International Journal of Engineering Trends and Technology
ISSN: 2231-5381/ https://doi.org/10.14445/22315381/IJETT-V74I6P105

Volume 74 Issue 6, 66-74, June 2026

Original Article

Computational Tool for Automatic Term Extraction - ATEM

A. Morales Ríos[1], C.M. Medina Otálvaro[2], J.C. Blandón Andrade[3], C.M. Zapata Jaramillo[4]

[1,2,3]*Systems and Telecommunications Engineering, Catholic University of Pereira, Pereira, Colombia.*
[4]*Department of Computer and Decision Sciences, National University of Colombia, Medellín, Colombia.*

[1]*Corresponding Author : alejandro.morales@ucp.edu.co*

Received: 12 May 2025 Revised: 23 March 2026 Accepted: 28 March 2026 Published: 27 June 2026

Abstract *- Automatic term extraction enables the identification of the most representative terms within a corpus through computational processes. This process facilitates the creation of lexicographic materials or common databases, which are pivotal for knowledge acquisition in science as they help eliminate ambiguity in definitions pertaining to a specific domain. Specialized literature highlights the need for a common foundation on best practices for the Internet of Things (IoT) to consolidate knowledge and adapt new working methods. However, the manual creation of terminological resources is inefficient, does not keep pace with the rapid evolution of subjects, and is both time-consuming and costly. This article introduces ATEM, a term extraction tool for web and mobile environments that incorporates a hybrid method for identifying relevant terms in English-language scientific literature on IoT. ATEM is developed using a Service-Oriented Architecture (SOA) and employs programming languages such as JavaScript and Python. It also uses tools like the Flask framework and NLP-specific libraries such as NLTK and SpaCy. The computational tool includes the CValue algorithm, along with statistical and linguistic techniques in several steps: (i) corpus reception; (ii) text preprocessing; (iii) stop-word removal; (iv) Part-of-Speech (POS) tagging; and (v) filtering through linguistic and statistical rules. This results in a list of potential terms and a weight indicating their relevance within the corpus. The method was tested on five corpora from different domains, and ATEM processes and retrieves terms with 75% precision and 89% recall, highlighting its versatility across corpora. According to the tests, ATEM supports terminological extraction from IoT literature. It contributes to: (i) the development of lexicographic resources; (ii) language translation; and (iii) the creation of shared databases.*

Keywords *- Automatic Term Extraction, C-Value Algorithm, Hybrid Linguistic-Statistical Methods, Internet of Things, Natural Language Processing.*

1. Introduction

Natural Language Processing (NLP) is a fundamental discipline for understanding and computerizing human language mechanisms [1], with Information Extraction (IE) serving as a critical branch for retrieving structured data from unstructured text [2]. Within IE, Automatic Term Extraction (ATE) aims to identify and rank terminological units that represent the conceptual structure of a specialized domain [3]. The extracted terms are used to create lexicographic resources such as glossaries, ontologies, and shared databases, which are essential for reducing ambiguity and consolidating knowledge in scientific communication [4, 5].

Some authors emphasize the importance of usability and accessibility in term extraction tools. Gemkow et al. [6] present a system with a dual statistical filter for term extraction, aimed at generating updated glossaries in the field of requirements engineering. Golik et al. [7] propose an automatic term extractor based on a linguistic approach to produce lexicographic materials in the biomedical domain. Chatterjee and Kaushik [8] introduce a term extraction tool that employs linguistic rules in combination with the RENT algorithm. Luque and Seghiri [9] use the Sketch Engine tool to generate a shared vocabulary for the 3D printing domain.

However, the advancement of different technologies, such as the Internet of Things (IoT), poses a significant challenge. As noted by Jacobson et al. [10], there is an urgent need for a shared terminological foundation to support the standardization of best practices in the IoT domain. However, traditional manual terminology compilation is described as "long, repetitive, tedious, and costly" [11], often failing to keep pace with the exponential growth of scientific literature. Although recent progress in ATE has covered areas such as biomedicine [7, 12, 13] and agriculture [8], current research reveals a gap in tools specifically tailored for the dynamic vocabulary of IoT. Rigouts Terryn et al. [14] argue for the necessity of user-friendly online demonstrations to bridge the

gap between complex algorithms and end-users. Furthermore, existing ATE solutions often rely on desktop-based environments or command-line interfaces, lacking the portability required by modern researchers who need accessible, cross-platform tools [15, 16].

