AcX: system, techniques, and experiments for Acronym eXpansion

João L. M. Pereira
INESC-ID and IST, Universidade de Lisboa, and
University of Amsterdam
joaoplmpereira@tecnico.ulisboa.pt

João Casanova*
Hitachi Vantara
joao.casanova@hitachivantara.com

Helena Galhardas
INESC-ID and IST, Universidade de Lisboa
helena.galhardas@tecnico.ulisboa.pt

Dennis Shasha
Courant Institute, New York University
shasha@cs.nyu.edu

Abstract

In this information-accumulating world, each of us must learn continuously. To participate in a new field, or even a sub-field, one must be aware of the terminology including the acronyms that specialists know so well, but newcomers do not.

Building on state-of-the art acronym tools, our *end-to-end acronym expander system* called *AcX* takes a document, identifies its acronyms, and suggests expansions that are either found in the document or appropriate given the subject matter of the document. As far as we know, AcX is the first open source and extensible system for acronym expansion that allows mixing and matching of different inference modules. As of now, AcX works for English, French, and Portuguese with other languages in progress.

This paper describes the design and implementation of AcX, proposes *three new acronym expansion benchmarks*, compares state-of-the-art techniques on them, and proposes ensemble techniques that improve on any single technique. Finally, the paper evaluates the performance of AcX in end-to-end experiments on a human-annotated dataset of Wikipedia documents. Our experiments show that human performance is still better than the best automated approaches. Thus, achieving Acronym Expansion at a human level is still a rich and open challenge.

PVLDB Reference Format:
João L. M. Pereira, João Casanova, Helena Galhardas, and Dennis Shasha. AcX: system, techniques, and experiments for Acronym eXpansion. PVLDB, 15(11): XXX-XXX, 2022.
doi:XX.XX/XXX.XX

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/joaolmpereira/acronym-expander.

1 Introduction

Take a great historical literary figure of any culture and put him or her in the present. That person might barely understand a newspaper headline partly because of the acronyms that have no expansion. For example, a headline of the Washington Post on May 6, 2022 reads: "FDA limits use of J&J vaccine over rare blood clots".

Contemporary scholars encounter similar challenges when entering a new field or a sister field. A typical document about wireless communication is practically unintelligible to a computer scientist with its talk of 3gPP, 5G. BS etc. Documents written for specialists often neglect even to define the acronyms they use [6].

Further, the proper expansion of an acronym depends on context. For example, "ISBN" can mean International Standard Book Number in a publishing context, Integrated Satellite Business Network in a satellite communication context, and International Society for Behavioral Neuroscience in a cognitive scientific context. Thus, any system that hopes to help readers understand the intended meaning of an undefined acronym in a document must expand that acronym using its context.

1.1 High Level Architecture of an Acronym Expansion System

An *end-to-end acronym expander system* comprises the following two components: (i) Recognition (also known as **extraction** and **identification**) of each acronym and (when present) its expansion within a text. For example, if a given text has "ISBN (Integrated Satellite Business Network)" then "ISBN" would be the acronym and "Integrated Satellite Business Network" would be the expansion. We call this *in-expansion* because this can be done for a particular document based solely on its own text. (ii) In the case that an acronym is not expanded in the text of a document, *out-expansion* chooses an expansion from a large parsed corpus (training corpus) of other documents (e.g., Wikipedia).

This paper makes the following system and data contributions:

- The **end-to-end Acronym eXpander (AcX) system** accepts a text document as input and outputs a list of acronym-expansion pairs for the acronyms found in the document, whether or not the expansions are in the document. As far as we know, AcX is the first open source and extensible system for acronym expansion that allows both the mixing and the combination of different inference modules.
- A **benchmark of in-expansion techniques (in-expansion benchmark)**. We make use of four biomedical datasets previously proposed in the literature (i.e., Medstract [29], BIOADI [29], Schwartz and Hearst [29], and Ab3p [29])

*This work was performed while the author was a MSc student at IST, Universidade de Lisboa.

Proceedings of the VLDB Endowment, Vol. 15, No. 11 ISSN 2150-8097.
doi:XX.XX/XXX.XX

and one of the biggest sentence based datasets from the scientific domain (i.e., SciAI [80]). Additionally, we have created a new dataset composed of Wikipedia documents from the *Computing* category.

