in the *heart failure* domain could be attributed to the fact that in the medical domain, many terms are of Latin or Greek origin. This could facilitate cross-lingual transfer as the sub-words can be very similar between languages (especially for closely related languages such as English, French, and Dutch). The best performance was found in implicit

in-domain cross-lingual transfer with self-verification design among all experimented settings.

### 5.4 Environmental Impact

In total, we performed 144 experiments for the main outcomes and 82 experiments for the ablation studies, each of which lasted 9 hours on average. We ran all the experiments in Metropolitan France. According to Green Algorithms, we estimated that the experiments with *Llama-2-Chat* generated about 7,381 kg of CO2 equivalent in both versions of the ACTER dataset (4,703 kg for the main experiments, 2,678 kg for the ablation).

## 6 Discussion

We focus on understanding the predictive performance of our prompting classifiers through comprehensive error analysis on the impact of term length and the practical use of LLMs for lesser-represented languages in term extraction.

### 6.1 The Impact of Term Length

Figure 6 visualizes the distribution in term length $k = [1, 2, 3, 4, \leq 5]$ of the correct and incorrect prediction of the best settings of *LlamATE* for each data version and language. This includes the monolingual domain-transfer explicit in-domain setting with self-verification for both versions of English, the monolingual domain-transfer implicit in-domain setting with self-verification for the ANN version of Dutch from the monolingual domain transfer, and the cross-lingual-transfer implicit in-domain setting from cross-lingual transfer with self-verification for the rest (both versions of French and NES version of Dutch). As can be seen, in English, the proportion of terms correctly predicted by *LlamATE* was higher than the proportion of incorrectly predicted terms when their term length was less than or equal to four words. In the case of French, this ratio was maintained even for terms with five words. In Dutch, on the other hand, the proportion of correctly predicted terms was only higher for terms with one or two words.

### 6.2 Practical Use of LLMs for Lesser-represented Languages

Our experiments suggested that the performance of *LlamATE* surpassed one of the language models sharing the same number of training examples, but they still fell short of models trained on dedicated annotated data (fully supervised models). However, the performance can be judged satisfactory enough for pre-annotation use, to complement or accelerate manual annotation. For example, *LlamATE* could be used as a first pass to identify potential candidate terms. This pre-populated list significantly reduces the manual effort required for human annotators, who then focus on verifying and refining the suggestions. By suggesting

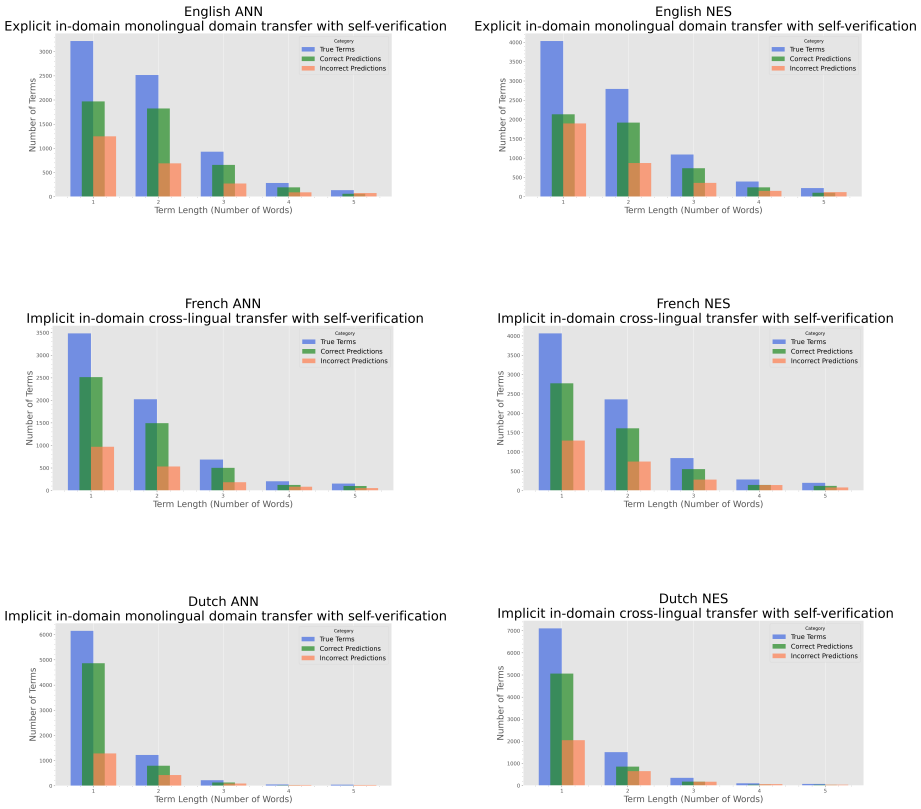


Fig. 6: Distribution of Correct and Incorrect Prediction Regarding Term Lengths.