This paper develops ATEM, a computational, cross-platform term extraction tool for web and mobile environments that includes NLP techniques and the CValue algorithm for terminological extraction from English language scientific literature on IoT. Unlike generic keyword extractors like YAKE! [17] that rely solely on statistical local features, or domain-specific tools limited to single platforms, ATEM employs a hybrid approach. The tool is developed under Service-Oriented Architecture (SOA) and uses technologies such as JavaScript with its React library, Python, and its Flask framework, along with specialized NLP libraries like NLTK and SpaCy. This study aims to design an extraction process adapted to the linguistic and conceptual characteristics of the IoT-related literature, as well as to evaluate the performance of the proposed hybrid method in relation to statistical baselines and provide a multi-platform architecture that improves portability and usability for researchers.

Our work contributes the following: i) the development of a specialized extraction pipeline that addresses the linguistic particularities of IoT literature; ii) the implementation of a hybrid method (Linguistic + Statistical) that outperforms purely statistical approaches in terms of precision; and iii) an innovative architectural proposal (SOA) that ensures tool portability across web and mobile environments, taking into account the usability limitations noted in previous studies. This paper is organized as follows: Section 2, related work. Section 3 shows the ATEM development method. Section 4 presents the results. Section 5 discusses these findings from the article. Finally, Section 6 offers the conclusions.

2. Background

Automatic Term Extraction started as simple frequency-based counts and evolved to complex hybrid systems that combine linguistic analysis with statistical metrics [15, 18], [19]. This section presents several contributions, classifying them according to their methodological approach and application domain.

2.1. Statistical and Unsupervised Approaches

Nakagawa and Mori [20] propose a powerful method focused on compound nouns and the statistics of their components, enabling the identification of multiword terms. Campos et al. [17] focus on statistical features of the text without requiring a training corpus. Although these methods are computationally efficient, they often lack the semantic depth required to filter non-terminological noise in highly specialized domains.

2.2. Linguistic and Hybrid Approaches

Liwei [21] demonstrated that hybrid methods, which combine linguistic patterns with statistical evaluation (such as DC-Value and information entropy), significantly outperform purely statistical methods in technical domains. This is further supported by Gemkow et al. [6], who developed a hybrid system for Requirements Engineering that employs a dual statistical filter to update glossaries, and by Cram and Daille [22], who developed TermSuite, a tool leveraging UIMA token regular expressions to manage term variants at both morphological and syntactic levels.

2.3. Domain-Specific Implementations

The effectiveness of Automatic Term Extraction (ATE) is directly related to domain customization.

2.3.1. Biomedicine

Golik et al. [7] perform extraction through the analysis of participles and prepositional complements. Santamaría and Krallinger [12] present CUTEXT, a system for Spanish medical resources, while Kafando et al. [13] recently proposed ITEXT-BIO, a system that combines the C-Value method with pattern mining to achieve high precision in biomedical analysis.

2.3.2. Agriculture and other Fields

Chatterjee and Kaushik [8] present a system for the agricultural domain, highlighting how specific regular expression (regex) patterns can improve recall. In the context of cybersecurity, Aygun and Kaya [23] used Word2Vec to determine semantic affinity, while Luque and Seghiri [9] employed corpus-based methods for 3D printing terminology.

2.4. The Research Gap

Despite these advancements, several limitations remain. As noted by Pais and Ion [24] in the TermEval 2020 evaluation, system performance varies considerably depending on the architectural implementation and the degree of domain adaptation. Most existing tools, such as TBXTools by Oliver and Vásquez [25], are robust but primarily designed for desktop environments, a characteristic that limits their accessibility for collaborative and mobile research. In addition, although rigorous systems exist for domains such as medicine and agriculture, the Internet of Things (IoT) domain-characterized by a convergence of hardware, software, and network terminology-still lacks a specialized tool that combines high-precision hybrid extraction with the accessibility of a web/mobile Service-Oriented Architecture (SOA). ATEM addresses this gap by adapting the C-Value algorithm within a modern and portable software framework. Table 1 presents a comparative framework of terminology extraction methods, with the purpose of illustrating their application domains, methodological approaches, and main contributions.