- A **benchmark of out-expansion techniques (out-expansion benchmark)**. We evaluate out-expansion techniques on three datasets from different domains previously used in related work that contain documents (i.e., MSH [62], SciWISE [62], and CSWiki [77]) and one that is constructed from independent sentences from the scientific domain (i.e., SciAD [79] revised by Egan and Bohannon [20]).
- A **benchmark of end-to-end acronym expander systems (end-to-end benchmark)**. We create the first end-to-end dataset of human-annotated documents that includes both in- and out-expansions. We have built a human-annotated end-to-end benchmark because (i) previous annotated in-expansion datasets do not include acronyms with out-expansions and (ii) previous out-expansion datasets use automatic mechanisms to identify acronyms, but those mechanisms are neither accurate nor complete. Thus, human annotation offers a kind of gold standard.

This paper is organized as follows: Section 2 presents related work, particularly for acronym expansion, but including references to entity linking. Section 3 describes the AcX system. The next three sections (Sections 4, 5, and 6) describe the proposed benchmarks and analyze the benchmark experimental results. Section 7 contains an error analysis of acronym expansion. Finally, Section 8 presents the main conclusions and ideas for future work.

2 Related Work

This section describes the work that is closely relevant to acronym expansion, including in-expansion only (Section 2.1), out-expansion only (Section 2.2), and end-to-end systems (Section 2.3).

2.1 In-expansion

Pustejovsky et al. [63] present a technique that parses the input text in order to reduce the context within which to search for a candidate expansion. Schwartz and Hearst (SH) [67] describe a technique that considers two possible placements of expansions and acronyms in text (before or after), and chooses the correct expansion by matching acronym characters with potential expansion characters.

The MadDog [78] in-expander introduces variations of SH technique [67] which refine the candidate expansions using a sequence of rules. Nabeesath and Nazeer [66] suggest new pattern heuristics as well as space reduction heuristics. Azimi et al. [5] use the same patterns as Schwartz and Hearst (SH) [67] but relax the heuristics for acronym-expansion extraction: an acronym simply needs to be a token composed of capital letters of some length n and an expansion should be composed of n tokens.

Yarygina and Vassilieva [89] incorporate user feedback and two decision tree classifiers in order to filter candidate acronym-expansion pairs. Glass et al. [24] propose a technique that focuses on several languages other than English, and scores candidate pairs by using word embeddings in order to measure the similarity between candidate acronyms and expansions.

Liu et al. [45] and Veyseh et al. [80] formalize the task of finding expansions for an acronym as a sequence labeling problem solvable by Conditional Random Fields (CRFs) [39] based techniques. The SciDr [72] in-expander and Zhu et al. [91] also interpret acronym-expansion extraction as a sequence labeling task and make use of pre-trained BERT-based models coupled with ensemble techniques to achieve higher model performance than previous techniques. SciBERT is a language model based on Transformers and pre-trained on research papers from Semantic Scholar[1]. SciBERT is fine-tuned in SciDr [72] with training data for the sequence labeling task. The SciDr [72] in-expander uses an ensemble (blending) process [71]. It splits the training data into train and validation sets. Five different SciBERT models (e.g., number of epochs and learning rate values) are constructed based on the training set. The expansions of the SciBERT models and of the rule-based baseline technique of the SDU@AAAI competition[2] based on Schwartz and Hearst [67], and additional syntactic features extracted from the word-to-tag mapping are used to train five Conditional Random Fields (CRFs) [39] in a 5-fold cross-validation setting. The ensemble technique for these CRF models is based on hard voting.

Chopard and Spasić [14] also make use of word embeddings and calculate the *Word Mover's Distance* [38] in order to select the correct expansion from the candidate expansions of an acronym. Jacobs et al. [30] makes use of a Support Vector Machine (SVM) to select the correct expansion from several candidate expansions for an acronym. Similarly, to select the correct expansion for biomedical documents, Kuo et al. [37] use an SVM as well as Logistic Regression and Naïve Bayes models.

Another line of work extracts acronyms not from text but from Web Data like query click logs [31, 76].

The fields of Named Entity Recognition and Coreference Resolution address similar tasks. Named Entity Recognition [85] finds entities mentioned in texts and labels them with high level categories like *person* and *organization*; or, for special applications, as molecular biology entities covered in BioNLP tasks [17, 25] like *cells* and *proteins. Coreference Resolution* [41, 47, 55] *is the task of finding all expressions that refer to the same entity in a text*[3]. Thus, references like *I*, *my*, *she* or even *this person* may refer to a given person entity. Coreference Resolution can be applied to in-expansion where the expansion and the acronym are references to the same entity.