candidate terms, *LlamATE* accelerates the overall annotation process, leading to faster completion of tasks where the identification of relevant terms is crucial. Moreover, these pre-populated lists can be used in "active learning" where *LlamATE* suggests terms, a human validates or refines them, and *LlamATE* uses this feedback to improve its suggestions on subsequent tasks. This iterative process leads to a continuously improving model even with limited annotated data. Thus, this is still a promising tool for finding the candidate terms, especially when working with limited data of the same language and/or domain or when no annotated data is available for a given language and/or domain. While it may not be as good as models trained on dedicated annotated data (fully supervised ones), it can significantly speed up the process by suggesting term candidates that are later reviewed and refined by human experts.

### 6.3 Limitations

*Random Demonstrations* : The positive and negative demonstrations in the main prompts are shuffled randomly and the negative examples in the self-verification prompts are selected randomly from the sentence that existed as demonstrations in the main prompts. This randomness can lead to noise in performance measurements. Replicating all the combinations of positive and negative examples would allow us to draw more solid conclusions, but would also come at a considerable environmental cost.

*API Dependency* : Our experiments with *Llama-2-Chat-70b-chat-hf* were performed via the HuggingFace's Inference API[18]. The use of APIs to prompt LLMs helps overcome infrastructure limitations when these models contain several tens of billions of parameters. However, there are some limitations to take into account, mostly concerning our limited control over the LLM, data privacy, and latency in performance. LLM APIs may collect data about our interactions with the API, which could raise data privacy concerns. At the same time, it may have higher latency for the free version, especially when the number of requests is overloaded (*"Rate limit reached. You reached free usage limit (reset hourly)."*). This can be a problem if we need to build a term extractor as an application that retrieves huge requests and requires real-time responses.

*Data Contamination/Leakage* : A potential problem with using *LlamATE* to extract candidate terms from corpora such as ACTER is the risk of data contamination or leakage. Since the dataset is publicly available on GitHub in a widely used format, there is a non-negligible chance that the LLMs may have encountered similar examples during pre-training or fine-tuning. This could lead to the model generating biased or overly familiar responses, which could affect the accuracy and novelty of the extracted terms. To mitigate this risk, it is important to carefully evaluate the model's performance on unseen data and consider techniques such as data augmentation or adversarial training to improve its robustness and generalization ability.

[18] `https://huggingface.co/docs/api-inference/index`

*The Diversity of Languages* : We have focused on three Indo-European languages, namely English, French, and Dutch. Note that these languages all belong to the Indo-European language family and use the same Latin alphabet in their writing system, albeit with different branches. English comes from the Germanic branch (closer to Dutch) and has a large vocabulary due to historical influences (French, Latin, Germanic). Dutch has some vocabulary in common with English but has a Germanic sound. French comes from the Italic/Romance branch, with many words originating from Latin, which also has the greatest similarities with the Germanic branch. It is therefore not known how *LlamATE* can be generalized to typologically different languages from different family branches.

## 7 Ablations

To better understand the contribution of each step, we carried out a series of complementary experiments, including the verification of [1] the sizes of *Llama-2-Chat*, [2] the optimal output format, [3] the optimal number of demonstrations, and [4] interactive or non-interactive demonstrations. We captured the optimal choice for each step as highlighted in bold arrows in Figure 7.

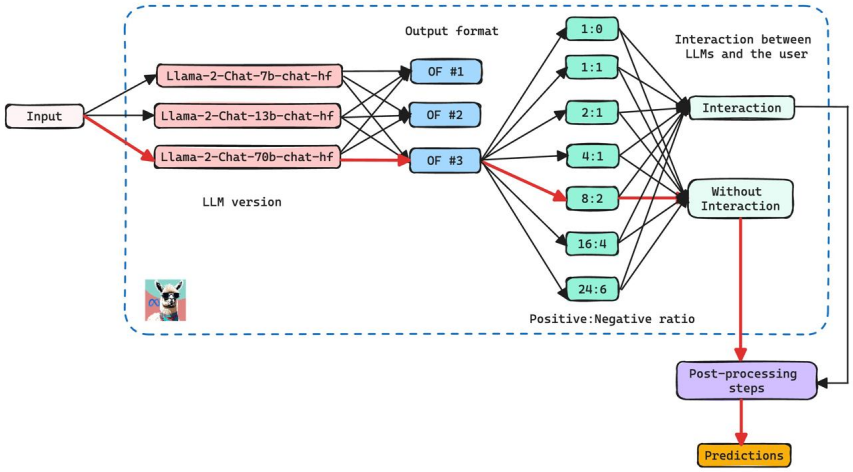


Fig. 7: Our 4-step procedure to choose the optimal configuration in the in-domain few-shot prompting workflow. The bold red arrow demonstrated the optimal path.