Table 1. Summary of terminology extraction methods

Authors	Domain /Focus	Approach /Method	Key Contribution/ Focus
Nakagawa & Mori [20]	General	Statistical (Compound Nouns)	Scoring based on component statistics
Campos et al. [17]	General	Statistical (Unsupervised)	YAKE! algorithm; lightweight, single-document extraction
Golik et al. [7]	Requirements Engineering	Hybrid (Stat + Ling)	Automatic glossary extraction from specifications
Cram & Daille [22]	General	Hybrid / Rule-based	TermSuite: handling term variants and multilingual support.
Chatterjee & Kaushik [8]	Agriculture	Hybrid (Regex + NLP)	RENT algorithm; domain-specific pattern recognition
Santamaría & Krallinger [12]	Medicine	Hybrid	CUTEXT system; multilingual resource construction
Kafando et al. [13]	Biomedicine	Hybrid (C-Value + Mining)	ITEXT-BIO; intelligent extraction for bio-analysis
Aygun & Kaya [23]	Cybersecurity	Semantic / ML	Word2Vec approach for semantic affinity
Liwei [21]	Technical (Chinese)	Hybrid (DC-value + Entropy)	Addresses unithood and termhood in patent mining
Rigouts Terryn et al. [14]	General	Hybrid	D-Terminer; emphasis on online usability and user interface
ATEM (Proposed)	IoT	Hybrid (C-Value + NLP)	Service-Oriented Architecture (SOA) for web/mobile portability

3. Materials and Methods

The computational tool is developed and validated on a computer with an 11th-generation Intel Core i5 processor, 8 GB of RAM, and Windows 10 as the operating system. A traditional waterfall software development lifecycle model is used, applying the five stages proposed by Sommerville [26]: (i) requirements analysis; (ii) design; (iii) Code development; (iv) testing; and (v) deployment. The tool's architecture follows a service-oriented paradigm [27], decoupling the software into a Backend component (which includes web services via the HTTP protocol) and one or more Frontend components that consume this service from different platforms. This architecture enables cross-platform development, resulting in both web and mobile versions. The Frontend is coded in JavaScript with React and React libraries, while the Backend is implemented in Python with the Flask framework and the integration of NLP libraries such as SpaCy and NLTK. Python 3.10 and Node.js 18 were used as the runtime environments, and the NLP components were executed using SpaCy 3.x (model en_core_web_sm) and NLTK 3.8 to ensure reproducibility across executions. A computational tool is proposed as a solution for automatically extracting terms from scientific literature on IoT, and it is developed in the stages presented in Figure 1.

The process begins at the Frontend with the reception of the IoT corpus, in PDF format or plain text. The corpus is sent to the Backend via an HTTP POST request, where the terminological extraction process takes place. PDF documents were converted to plain text using the pdfminer-six libraries to ensure consistent extraction across sources. The tool assumes that all corpora are in English and excludes scanned documents without OCR preprocessing. To mitigate security issues, the system includes SSL protections on the client and backend sides. On the Backend, no ethical aspects related to the information are compromised, as user inputs are not stored.

3.1. Preprocessing

Text processing in NLP requires a preprocessing stage, where unwanted characters and strange symbols are removed, and the text is standardized to be cleaner and more manageable [28]. The first step performed in the Backend involves applying preprocessing to the corpus. Initially, the removal of strange characters is carried out using (1) for each character. Then, the resulting text is converted to a lowercase format to standardize the corpus. Additionally, non-alphabetic symbols are removed using an ASCII-based filter, and tokenization is performed with SpaCy to ensure consistent segmentation of the text.

$$V = (C \in A) \wedge \neg S \quad (1)$$

Where:

- V: Boolean variable representing the validity of the character.
- C: Represents the character to be evaluated.
- A: A set of characters that belong to the alphabet according to the ASCII code.
- S: Boolean condition that evaluates if the character is a strange symbol.

The terminological extraction tool focuses on generating lexicographic materials from the scientific literature on IoT. That is, the received corpus includes various scientific papers, where there are several repeated words (names of the journal, titles, and labels characteristic of academic papers) that do not form a term but affect the efficacy of term extraction. Therefore, once the text is cleaned of strange characters and standardized to lowercase, a Gazetteer process with a customized Stop-list is applied to eliminate the words listed in Table 2. The customized stop-list contains 42 entries and is applied through an exact-match gazetteer filter to avoid removing partially relevant sequences.