2.2 Out-expansion

Classic Context Vector [2, 42, 62] is a typical baseline for out-expansion. It represents the context of an acronym/expansion x by the frequencies of the words in all documents containing x. Li et al. [42] propose two techniques based on word embeddings from Word2Vec [49] to address the out-expansion problem. Their best technique, called Surrounding Based Embedding, combines the Word2Vec embeddings of the words surrounding the acronym or the expansion. Similarly to Surrounding Based Embedding, Ciosici et al.

[1]https://www.semanticscholar.org/
[2]https://github.com/amirveyseh/AAAI-21-SDU-shared-task-1-AI/blob/master/code/character_match.py
[3]https://nlp.stanford.edu/projects/coref.shtml

[16] propose Unsupervised Acronym Disambiguation that replaces each expansion occurrence in a collection of text documents by a normalized token and retrains the Word2Vec google news model [49] on that collection. The resulting model produces an embedding for each normalized token, i.e., an expansion embedding.

Thakker et al. [77] creates document vector embeddings, using Doc2Vec, for each document. For each set of documents D containing an expansion for an acronym A, the system trains a Doc2Vec model on D which is used to infer the embedding for an input document i containing an undefined acronym A.

Charbonnier and Wartena [12] proposed an out-expansion technique based on Word2Vec embeddings weighted by Term Frequency-Inverse Document Frequency scores to find out-expansions for acronyms in scientific document captions.

MadDog [78] proposes a sequential model to encode context in sentences followed by a feedforward network to classify the input sentence with an expansion. Competitors of the SDU@AAAI competition [79] mainly use pre-trained language models based on Transformer neural networks like BERT [19] and SciBERT [7]. SciDr [72] formulates the out-expansion problem as a substring prediction task. Given a list of expansions concatenated with a sentence as input, it uses the pre-trained language model SciBERT [7] and retrains that model in 5 cross-validations of the sentences dataset to predict the substring, i.e., start and end word indices corresponding to the predicted expansion. The authors also assemble additional SciBERT models trained on external data.

A related line of work explored the expansion of acronyms in enterprise texts [22, 43]. For instance, in Li et al. [43], enterprise textual documents as well as Wikipedia documents are used as training data. Other works explored acronym out-expansion in biomedical domains [44, 50, 51, 56, 63, 75, 83, 84, 90]. In our work, we explore the general acronym expansion problem where the input document domain or source is not previously known.

Entity Disambiguation (ED) (often referred to as Entity Linking) is the task that links an entity found in text by Named Entity Recognition (NER) to a knowledge base, usually Wikipedia pages [53, 69, 70]. This field is analogous to out-expansion because an expansion can be seen as (and in some cases is) a Wikipedia page title. Several techniques have been proposed to address this task. The survey [69] identifies the work of [87] that is part of the LUKE project[4] as the best or one of the best on several datasets, some based on Wikipedia. LUKE (Language Understanding with Knowledge-based Embeddings) [86] is a pre-trained language model that learns to predict masked words and entities. LUKE also employs a global model that, given a set of entities in a document, assigns a ranking among these entities based on confidence. Other works on Entity Disambiguation explore the task in the face of limited resources [23, 48, 58, 81, 82, 88] corresponding to zero-shot learning settings where the labels (i.e., entities) in the test set are unknown at training time. Such circumstances occur in acronym out-expansion because some expansions have a very low frequency in document collections, sometimes appearing just once.

Moreover, Entity Disambiguation works have explored Natural Language Techniques that we also used in order to represent documents like Term Frequency–Inverse Document Frequency (TF-IDF) [34] in [13], Latent Dirichlet Allocation (LDA) [9] in [61], and Doc2Vec [40] in [68, 92].

At BioNLP Open Shared Tasks 2019, Bacteria Biotope [10] considers the goal of linking microbial taxa, habitats, and phenotype to biological knowledge bases. To enrich the input, the authors provided the in-expansions for the acronyms found in their dataset using Ab3p [73]. The winner [35] matched the Word2Vec embeddings of entities in the text with the concepts in the knowledge base. However, an acronym as an entity mention would have the same Word2Vec embeddings regardless of the document.

The Cross-Document Coreference Resolution task [46] matches entities in one document to entities in other documents. Thus, acronym out-expansion is a special case of Cross-Document Coreference Resolution. However, out-expansion is easier, because the various documents containing a particular expansion can be compared collectively with the input document to determine whether the expansion is appropriate for the acronym in the input document.

Less directly related, but insightful, is the literature on Word Sense Disambiguation (WSD) [52, 54] because that work also must make use of the context around a token (in our case, an acronym; in the word sense literature, a word). Raganato et al. [64] proposed a benchmark for word sense disambiguation.