### 7.1 Model Sizes and Prompt's Output Designs

*Model Sizes* : We evaluate the performance of different versions of *Llama-2-Chat*, including *Llama-2-Chat-7b-chat-hf*, *Llama-2-Chat-13b-chat*, and *Llama-2-Chat-70b-chat-hf*. These models share the same training size (4k tokens), content

length (2T tokens), and percentage (%) of data the models contain in English (89.7%), French (0.16%), and Dutch (0.12%), respectively for pre-training. Table 2 demonstrates the difference in characteristics of *Llama-2-Chat* regarding the number of parameters used for pre-training, Grouped-Query Attention (GQA).

Table 2: Characterization of models used in our experiments.

| **Models** | *Llama-2-Chat-7b-chat-hf* | *Llama-2-Chat-13b-chat-hf* | *Llama-2-Chat-70b-chat-hf* |
|---|---|---|---|
| **# Params** | 7b | 13b | 70b |
| **GQA** | No | No | Yes |

*Output Formats* : We investigate three common outputs of language models to term extraction and adapt them to the output format (*OF*) of our prompts with in-domain few-shot demonstrations, as visualized in Figure 8. They include (1) *OF #1*: Sequence-labeling output where the output contains the information for each word label in the BIO regime; (2) *OF #2*: List of candidate terms output which is the same format as our original gold standard; and (3) *OF #3*: Generative output where we use unique tokens "@@" and "##" to surround the candidate terms.

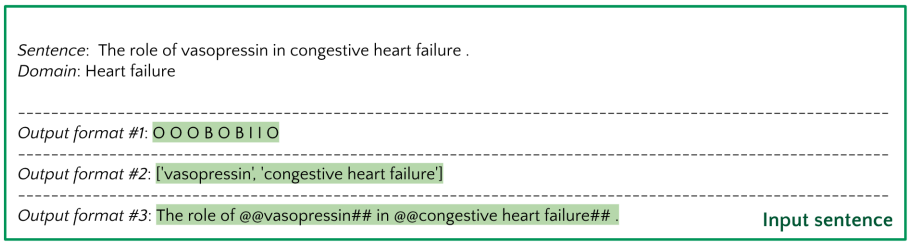


Fig. 8: An example of three output designs.

*Results* : We evaluated the performance of these output formats with different versions of *Llama-2-Chat* in the in-domain few-shot setting and reported the evaluation in Table 3 where we highlighted in grey the best performing format for each version. The results exhibited a considerable performance gap depending on the output format. *LlamATE* struggled with low precisions and recalls for sequence labeling (*OF #1*) compared to the other two formats. This suggested the gap between the semantic labeling task and the text generation one, which *Llama-2-Chat* has been trained for. Meanwhile, *OF #2* and *OF #3* formats showed up to 8 times higher scores than *OF #1* regardless of language and model size.

Besides, the LLMs showed variations given the language and model size. On the one hand, *Llama-2-Chat* with a smaller number of parameters (7*b* and 13*b*) proved to be a better fit for *OF #2*. On the other hand, although *OF #2* has a higher recall than *OF #1* and *OF #3* at all model sizes, the largest version (70*b*) outperformed the others with the highest values at total F1 in *OF #3*