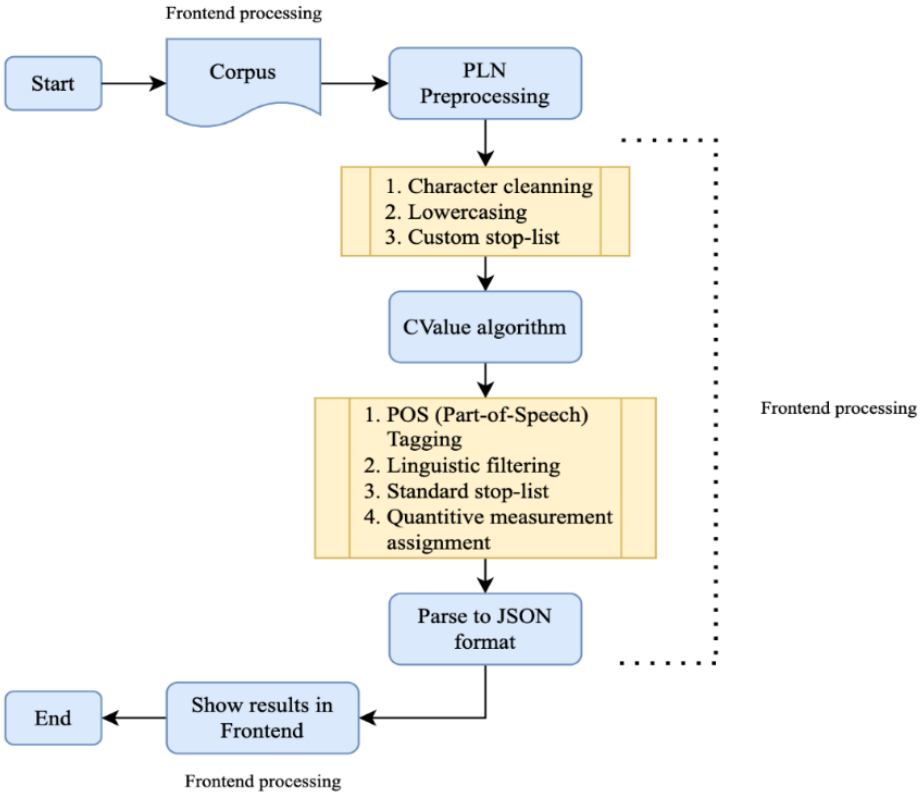

Fig. 1 Flow chart on proposed method

Table 2. Personalized stop-list

Category	Words
Journal name	ACM, Ebsco, Scopus, IEEE Xplore, Digital Library, ScienceDirect, Web of Science
Titles	Introduction, abstract, background, related works, keywords, methods, materials, discussion, conclusion, recommendation
Others	published at, authors, conference, available at

Finally, the purpose of text preprocessing is to standardize the text, making it cleaner and facilitating the continuation of the terminological extraction process.

3.2. CValue Algorithm

CValue is a domain-independent algorithm designed to automatically extract multiword terms [29], unlike other algorithms that only capture single-word terms. CValue is based on the postulate that terms in technical domains (such as IoT) often include several words, and the occurrence of nested terms is more significant compared to single-word terms [30]. This algorithm operates in two different instances, first linguistically and then statistically.

3.2.1. CValue-Linguistic Components

First, the process of POS tagging is carried out for each word in the corpus, determining its grammatical unit.

Figure 2 presents an example of this process applied to the phrase “the large-scale abuse of Internet of Things devices”.

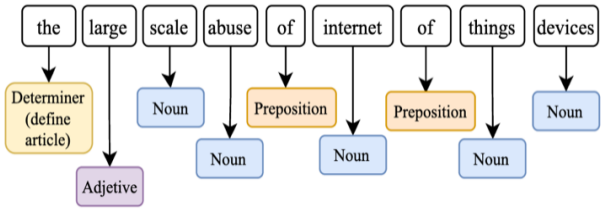

Fig. 2 POSTagger example

$$((Adj \mid Noun) + \mid ((Adj \mid Noun) * (NounPrep) ?) (Adj \mid Noun) *) Noun.$$

Fig. 3 Linguistic filter

Subsequently, the tagged text is processed through a linguistic filter, following the guidelines presented by Frantzi et al. [29] as shown in Figure 3. The application of the linguistic filter determines the word combinations that constitute a term, resulting in a list of potential terms.

Finally, a Stop-list is applied to this list of candidate terms to exclude stop words, that is, words that by themselves carry no significant meaning. The linguistic patterns follow the noun-phrase structure (Adj|Noun)+ Noun, operationalized through SpaCy POS tags (JJ, NN, NNS, NNP), allowing consistent detection of multiword term candidates.