2.3 End-to-end Acronym Expanders

To our knowledge, systems that expand acronyms use a predefined dictionary of acronym-expansions [1, 26] as opposed to trying to discover the proper expansion based on context.

Only two end-to-end systems use context for out-expansion. First, Ciosici and Assent [15] propose an end-to-end abbreviation/acronym expansion system architecture that performs out-expansion. Unfortunately, their demo paper provides few technical details and their code is proprietary.

The MadDog system [78] contains a rule-based in-expander technique that improves on [67] and an out-expander based on neural networks: a sequential model to encode context followed by a feedforward network to classify the input with an expansion. They also trained their models on a large corpus of sentences.

Neither of these systems provides a framework with easy plug-in for different in and out-expansions techniques nor uses other data sources. Moreover, neither was evaluated on an end-to-end acronym expander benchmark.

3 AcX: an End-to-end Acronym eXpander System

The AcX system (see Figure 1) consists of: (i) A *Database Creation* process which generates an *Expansion Database*[5] that contains documents, acronyms and their corresponding in-expansions. The Expansion Database also associates each <acronym, in-expansion> pair with a *representation* of the document where that acronym and in-expansion were found. The representation characterizes

[4]https://github.com/studio-ousia/luke

[5]When benchmarking, the expansion database will provide us with both a training set and a test set.

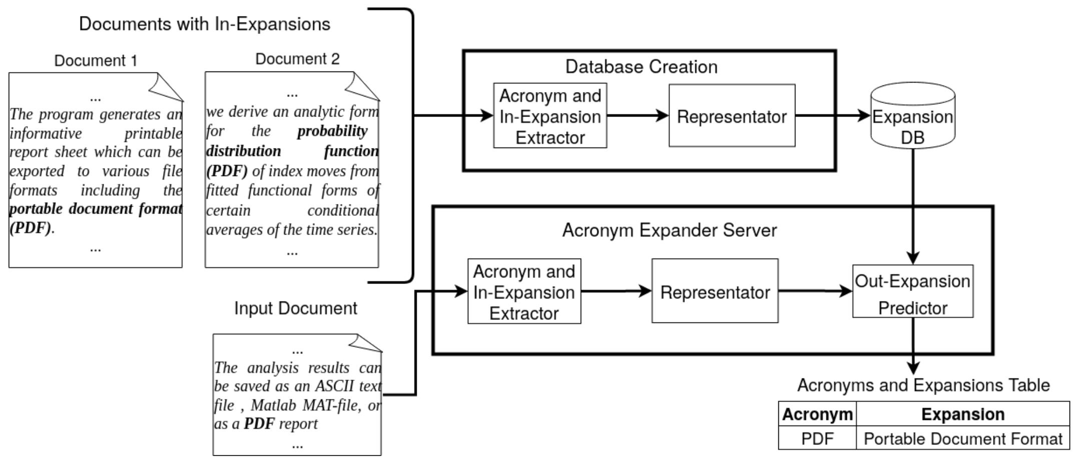

Figure 1: Acronym eXpander (AcX) system. The top stream denotes the creation of the Expansion Database that associates each <acronym, in-expansion> pair with some representation of the document(s) where that pair was found. The bottom stream shows the processing of an input document d by combining acronym in-expansion when possible and a representation of d. For an acronym A with no expansion in d, the representation of d is compared with the representations in the Expansion Database of documents containing A to find the context-appropriate expansion.

the content of the document. To support other domains and languages, we pass documents in the desired domains/languages to the Database Creation process. (ii) The *Acronym Expander Server* that accepts one document at a time from a user and outputs a list of acronyms found in the input document and the corresponding expansions found by the system (whether as in-expansions or as out-expansions).

For each document with in-expansions, the *Database Creation* process runs the following pipeline:

(1) an *Acronym and In-Expansion Extractor* obtains the <acronym, expansion> pairs from the document using only within-document evidence.
(2) a *Representator* (there are many possible representators e.g., Latent Dirichlet Allocation that output topics) maps the document to a document representation that holds document contextual information.
(3) the *Expansion Database* stores the in-expansions, acronyms, and document representations on disk, currently SQLite [28].

Given a new input document d supplied by a user, the *Acronym Expander Server* executes the following pipeline:

(1) applies the *Acronym and In-Expansion Extractor* used to build the Expansion Database to extract all the acronyms having expansions in the input document d.
(2) utilizes the same *Representator* (say, topics from Latent Dirichlet Allocation) used to characterize each document in the Expansion Database to map d to a document representation.
(3) for each acronym A having no in-expansion in d, the server runs the *Out-Expansion Predictor* to choose a context-appropriate out-expansion. Formally, an expansion E is selected for an acronym A in d if the representations of the documents $doc(A, E)$ with expansion E share more characteristics with the representation of d by some criteria (e.g., closest cosine similarities or labeled by some machine learning classifier for A) than the documents in $doc(A, E')$ for every alternative expansion E'. Thus, for example, if the context of d is publishing, then "PDF" should likely expand to "Portable Document Format" but if the context of d is probability or statistics, then "PDF" should expand to "probability distribution function."