| Settings | ANN versions | | | | | | | | | NES versions | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | English | | | French | | | Dutch | | | English | | | French | | | Dutch | | |
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| In-domain $LlamATE_{Llama-2-Chat-7b}$ | | | | | | | | | | | | | | | | | | |
| *OF #1* | 12.4 | 4.8 | 6.9 | 7.5 | 9.3 | 8.3 | 19.2 | 14.4 | 16.5 | 17.3 | 7.3 | 10.3 | 8.4 | 11.0 | 9.5 | 16.6 | 23.8 | 19.6 |
| *OF #2* | 40.4 | 62.6 | 49.1 | 36.3 | 59.2 | 45.0 | 40.4 | 73.1 | 52.0 | 42.9 | 63.4 | 51.2 | 36.0 | 61.6 | 45.4 | 40.3 | 75.6 | 52.6 |
| *OF #3* | 40.3 | 26.8 | 32.2 | 58.5 | 23.4 | 33.4 | 53.8 | 41.6 | 46.9 | 45.0 | 32.5 | 37.7 | 52.1 | 34.5 | 41.5 | 48.8 | 52.3 | 50.5 |
| In-domain $LlamATE_{Llama-2-Chat-13b}$ | | | | | | | | | | | | | | | | | | |
| *OF #1* | 12.1 | 1.7 | 3.0 | 11.2 | 6.6 | 8.3 | 25.6 | 5.9 | 9.6 | 25.9 | 2.4 | 4.4 | 8.4 | 5.3 | 6.5 | 23.5 | 5.9 | 9.4 |
| *OF #2* | 35.0 | 63.4 | 45.1 | 38.4 | 59.2 | 46.6 | 43.3 | 75.0 | 54.9 | 38.4 | 66.1 | 48.6 | 33.8 | 60.2 | 43.3 | 41.4 | 73.6 | 53.0 |
| *OF #3* | 40.0 | 36.9 | 38.4 | 41.0 | 48.7 | 44.5 | 46.1 | 56.2 | 50.7 | 40.3 | 47.5 | 43.6 | 35.7 | 49.4 | 41.4 | 47.9 | 49.1 | 48.5 |
| In-domain $LlamATE_{Llama-2-Chat-70b}$ | | | | | | | | | | | | | | | | | | |
| *OF #1* | 15.6 | 5.7 | 8.3 | 4.6 | 3.9 | 4.2 | 23.7 | 8.2 | 12.2 | 21.4 | 4.7 | 7.7 | 7.9 | 9.5 | 8.6 | 18.7 | 17.5 | 18.1 |
| *OF #2* | 36.8 | 65.9 | 47.2 | 38.0 | 64.8 | 47.9 | 42.3 | 74.8 | 54.0 | 39.9 | 67.2 | 50.1 | 33.2 | 61.8 | 43.2 | 41.1 | 74.8 | 53.1 |
| *OF #3* | 46.4 | 50.0 | 48.1 | 47.1 | 51.4 | 49.2 | 50.5 | 67.3 | 57.7 | 48.3 | 54.9 | 51.4 | 40.8 | 57.7 | 47.8 | 53.1 | 57.9 | 55.4 |

Table 3: Evaluation in performance of different output formats on the *heart failure* test set with in-domain demonstrations (OF = Output Format).

and solved to some extent the noise in the predictions introduced by the two previous formats. As the model only needed to mark the position of the terms and make copies for the rest, it was able to [1] reduce the difficulty in generating text that fully encodes label information (as in *OF #1*) of the input sequence, and [2] reduce the certain proportion of incorrect output formats generated (as in *OF #2*) due to its overgeneration.

Based on the best predictive performance and the cleanliness of the predictions to reduce the noise and the effort of postprocessing, we decided to use the *Llama-2-Chat-70b-chat-hf* version with *OF #3* and apply to both versions of all three languages.

### 7.2 Optimal Number of Demonstrations

To study the behavior of *LlamATE* in a few-shot context, we simulate the few-shot context by providing the models with only a few annotated examples with and without interaction on the English *heart failure* test set. We verify the optimal number of demonstrations and the proportion of positive and negative examples that should be provided to *LlamATE* so it maximizes the number of correct predictions. Furthermore, we evaluate whether *LlamATE* performs better if there exists an interaction between the LLM and the user during the prompting process.

*Number of Demonstrations* : Regarding few-shot demonstration, seven positive vs. negative amount of examples have been experimented with, including 1:0, 1:1, 2:1, 4:1, 8:2, 16:4, and 24:6. We start with a positive example to make sure the LLM understands how to formulate the prediction outputs, which zero-shot

demonstration failed to perform and increase the number of examples based on the ratio of positive and negative examples at the sentence level in the corpus.

*Interaction vs. No Interaction* : Regarding the interaction between LLM and the user, for each ratio, we experiment with two scenarios, where [1] we feed all the examples as demonstration at the same time (no interaction), and [2] we feed each one of the examples as demonstration consecutively as the interaction between the LLM and the user (interaction). Based on the results of Section 7.1, we applied the above settings to *Llama-2-Chat-70b-chat-hf* version using *OF #3* with in-domain examples.