3.2.2. CValue-Statistical Components

Each potential term in the candidate term list is processed in the statistical component. The goal of this stage is to assign a quantitative measure to each potential term to indicate its relevance within the text, taking into account important aspects such as its overall frequency of occurrence. To this end, the authors of CValue propose a method (Figure 4) that allows them to obtain the probability that a candidate phrase is true.

$$CValue = \begin{cases} \log_2 |a| \cdot f(a) & \text{if } a \text{ does not appear in other candidates} \\ \log_2 |a| \cdot (f(a) - \frac{1}{P(T_a)} \sum_{b \in T_a} f(b)) & \text{otherwise} \end{cases}$$

Where:

a : Candidate phrase.

$|a|$: Length of the candidate phrase.

$f(a)$: Frequency of ocurrence of a in the corpus.

Ta : Set of candidates of a longer length containing a.

$P(Ta)$: Number of candidates that include a.

$\sum_{b \in Ta} f(b)$: Number of ocurrences of$a$ as a substring of b.

Fig. 4 CValue algorithm

The process generates a set of candidate terms, each assigned a statistical score that reflects its relevance within the corpus; the higher the score, the greater the term's relevance. To ensure reproducibility, the extraction was configured with a minimum frequency threshold of two occurrences and a maximum candidate length of five words, thereby aligning with standard recommendations for multiword term extraction.

3.3. Format as JSON

The list of terms and their corresponding weights is converted into JSON format and returned as the response to the HTTP POST request. Subsequently, we include information such as execution time, the length of the analyzed corpus, and the total number of extracted terms. The JSON output includes the term, its C-Value score, frequency, nested occurrences, processing time, and corpus size expressed in both characters and tokens.

3.4. Presenting Results on the Frontend

Finally, the Frontend instance receives the response from the terminology extraction process. The results are presented in a clear, user-friendly format, including terms in a paginated table, process metrics, a word cloud graphic, and the option to download the terms in XLSX format. The table is paginated to handle large vocabularies, and the XLSX export preserves term weights and frequencies for external validation.

4. Results

It was possible to process a corpus constructed with academic papers in English, with the computational tool proposed in this paper, for performing terminology extraction from the scientific literature on IoT. The tool responded promptly and presented acceptable metrics with a precision of 75% and a recall of 89%.

4.1. Reception and Preprocessing of the Corpus

The process begins on the Frontend, where an English-language corpus of 6,526,786 characters in length was loaded. This corpus was sent to the Backend via an HTTP-POST request as a string.

Then, the Backend received the corpus and began preprocessing the text by removing special characters, standardizing it to lowercase format, and finally applying the custom stop-list. After preprocessing, the corpus experienced a 20% reduction in length, resulting in a corpus of 5,221,428 characters.

4.2. CValue Algorithm

Figure 5 shows the result of applying the CValue algorithm, which successfully identified 123,567 candidate terms. The result was adjusted in a dynamic array, where each position corresponds to a candidate term and its weight.

```
"terms": [
  [
    "text": "best practices",
    "value": 6.043173008425223
  ],
  [
    "text": "level analysis of
    the state",
    "value": 4.64962120684067
  ],
  [ "text": "sets of best
    practices",
    "value": 4.00720447336091
  ],
  [ "text": "analysis of the
    state",
    "value": 4.00720447336031
  ]
]
```


Fig. 5 CValue algorithm results

4.3. Presentation of Results on the Frontend

In the final stage, the Frontend received the result of the terminology-extraction output in JSON format. The presentation of the results is shown in Figure 6.

Fig. 6 Presentation of results

5. Discussion

This paper supports the need for computational methods that perform terminology extraction to generate lexicographic materials, especially in the domain of IoT.

Similarly, it agrees with other authors on the high impact of lexicographic materials in the scientific community because they enable the creation of common foundations, knowledge structuring, and the elimination of ambiguities in definitions.

This paper presents the development of a computational tool for automatic term extraction from scientific literature in IoT, and proposes the generation of lexicographic materials. Initially, the computational tool was tested with a corpus of 6,526,786 characters in length, corresponding to a compilation of 97 scientific papers on IoT, to validate the correct functioning of the tool.

The system was able to process the corpus and perform the extraction, generating a list of 123,567 terms in an execution time of 9 minutes and 10 seconds. The validation of the computational tool was carried out based on precision and recall criteria, following the calculations explained by Barrón Cedeño [19]. For this purpose, 5 corpora from different domains were used:

- Software Engineering
- Medicine
- Social migrations
- Agriculture
- IoT

First, terms in each corpus were identified and counted manually. Then, each document was processed with the tool, and candidate terms were extracted. Subsequently, precision and recall formulas were applied, and a confusion matrix was constructed to map the results shown in Table 3.