For a language other than English, the in- and out-expansion techniques should be tuned to the new language. They may benefit from changing preprocessing steps such as tokenization for the new language or from adopting a language model trained on the new language or even adopting a multilanguage model.

3.1 Acronym and In-Expansion Extraction

Acronym and in-expansion extraction can use rule-based or machine learning techniques. In our rule-based implementations (i.e. Schwartz and Hearst [67] and MadDog [78]), we used roughly the following three-step process as described in [57]:

(1) *Acronym extraction:* identifies acronyms in a document, e.g., PDF in Figure 1. We modified Schwartz and Hearst [67] to find candidate acronyms even when there is no expansion found in a given document. The technique excludes tokens in which all alphabetic characters except the first character are lower case. We also reject acronyms of two characters where the first is a letter and the second is a dot "." to avoid person names.
(2) *Candidate expansion extraction:* builds candidate pairs of acronyms and possible in-expansions <acronym, expansion>

from information in the document, e.g., <PDF, formats including the portable document format> from Document 1 in Figure 1.

(3) *Candidate refinement:* evaluates each candidate pair using a variety of heuristics (e.g., find the shortest expansion that matches the acronym) to obtain a final in-expansion for each acronym that has at least one candidate in-expansion within the document, e.g., portable document format from <PDF, formats including the portable document format>.

For the in-expanders of SciBERT and SciDr, the extraction of acronyms and expansions is formalized as a sequence tagging problem where each token can have one of three tags: (i) a token in an acronym (e.g., CD in CD-ROM), (ii) a token in an expansion, or (iii) other token. For example, from Document 1 in Figure 1, PDF would be tagged as an acronym token, each token portable, document, and format would be tagged as a token in an expansion. The remaining tokens in Document 1 would have the "other token" tag. AcX builds a machine learning model on the tagged data. The output of such machine learning models is then converted to acronym-expansion pairs by matching the acronym characters against expansions.

Our system supports ensemble in-expansion through SciDr. That ensemble technique can be easily extended to include additional in-expansion techniques.

3.2 Representator

Representors in the AcX system summarize documents in order to capture knowledge about their semantics. Although AcX supports sentence-level out-expansion techniques, using the whole document is more effective than using just parts of the text because the whole document captures the overall context better.

Some representators assign a set of topic terms to a document. If two documents have many topic terms in common, then they are considered to be semantically related.

Other representators use embeddings [40] to characterize a document. An *embedding* is a vector of real numbers in a high dimensional space. Embedding techniques map an object encoded in a one-hot representation, a very sparse and high dimensional vector of binary values, into a very dense and lower dimensional vector of real values (i.e., embedding). A small distance between embedding vectors suggests document similarity.

AcX encloses several techniques that can semantically represent an entire set of documents that contain the same expansion for a given acronym. Specifically, let $docs(A, E)$ denote the set of full document texts in which a given acronym A is defined by a single expansion E (e.g., all documents in which acronym PDF is explicitly expanded as portable document format):

Here are some representations of such a collection of documents:

- *Classic Context Vector (CCV)* [2], represents an expansion E by the set of words in $docs(A, E)$ along with their counts.
- *Document Context Vector (DCV)* (our variation of context vector), builds on context vector, however it represents each document $d \in docs(A, E)$ individually by the set of word occurrences in d. For example, the word occurrences corresponding to Document 2 in Figure 1 would contain, among others, the values {of: 3}, {the: 2}, {derive: 1}, {analytic: 1}, {form: 1}.
- *Term Frequency–Inverse Document Frequency (TF-IDF)* [34], gives a large weight to a term t in each document $d \in docs(A, E)$ if t is found frequently in d and infrequently in the entire document corpus. Each document is then characterized by its highly weighted terms. For example, the TF-IDF score for the word the in Document 2 in Figure 1 is $\frac{2}{27} \cdot log(\frac{2}{2}) = 0$ because this word appears in both documents.
- *Latent Dirichlet Allocation (LDA)* [9] assigns topics to documents using a Dirichlet probabilistic model. For example, Document 2 in Figure 1 could be represented by the following topics: topic1={{analytics: 0.7}, {series: 0.3}} and topic2={{functional: 0.8}, {form: 0.2}}.
- *Doc2Vec* [40] is a document embedding technique based on Word2Vec [49] which assigns vectors to words in such a way that words that appear in the same context have a high cosine similarity. For example, the words functional and conditional would be assigned similar vectors. Thus, using the principles of Word2Vec, Doc2Vec assigns vectors to entire documents. For example, documents 1 and 2 in Figure 1 would be assigned mutually distant vectors.
- *Sentence Bidirectional Encoder Representations from Transformers (SBERT) [65]* constructs sentence embeddings that can be compared to determine sentence similarity. AcX splits the input document text to fit into the SBERT input limit (e.g., 384 tokens), and then we average the resulting embedding vectors to get a document representation.