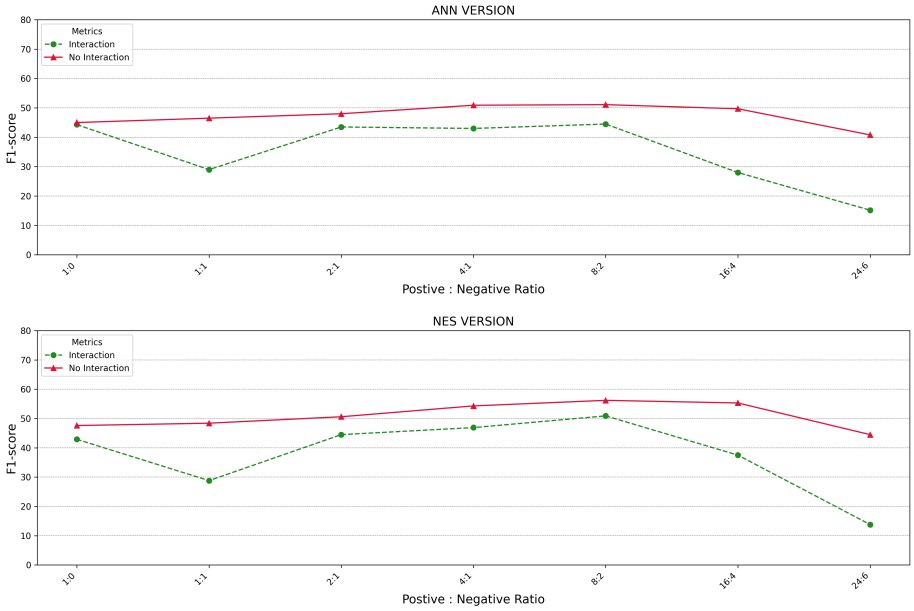


Fig. 9: Evaluation in F1-score of the different positive and negative number of demonstrations on English *heart failure* set.

*Results* : The performance plots are visualized in Figure 9. The results provided a consistent performance given the version of the English corpora (ANN and NES) with interaction and without interaction between LLMs and the user. Interestingly, using 8 positive and 2 negative examples for LLMs demonstration consistently achieved the best results regarding the F1-score. Furthermore, in both cases, prompting without interaction demonstrated a better and more consistent performance without noise in the prediction compared to the interactive versions. When feeding examples individually, *LlamATE* might get stuck on irrelevant details in a specific example, leading to inconsistent behavior when pro-

cessing future examples. Including all examples as demonstrations in the prompt provided a broader context, reducing the impact of individual noisy examples.

Therefore, we finalized our configuration for in-domain and cross-domain few-shot demonstration prompting as follows: *Llama-2-Chat-70b-chat-hf*, *OF #3*, few-shot demonstrations covering 8 positive, followed by 2 negative examples without interaction.

## 8 Conclusions

In summary, we investigated how well LLMs were able to extract domain terms and compared the performance in different languages and with different prompting strategies. We used a technique called in-context learning, where we prompted the LLMs with some sentences as examples covering terms and non-terms in the desired domain and language (we called this method *LlamATE*). Interestingly, *LlamATE* did not need to be explicitly told the domain of the examples in the prompt (explicit vs. implicit), both for examples coming from the same domain and for cross-domain examples. We tested our approach on the English, French, and Dutch datasets of the ACTER corpora. The results showed that *LlamATE* learned best from a few examples from the same domain, even without explicitly naming the domain, and transferred knowledge from languages that were well covered in LLMs (e.g., English) to less-represented languages in LLMs (e.g., French, Dutch). To improve performance, we also included a step in which *LlamATE* double-checks its answers (self-verification with and without explanation). Overall, this technique (*LlamATE* with instruction prompt and self-verification) offers a promising approach for less-represented term extraction tasks. While they were not a replacement for fully supervised models, they could increase efficiency and accuracy by streamlining the process of pre-annotation and speeding up manual annotation effort.

In the future, we would like to evaluate the performance of our LLMs prompting strategy for within-domain and general term labels from the ACTER dataset (or investigate how these categories are defined). Since the corpus used to train *Llama-2-Chat* contains a lot of medical data, it would be interesting to compare the results with another specialized domain where possibly less training data was used to train the foundation model. Thus, we would like to investigate the difference in performance when using few-shot approaches for more specialized domains (e.g., *heart failure*) than for more general domains (e.g., *corruption*) and for typologically different languages from different family branches than the Indo-European ones.

## Acknowledgments

The work was partially supported by the Slovenian Research and Innovation Agency (ARIS) core research program Knowledge Technologies (P2-0103) and

project KOBOS (J6-3131). The first author was partly funded by Region Nouvelle Aquitaine. This work has also been supported by the TERMITRAD (2020-2019-8510010) project funded by the Nouvelle-Aquitaine Region, France. The work was also supported by the project Cross-lingual and Cross-domain Methods for Terminology Extraction and Alignment, a bilateral project funded by the program PROTEUS under the grant number BI-FR/23-24-PROTEUS006.