The variable ’T’ refers to the terms in the corpus, while ’Ts’ refers to the terms in common between the corpus and those found by the ATEM tool. The variable ’S’ refers to the number of terms found by the ATEM tool.

With this data, Precision and Recall for each domain were subsequently calculated [19]. The average results showed that the tool has a precision of 75% and a recall of 89%. The corpus with the highest precision was in the field of agriculture, while the document on IoT achieved the highest recall.

The tool does present an opportunity for improvement, which may be related to the ambiguities of natural language, as well as the criteria for term relevance within a domain and human errors in manual term detection.

The results demonstrated that the tool allows the loading of corpora from different domains, including IoT, to then identify candidate terms. Some methods from other authors also achieved interesting results.

Chatterjee and Kaushik [8] developed a terminology extraction method in the agricultural sector domain and achieved a precision above 80% and a recall greater than 60%.

Gemkow et al. [6] created a system to generate glossaries for the Requirements Engineering domain, achieving a recall of 74.9% and a precision of 73.4%. Golik et al. [7] designed a linguistic method to extract terms from texts related to biomedicine and obtained a precision of 55.5% and a recall of 33.9%.

Table 3. Results of validation for the ATEM tool

Corpus	T (Terms)	S (System Terms)	Ts (True Positive)	Precision (Ts/S)	Recall (Ts/T)	Accuracy (Ts / (T + S – Ts))
Engineering Requirements	30	38	27	71,05%	90%	65,85%
Medicine	72	87	67	77,01%	93,05%	72,83%
Migrations	51	59	38	64,40%	74,50%	52,77%
Agriculture	68	72	63	87,5%	92,64%	81,82%
IoT	72	94	70	74,46%	97,22%	72,92%
Average				75%	89%	70%

5.1. Comparative Discussion with the State of the Art

When compared to the main methods found in earlier research, it is easier to understand how well ATEM works. ATEM has a higher overall recall (89%) and a competitive precision (75%) than a purely statistical system like YAKE! [17] or Nakagawa and Mori's early frequency-based methods [20]. This shows that hybrid methods have a more balanced extraction capacity in a specialized area. This is in line with what Liwei [21] found, which was that adding linguistic constraints greatly increases extraction reliability, cutting down on noise from non-terminological units.

Gemkow et al. [6] and Cram and Daille's TermSuit [22] are two examples of hybrid systems that work well because they use both linguistic filters and statistical metrics. ATEM's accuracy is like that of the dual-filter method in Requirements Engineering (75%), and it is better than the recall reported by Gemkow et al. (73.4%). This shows that ATEM's adaptation of the C-Value algorithm for broad-domain use is effective. ATEM also works across many different domains, whereas RENT in agriculture [8] and ITEXT-BIO in biomedicine [13] only work in one highly structured domain. However, ATEM gets results that are in the same performance range as those tools. ATEM had the highest recall in the IoT corpus (97.2%), which shows that it is a good tool for extracting terms from a field that is changing quickly.

Also, even though some current tools work well in their own areas, they are often not useful because of architectural limits. Systems like CUTEXT [12], ITEXT-BIO [13], TBXTools [25], and TermSuite [22] are mostly desktop-based or need command-line interaction, which makes them hard to use for interdisciplinary teams or researchers to go. On the other hand, ATEM uses a Service-Oriented Architecture that can be used on both the web and mobile devices. This improves one of the problems that Pais and Ion [24] pointed out in TermEval about portability and adaptability across domains. This benefit allows ATEM to connect high-precision extraction methods with the need for lightweight, cross-platform tools that work with modern research workflows. The comparative analysis shows that ATEM is competitive with established domain-specific systems overall. It also has strong methods, can be used in many different domains, and is much easier to use. These strengths make ATEM a useful tool for extracting terminology in fast-changing tech fields, especially the IoT, where new lexicographic resources are needed but not enough research has been done on them.

6. Conclusions and Future Work

This article presents a computational tool for terminology extraction in the Internet Of Things (IoT) domain. It describes the development of a cross-platform solution that overcomes the usability limitations of traditional extraction software, which is typically restricted to desktop applications. The adoption of a Service-Oriented Architecture (SOA), supported by technologies such as React (JavaScript) and the Flask framework (Python), proved effective in decoupling internal computational processing (Backend) from the user interface (Frontend). This architectural design enables operation across both web and mobile environments without compromising system performance.