3.3 Out-Expansion Predictor

To choose an out-expansion for an acronym A in an input document d having no expansion for A, the Out-Expansion Predictor component considers each candidate out-expansion E for A and compares d to some representation of $docs(A, E)$.

In the case of Classic Context Vector (CCV), we compare d with the vector representation of $docs(A, E)$. For the remaining techniques, we compare d with each document representation of $d' \in docs(A, E)$.

Using cosine similarity, the Out-Expansion Predictor will choose an out-expansion E over a different expansion E' if any document $d' \in docs(A, E)$ is more similar to d than all $d'' \in docs(A, E')$.

The AcX system also supports classification-based approaches that work as follows. Consider all the documents, denoted $alldocs(A)$ containing in-expansions of acronym A. Some documents in $alldocs$ (A) have an in-expansion of $E1$ for A, some have $E2$ for A and so on. Given the representations of documents in $alldocs(A)$ as features and the expansions ($E1$, $E2$, etc) as labels, the out-expansion problem becomes a machine learning classification problem. When a new document d is given to AcX, the representation of d is input to the classifier which labels d with an expansion.

The classifiers we support so far are:

- *Support Vector Machines (SVMs)* [18] fit a hyper-plane that optimally separates binary labeled data in the feature space. Non-binary classification is performed by a "one-vs-all" technique where a binary SVM classifier predicts with a certain probability if an input document belongs to a particular class (where each class corresponds to a particular

expansion). The class (and therefore expansion) with the highest probability is selected. We used the LibLinear [21] implementation included in sckit-learn toolkit [60].
- *Logistic Regression (LR)* [32] fits a logistic function to classify binary classes (again a class corresponds to an expansion). Non binary classification is again performed by a "one-vs-all" technique. We used the LibLinear [21] implementation included in scikit-learn toolkit [60].
- *Random Forests (RF)* [11] fit a particular number of decision trees (default 100) trained on randomly selected samples. There will be one random forest per acronym A. The representation of a document having no in-expansion for A will be input to the random forest. Each tree will predict one expansion with some probability. The random forest selects the class whose average probability is the highest. We used the scikit-learn [60] implementation.

In addition to these classifiers, for evaluation purposes or for anyone who wants to try other techniques, AcX supports the following additional techniques: Surrounding Based Embedding (**SBE**) [42], Thakker et al. [77], Unsupervised Abbreviation Disambiguation (**UAD**) [16], the SciDr out-expander (**SciDr-out**) [72], the MadDog out-expander (**MadDog-out**) [78], and **LUKE** [87], a state-of-the-art technique for Entity Disambiguation. For UAD, SciDr-out and MadDog-out, AcX performs sentence segmentation and, given the results from each sentence, decides which expansion to assign to the text. For UAD, we select the most frequent predicted expansion among the sentences in the document.

We have extended SciDr-out to consider all the sentences containing the acronym A instead of just one sentence as in SciDr-out's original implementation. SciDr-out associates an acronym with its possible expansions concatenated together. The system then finds the substring of that concatenated string with the highest probability and outputs that as the expansion. For example, the concatenated expansion of "PDF" might be "probability density function portable document format". SciDr-out will choose some substring of that concatenated expansion.

We have extended MadDog-out to enable it to train in any new documents, instead of using only their original machine learning models. MadDog-out processes the last sentence of any document containing acronym A to determine the most likely expansion.

For LUKE, we had to modify the internals to work with acronyms and expansions. We use their pre-trained model and perform fine-tuning in our training data using the procedure described by the authors in [87], except that we allow the entity embeddings (now expansion embeddings) to be updated during training. This modification allows the generation of embeddings for expansions out of the original model vocabulary.

4 In-expansion Benchmark, Evaluation and Results

We describe our benchmark of in-expansion techniques in Section 4.1 and evaluate state-of-the-art techniques on this benchmark in Section 4.2.