# Bibliography


UTKA Andrius. Automatic extraction of lithuanian cybersecurity terms using deep learning approaches. In *Human Language Technologies–The Baltic Perspective: Proceedings of the Ninth International Conference Baltic HLT 2020*, volume 328, page 39. IOS Press, 2020.

Nikita A Astrakhantsev, Denis G Fedorenko, and D Yu Turdakov. Methods for automatic term recognition in domain-specific text collections: A survey. *Programming and Computer Software*, 41(6):336–349, 2015.

Jérôme Azé, Mathieu Roche, Yves Kodratoff, and Michele Sebag. Preference learning in terminology extraction: A roc-based approach. *arXiv preprint cs/0512050*, 2005.

Matthias Bay, Daniel Bruneß, Miriam Herold, Christian Schulze, Michael Guckert, and Mirjam Minor. Term extraction from medical documents using word embeddings. In *2020 6th IEEE CiSt*, pages 328–333. IEEE, 2021.

Chris Biemann and Alexander Mehler. *Text mining: From ontology learning to automated text processing applications*. Springer, 2014.

Elena Bolshakova, Natalia Loukachevitch, and Michael Nokel. Topic models can improve domain term extraction. In *European Conference on Information Retrieval*, pages 684–687. Springer, 2013.

M Teresa Cabré Castellví, Rosa Estopa Bagot, and Jordi Vivaldi Palatresi. Automatic term detection: A review of current systems. *Recent advances in computational terminology*, 2:53–88, 2001.

Merley Conrado, Thiago Pardo, and Solange Rezende. A machine learning approach to automatic term extraction using a rich feature set. In *Proceedings of the 2013 NAACL HLT Student Research Workshop*, pages 16–23, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL `https://aclanthology.org/N13-2003`.

Merley da Silva Conrado, Ariani Di Felippo, Thiago Alexandre Salgueiro Pardo, and Solange Oliveira Rezende. A survey of automatic term extraction for brazilian portuguese. *Journal of the Brazilian Computer Society*, 20(1):1–28, 2014.

Béatrice Daille, Éric Gaussier, and Jean-Marc Langé. Towards Automatic Extraction of Monolingual and Bilingual Terminology. In *COLING 1994 Volume 1: The 15th International Conference on Computational Linguistics*, 1994.

Julien Delaunay, Hanh Thi Hong Tran, Carlos-Emiliano González-Gallardo, Georgeta Bordea, Mathilde Ducos, Nicolas Sidere, Antoine Doucet, Senja Pollak, and Olivier De Viron. Coastterm: A corpus for multidisciplinary term extraction in coastal scientific literature. In *International Conference on Text, Speech, and Dialogue*, pages 97–109. Springer, 2024.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36, 2024.

Ning Ding, Guangwei Xu, Yulin Chen, Xiaobin Wang, Xu Han, Pengjun Xie, Hai-Tao Zheng, and Zhiyuan Liu. Few-nerd: A few-shot named entity recognition dataset. *arXiv preprint arXiv:2105.07464*, 2021.

Patrick Drouin. Term extraction using non-technical corpora as a point of leverage. *Terminology*, 9(1):99–115, 2003.

Ahmed El-Kishky, Yanglei Song, Chi Wang, Clare R. Voss, and Jiawei Han. Scalable topical phrase mining from text corpora. *Proc. VLDB Endow.*, 8(3):305–316, nov 2014. ISSN 2150-8097. https://doi.org/10.14778/2735508.2735519. URL `https://doi.org/10.14778/2735508.2735519`.

Denis Fedorenko, N Astrakhantsev, and D Turdakov. Automatic recognition of domain-specific terms: an experimental evaluation. *Proceedings of the Institute for System Programming*, 26(4):55–72, 2014.

Jody Foo and Magnus Merkel. Using machine learning to perform automatic term recognition. In *LREC 2010 Workshop on Methods for automatic acquisition of Language Resources and their evaluation methods, 23 May 2010, Valletta, Malta*, pages 49–54. European Language Resources Association, 2010.

Katerina T Frantzi, Sophia Ananiadou, and Junichi Tsujii. The c-value/nc-value method of automatic recognition for multi-word terms. In *International conference on theory and practice of digital libraries*, pages 585–604. Springer, 1998.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. How close is chatgpt to human experts? comparison corpus, evaluation, and detection, 2023.

Xiaowei Han, Lizhen Xu, and Feng Qiao. Cnn-bilstm-crf model for term extraction in chinese corpus. In *International Conference on Web Information Systems and Applications*, pages 267–274. Springer, 2018.

Amir Hazem, Mérieme Bouhandi, Florian Boudin, and Béatrice Daille. TermEval 2020: TALN-LS2N System for Automatic Term Extraction. In *Proceedings of the 6th International Workshop on Computational Terminology*, pages 95–100, 2020.