Regarding the extraction method, the hybrid approach we propose-integrating NLP-based linguistic patterns with the statistical C-Value algorithm-demonstrated strong performance compared to frequency-only methods. Validation across five corpora yielded an average precision of 75% and a recall of 89%. Notably, the system achieved its highest recall (97%) in the IoT corpus, which confirms its effectiveness in capturing multiword terminology. This outcome directly contributes to strengthening best practices and reducing conceptual ambiguity in IoT-related research.

The tool exhibits several domain-specific limitations: (i) the current linguistic workflow is restricted to English-language corpora, which constrains its immediate applicability in multilingual research contexts; (ii) the system's accuracy is highly dependent on the quality of the input text; distortions introduced during PDF-to-text conversion may occasionally interfere with linguistic tagging and introduce noise into the candidate list; and (iii) although the noun-phrase patterns cover most technical terms, the system may overlook complex or nonstandard syntactic variants that fall outside the defined linguistic rules.

As future work, the following directions are proposed: First, it is proposed to extend the tool's capabilities to support multilingual extraction, specifically adapting the linguistic filters for Spanish and Portuguese to broaden the tool's impact. Second, plans include the integration of Deep Learning

techniques, such as BERT-based contextual embeddings, to improve the semantic filtering of candidates and increase precision beyond the current statistical thresholds. Finally, incorporating an Optical Character Recognition (OCR) module will allow the processing of scanned documents, further expanding the scope of analyzable scientific literature.

References

[1] Augusto Cortez Vásquez, Hugo Vega Huerta, and Jaime Pariona Quispe, "Natural Language Processing," *Journal of Systems and Informatics Research*, vol. 6, no. 2, pp. 45-54, 2009. [Google Scholar]

[2] Alexander Gelbukh, "Natural Language Processing and its Applications," *Computer Sapiens*, vol. 1, pp. 6-11, 2010. [Google Scholar]

[3] Rosa Estopà Bagot, "Terminology Extraction: Elements for the Construction of an Extractor," *TradTerm*, vol. 7, pp. 225-250, 2001. [Google Scholar]

[4] Dag I.K. Sjøberg et al., *Building Theories in Software Engineering*, Guide to Advanced Empirical Software Engineering, Springer, London, pp. 312-336, 2008. [CrossRef] [Google Scholar] [Publisher Link]

[5] Alexander Alvaro Barón Salazar, "*Model for the Unified Definition of Practice as a Theoretical Construct in Software Engineering*," Doctoral Thesis, National University of Colombia, Medellín, Colombia, 2019. [Google Scholar] [Publisher Link]

[6] Tim Gemkow et al., "Automatic Glossary Term Extraction from Large-Scale Requirements Specifications," *2018 IEEE 26th International Requirements Engineering Conference (RE)*, Banff, AB, Canada, pp. 412-417, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[7] Wiktoria Golik et al., "Improving Term Extraction with Linguistic Analysis in the Biomedical Domain," *Research in Computing Science*, vol. 70, pp. 157-172, 2013. [Google Scholar]

[8] Niladri Chatterjee, and Neha Kaushik, "RENT: Regular Expression and NLP-based Term Extraction Scheme for Agricultural Domain," *Proceedings of the International Conference on Data Engineering and Communication Technology, ICDECT*, Springer, Singapore, vol. 468, pp. 511-522, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[9] Angela Luque Giraldez, and Miriam Seghiri Domínguez, "3DCOR: Creation of a Bilingual (English-Spanish) Corpus-based Glossary for Translating Technical Specifications of 3D Printers," *Proceedings of the III International Congress on Computational and Corpus Linguistics - CILCC 2020 and the V Workshop on Automated Text Processing and Corpus - WoPATeC*, University of Antioquia, Medellín, pp. 86-89, 2020. [Google Scholar]

[10] Ivar Jacobson, Ian Spence, and Pan-weing, "Is there a Single Method for the Internet of Things? Essence can Keep Software Development for the IoT from Becoming Unwieldy," *Queue*, vol. 15, no. 3, pp. 25-51, 2017. [Google Scholar]

[11] Tatiana Gornostay et al., "Terminology Extraction, Translation Tools and Comparable Corpora: TTC Concept, Midterm Progress and Achieved Results," *LREC 2012 Workshop on Creating Cross-Language Resources for Disconnected Languages and Styles (CREDISLAS)*, 2012. [Google Scholar] [Publisher Link]