4.1 A Benchmark of In-expansion Techniques

This section describes the benchmark we developed to evaluate in-expansion techniques. Section 4.1.1 details the datasets used in this benchmark. Section 4.1.2 lists the in-expansion techniques that we implemented for this benchmark. Section 4.1.3 defines the metrics that we used to evaluate the in-expansion extraction techniques.

4.1.1 Datasets The datasets included in this in-expansion benchmark are:

Medstract: This dataset is composed of 199 randomly selected MEDLINE[6] abstracts from the results of a query on the term "gene". The abstracts were manually annotated and then the annotations were corrected and improved by Schwartz and Hearst [67], Ao and Takagi [3], Pustejovsky et al. [63], Yarygina and Vassilieva [89] and Doğan et al. [29]. We use the last revised version of Doğan et al. [29] that contains 159 acronym-expansion pairs.

Schwartz and Hearst: This dataset consists of 1,000 randomly selected MEDLINE abstracts from the results of a query on the term "yeast". The abstracts were manually annotated by Schwartz and Hearst [67] and revised by Doğan et al. [29]. The revised version that we use contains 979 acronym-expansion pairs.

BIOADI: This dataset contains 1,201 abstracts from the BioCreative II gene normalization dataset. The dataset was original annotated by Kuo et al. [37] and revised by Doğan et al. [29]. It contains 1,720 acronym-expansion pairs.

Ab3P: This dataset results from the random selection of MEDLINE 1,250 abstracts. The dataset was manually annotated by Sohn et al. [73]. We use the revised version of Doğan et al. [29] that contains 1 223 acronym-expansion pairs.

SciAI: This dataset results from processing 6,786 English arXiv[7] papers. Those papers were split into sentences and sent to Amazon Mechanical Turk (MTurk) to be annotated by humans, resulting in 9,775 acronym-expansion pairs. This dataset was annotated for both acronyms and acronym-expansion pairs. The final dataset has 17,506 sentences, where 1% do not contain acronyms and 24% do not contain expansions. We use the SDU@AAAI competition [79] version[8] that was initially proposed by Veyseh et al. [80].

End-to-end: We developed a dataset that consists of 163 English Wikipedia documents randomly selected from the *Computing* category[9] in Wikipedia. It contains 1,139 acronym-expansion pairs. For this in-expansion benchmark, we consider only the acronym-expansion pairs with expansion in text. (Later, in Section 6.1, we use the whole set of acronym-expansions pairs to evaluate end-to-end systems.) Each document was annotated by two students among our 50 or so volunteers. We collected as many annotations as possible during approximately four weeks. Each student annotated at least two documents. During the annotation process,

[6]https://www.nlm.nih.gov/bsd/medline.html
[7]https://arxiv.org/
[8]https://github.com/amirveyseh/AAAI-21-SDU-shared-task-1-AI
[9]https://en.wikipedia.org/wiki/Category:Computing

each student identified each acronym in the document and mapped it to an expansion. Each acronym-expansion pair was labeled by the annotators, indicating whether the expansion was present in text. Any conflict between annotators was manually resolved by the authors. The Inter-Annotator Agreement (IAA) among each annotators (excluding the third annotator, the reviewer) using Krippendorff's alpha [36] with the MASI distance metric [59] is 0.68 for in-expansion pairs and 0.33 for out-expansion pairs. In a hypothetical scenario, if both annotators had given the same acronym-expansions, then the score would be 1. In this case, the human annotators disagree on out-expansions more often than on in-expansions. This is unsurprising because out-expansion requires consulting text sources other than the document at hand.

4.1.2 In-expansion techniques This benchmark includes the following in-expansion techniques (that are supported by our AcX system described in Section 3):

Rule-based: Schwartz and Hearst (**SH**) [67] technique and the MadDog [78] in-expansion (**MadDog-in**) technique which builds on the Schwartz and Hearst algorithm.

Machine Learning: **SciBERT** based technique used in [72] and the SciDr [72] in-expansion (**SciDr-in**) technique which ensembles SciBERT models and a rule-based technique based on SH with Conditional Random Fields. Moreover, we consider models used by these machine learning techniques that are trained *with external data* besides the individual training sets of each dataset. The external data is composed of Medstract, Schwartz and Hearst, BIOADI, and Ab3P train sets if the test set is biomedical. For SciAI and End-to-end test sets, the external data consists of all train sets (i.e., biomedical datasets, SciAI, and End-to-end).