Amir Hazem, Mérieme Bouhandi, Florian Boudin, and Béatrice Daille. Cross-lingual and cross-domain transfer learning for automatic term extraction from low resource data. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 648–662, 2022.

ISO ISO:1087. Terminology work and terminology science–vocabulary. *ISO1087*, 2019.

Alex Judea, Hinrich Schütze, and Sören Brügmann. Unsupervised training set generation for automatic acquisition of technical terminology in patents. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical Papers*, pages 290–300, 2014.

Kyo Kageura and Bin Umino. Methods of Automatic Term Recognition.A Review. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 3(2):259–289, 1996.

Mladen Karan, Jan Šnajder, and Bojana Dalbelo Bašić. Evaluation of classification algorithms and features for collocation extraction in croatian. In

*Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 657–662, 2012.

Jan Kocoń, Igor Cichecki, Oliwier Kaszyca, Mateusz Kochanek, Dominika Szydło, Joanna Baran, Julita Bielaniewicz, Marcin Gruza, Arkadiusz Janz, Kamil Kanclerz, Anna Kocoń, Bartłomiej Koptyra, Wiktoria Mieleszczenko-Kowszewicz, Piotr Miłkowski, Marcin Oleksy, Maciej Piasecki, Łukasz Radliński, Konrad Wojtasik, Stanisław Woźniak, and Przemysław Kazienko. Chatgpt: Jack of all trades, master of none, 2023.

Maren Kucza, Jan Niehues, Thomas Zenkel, Alex Waibel, and Sebastian Stüker. Term Extraction via Neural Sequence Labeling a Comparative Evaluation of Strategies Using Recurrent Neural Networks. In *INTERSPEECH*, pages 2072–2076, 2018.

Christian Lang, Lennart Wachowiak, Barbara Heinisch, and Dagmar Gromann. Transforming term extraction: Transformer-based approaches to multilingual term extraction across domains. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3607–3620, 2021.

Ngoc Tan Le and Fatiha Sadat. Multilingual automatic term extraction in low-resource domains. In *The International FLAIRS Conference Proceedings*, volume 34, 2021.

Annaïch Le Serrec, Marie-Claude L'Homme, Patrick Drouin, and Olivier Kraif. Automating the compilation of specialized dictionaries: Use and analysis of term extraction and lexical alignment. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 16(1):77–106, 2010.

Yang Lingpeng, Ji Donghong, Zhou Guodong, and Nie Yu. Improving retrieval effectiveness by using key terms in top retrieved documents. In *European Conference on Information Retrieval*, pages 169–184. Springer, 2005.

Marina Litvak and Mark Last. Graph-based keyword extraction for single-document summarization. In *Coling 2008: Proceedings of the workshop multi-source multilingual information extraction and summarization*, pages 17–24, 2008.

Nikola Ljubešić, Tomaž Erjavec, and Darja Fišer. Kas-term and kas-biterm: Datasets and baselines for monolingual and bilingual terminology extraction from academic writing. *Digital Humanities*, 7, 2018.

Alfredo Maldonado and David Lewis. Self-tuning ongoing terminology extraction retrained on terminology validation decisions. In *Proceedings of The 12th International Conference on Terminology and Knowledge Engineering*, pages 91–100, 2016.

Aliya Nugumanova, Darkhan Akhmed-Zaki, Madina Mansurova, Yerzhan Baiburin, and Almasbek Maulit. Nmf-based approach to automatic term extraction. *Expert Systems with Applications*, 199:117179, 2022.

John Pavlopoulos and Ion Androutsopoulos. Aspect term extraction for sentiment analysis: New datasets, new evaluation measures and an improved unsupervised method. In *Proceedings of the 5th Workshop on Language Analysis for Social Media (LASM)*, pages 44–52, 2014.

Behrang Qasemizadeh and Siegfried Handschuh. Evaluation of technology term recognition with random indexing. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation*, 2014.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Andraž Repar, Vid Podpečan, Anže Vavpetič, Nada Lavrač, and Senja Pollak. TermEnsembler: An Ensemble Learning Approach to Bilingual Term Extraction and Alignment. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 25(1):93–120, 2019.

Ayla Rigouts Terryn, Veronique Hoste, Patrick Drouin, and Els Lefever. TermEval 2020: Shared Task on Automatic Term Extraction Using the Annotated Corpora for Term Extraction Research (ACTER) Dataset. In *6th International Workshop on Computational Terminology (COMPUTERM 2020)*, pages 85–94. European Language Resources Association (ELRA), 2020a.