[12] Jesus Santamaria, and Martin Krallinger, "Construction of Medical Terminological Resources for Spanish: The CUTEXT Term Extraction System and Biomedical Term Repositories," *Natural Language Processing*, vol. 61, pp. 49-56, 2018. [Google Scholar]

[13] Rodrique Kafando et al., "ITEXT-BIO: Intelligent Term EXTraction for BIOmedical Analysis," *Health Information Science and Systems*, vol. 9, no. 1, pp. 1-23, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[14] Ayla Rigouts Terryn, Veronique Hoste, and Els Lefever, "D-Terminer: Online Demo for Monolingual and Bilingual Automatic Term Extraction," *Proceedings of the Workshop on Terminology in the 21st Century: Many Faces, Many Places*, European Language Resources Association, Marseille, France, pp. 33-40, 2022. [Google Scholar] [Publisher Link]

[15] Amelia De Irazazabal, and Erika Schwarz, *Terminological Databases as an AID to the Translator*, III Complutense Encounters on Translation, Cervantes Institute, 1993. [Google Scholar] [Publisher Link]

[16] M. Teresa Cabré, "TERMINTEGRAL: A Platform for Building Terminological Databases and Ontologies," *Linguistica Antverpiensia, New Series-Themes in Translation Studies*, vol. 3, pp. 245-261, 2004. [CrossRef] [Google Scholar] [Publisher Link]

[17] Ricardo Campos et al., "YAKE! Keyword Extraction from Single Documents using Multiple Local Features," *Information Sciences*, vol. 509, pp. 257-289, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[18] Sue Ellen Wright, and Gerhard Budin, *Handbook of Terminology Management: Application-Oriented Terminology Management*, John Benjamins Publishing, vol. 2, 2001. [CrossRef] [Google Scholar] [Publisher Link]

[19] Luis Alberto Barrón Cedeño, "*Automatic Extraction of Terms in Defining Contexts*," Master's Thesis, UNAM-Faculty of Engineering, Mexico, D.F., 2007. [Google Scholar] [Publisher Link]

[20] Hiroshi Nakagawa, and Tatsunori Mori, "A Simple but Powerful Automatic Term Extraction Method," *COLING-02: COMPUTERM 2002: Second International Workshop on Computational Terminology*, 2002. [Google Scholar]

[21] Zhang Liwei, "Chinese Technical Terminology Extraction based on DC-Value and Information Entropy," *Scientific Reports*, vol. 12, no. 1, pp. 1-12, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[22] Damien Cram, and Beatrice Daille, "TermSuite: Terminology Extraction with Term Variant Detection," *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics-System Demonstrations*, Berlin, Germany, pp. 13-18, 2016. [Google Scholar]

[23] İrfan AygÜn, and Mehmet Kaya, "Automatic Term Extraction on Turkish Scientific Texts," *2020 International Conference on Decision Aid Sciences and Application (DASA)*, Sakheer, Bahrain, pp. 1037-1040, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] Vasile Pais, and Radu Ion, "TermEval 2020: RACAI's Automatic Term Extraction System," *Proceedings of the 6th International Workshop on Computational Terminology*, European Language Resources Association, Marseille, France, pp. 101-105, 2020. [Google Scholar] [Publisher Link]

[25] Antoni Oliver, and Mercè Vàzquez, "TBXTools: A Free, Fast and Flexible Tool for Automatic Terminology Extraction," *Proceedings of the International Conference Recent Advances in Natural Language Processing*, Hissar, Bulgaria, pp. 473-479, 2015. [Google Scholar]

[26] Ian Sommerville, *Software Engineering*, 10th ed., Pearson Education Limited, 2016. [Publisher Link]

[27] P. Clements et al., "Documenting Software Architectures: Views and Beyond," *25th International Conference on Software Engineering, 2003. Proceedings.*, Portland, OR, USA, pp. 740-741, 2003. [CrossRef] [Google Scholar] [Publisher Link]

[28] Daniel Jurafsky, and James H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Boulder, CO, USA: Pearson/Prentice Hall, 2008. [Google Scholar]

[29] Katerina Frantzi, Sophia Ananiadou, and Hideki Mima, "Automatic Recognition of Multi-Word Terms: The C-Value/NC-Value Method," *International Journal on Digital Libraries*, vol. 3, no. 2, pp. 115-130, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[30] Ziqi Zhang, Jie Gao, and Fabio Ciravegna, "JATE 2.0: Java Automatic Term Extraction with Apache Solr," *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, European Language Resources Association, Portorož, Slovenia, pp. 2262-2269, 2016. [Google Scholar] [Publisher Link]