Ayla Rigouts Terryn, Véronique Hoste, and Els Lefever. In No Uncertain Terms: A Dataset for Monolingual and Multilingual Automatic Term Extraction from Comparable Corpora. *Language Resources and Evaluation*, 54(2):385–418, 2020b.

Ayla Rigouts Terryn, Véronique Hoste, and Els Lefever. HAMLET: Hybrid Adaptable Machine Learning approach to Extract Terminology. *Terminology*, 2021.

Ayla Rigouts Terryn, Veronique Hoste, and Els Lefever. D-terminer: Online demo for monolingual and bilingual automatic term extraction. In *Proceedings of the TERM21 Workshop*, pages 33–40. Language Resources and Evaluation Conference (LREC 2022), 2022.

Ayla Rigouts Terryn, Véronique Hoste, and Els Lefever. Tagging terms in text: A supervised sequential labelling approach to automatic term extraction. *Terminology: international journal of theoretical and applied issues in specialized communication*, 28(1):157–189, 2022.

Hanh Thi Hong Tran, Matej Martinc, Antoine Doucet, and Senja Pollak. Can cross-domain term extraction benefit from cross-lingual transfer? In *Discovery Science: 25th International Conference, DS 2022, Montpellier, France, October 10–12, 2022, Proceedings*, pages 363–378. Springer, 2022a.

Hanh Thi Hong Tran, Matej Martinc, Andraz Pelicon, Antoine Doucet, and Senja Pollak. Ensembling transformers for cross-domain automatic term extraction. In *From Born-Physical to Born-Virtual: Augmenting Intelligence in Digital Libraries: 24th International Conference on Asian Digital Libraries, ICADL 2022, Hanoi, Vietnam, November 30–December 2, 2022, Proceedings*, pages 90–100. Springer, 2022b.

Hanh Thi Hong Tran, Matej Martinc, Jaya Caporusso, Antoine Doucet, and Senja Pollak. The recent advances in automatic term extraction: A survey. *arXiv preprint arXiv:2301.06767*, 2023.

Hanh Thi Hong Tran, Carlos-Emiliano González-Gallardo, Julien Delaunay, Antoine Doucet, and Senja Pollak. Is prompting what term extraction needs? In Elmar Nöth, Aleš Horák, and Petr Sojka, editors, *Text, Speech, and Dialogue*,

pages 17–29, Cham, 2024a. Springer Nature Switzerland. ISBN 978-3-031-70563-2.

Hanh Thi Hong Tran, Matej Martinc, Andraz Repar, Nikola Ljubešić, Antoine Doucet, and Senja Pollak. Can cross-domain term extraction benefit from cross-lingual transfer and nested term labeling? *Machine Learning*, pages 1–30, 2024b.

HTH Tran, Matej Martinc, A Doucet, and S Pollak. A transformer-based sequence-labeling approach to the slovenian cross-domain automatic term extraction. In *Slovenian Conference on Language Technologies and Digital Humanities*, 2022c.

Spela Vintar. Bilingual Term Recognition Revisited: The Bag-of-equivalents Term Alignment Approach and its Evaluation. *Terminology. International Journal of Theoretical and Applied Issues in Specialized Communication*, 16 (2):141–158, 2010.

Jiangyu Wang, Chong Feng, Fang Liu, Xinyan Li, and Xiaomei Wang. Extract then adjust: A two-stage approach for automatic term extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 236–247. Springer, 2023a.

Rui Wang, Wei Liu, and Chris McDonald. Featureless Domain-Specific Term Extraction with Minimal Labelled Data. In *Proceedings of the Australasian Language Technology Association Workshop 2016*, pages 103–112, 2016.

Xiao Wang, Weikang Zhou, Can Zu, Han Xia, Tianze Chen, Yuansen Zhang, Rui Zheng, Junjie Ye, Qi Zhang, Tao Gui, et al. Instructuie: multi-task instruction tuning for unified information extraction. *arXiv preprint arXiv:2304.08085*, 2023b.

Petra Wolf, Ulrike Bernardi, Christian Federmann, and Sabine Hunsicker. From statistical term extraction to hybrid machine translation. In *Proceedings of the 15th Annual conference of the European Association for Machine Translation*, 2011.

Yu Yuan, Jie Gao, and Yue Zhang. Supervised learning for robust term extraction. In *2017 International Conference on Asian Language Processing (IALP)*, pages 302–305. IEEE, 2017.

Ziqi Zhang, Jie Gao, and Fabio Ciravegna. Semre-rank: Improving automatic term extraction by incorporating semantic relatedness with personalised pagerank. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(5):1–41, 2018.