4.1.3 Performance metrics Our benchmark uses the following metrics. The metrics apply to acronyms alone as well as to acronym-expansion pairs. The acronyms can be either in singular or plural form to be considered equal, and the expansions are equal if their lower case versions without dashes have an edit distance less than 3 or if the first 4 characters of each word are equal. If the same acronym or pair appears several times in the same document, it is counted only once:

Acronym Pair Precision: the number of correctly extracted acronym pairs divided by the number of acronym pairs extracted by that technique over all documents.

Acronym Pair Recall: the number of correctly extracted acronym pairs divided by the number of distinct acronym pairs present over all documents.

Acronym Pair F1-measure: the harmonic mean of the *precision* and *recall* of the system.

Training time: CPU or GPU time in seconds to train the machine-learning models that are used by the in-expansion technique.

Execution time: CPU or GPU time in seconds that the in-expansion technique takes to extract acronym-expansion pairs from a document in the dataset.

4.2 In-expansion Experimental Evaluation

In this section, we evaluate the in-expansion techniques using the benchmark presented in Section 4.1.

Setup. The in-expansion experiments were performed on a machine with an Intel® Core™ i5-4690K CPU with 4 cores, and 16 GB of RAM and an NVIDIA GeForce GTX 1070. Only SciBERT and SciDr-in used the GPU.

Results. We report the Precision, Recall, and F1-measure values for the average of the biomedical datasets (i.e., Medstract, Schwartz and Hearst, BIOADI and Ab3P), SciAI and End-to-end datasets in Table 1. The additional external data used to train SciBERT and SciDr-in for the biomedical application includes the data of all biomedical datasets excluding the test set (30%). For SciAI and End-to-end datasets, the external data used to train SciBERT and SciDr-in includes all documents in the other datasets (i.e., Medstract, Schwartz and Hearst, BIOADI, Ab3p, SciAI, and End-to-end). We report the fine-grained results per biomedical dataset and execution times per dataset in the extended version of this paper [10].

Interpretation: In this in-expansion benchmark, rule-based techniques SH and MadDog-in generally perform best for all datasets. The one exception is on the SciAI dataset where machine learning techniques from SciDr-in and SciBERT work better.

Rule-based systems work well for in-expansion, because acronyms follow human-understood rules, viz. roughly, acronyms should be in upper-case, each letter should represent a word, and the expansion should either precede or follow the first use. So it is natural that a rule-based system would do well. Machine learning work better when given more examples (SciAI dataset), however even ensembled with a rule-based technique (SciDr) the results were generally inferior to using the rule-based technique by itself.

While the expansions found by the rule-based techniques are not a superset of those found by the machine learning techniques, SciDr often fails because it adds extra words to the expansion string. On the other hand, SciDr can find unusual cases where not all acronym chars belong in the expansion, e.g., expansion *PIN-FORMED* of *pin1*.

Execution time analysis. Regarding execution time, we observed from our experiments that the rule-based techniques are much faster than the machine learning techniques. SH is the fastest technique on every single dataset taking less than **0.06** seconds on average to extract acronym-expansion pairs from a document.

In summary: Use a rule-based system for in-expansion, either **SH** or **MadDog-in**.

5 Out-expansion Benchmark, Evaluation and Results

We describe our benchmark of out-expansion techniques in Section 5.1 and evaluate state-of-the-art techniques on this benchmark in Section 5.2.

[10]Temporary location: web.tecnico.ulisboa.pt/ist164790/acx_extended.pdf

Acronym and In-expansion Technique	Biomedical Datasets – Avg.						SciAI						End-to-end					
	Acronym			Pair			Acronym			Pair			Acronym			Pair		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
SH	**99.31%**	**81.88%**	**89.72%**	**96.33%**	**79.52%**	**87.07%**	96.02%	82.36%	88.67%	92.85%	79.64%	85.74%	91.00%	**70.54%**	79.47%	86.00%	**66.67%**	75.10%
MadDog-in	98.45%	58.65%	73.35%	92.34%	54.97%	68.82%	**98.63%**	86.72%	92.30%	**96.91%**	85.21%	90.68%	**92.78%**	69.76%	**79.64%**	**88.65%**	**66.67%**	**76.10%**
SciBERT	85.22%	68.22%	75.71%	70.01%	56.01%	62.15%	95.69%	**94.05%**	94.86%	92.21%	**90.64%**	91.42%	65.62%	48.83%	55.99%	58.34%	43.41%	49.77%
SciBERT with External data	88.27%	75.98%	81.65%	75.97%	65.38%	70.27%	96.18%	**94.05%**	**95.11%**	92.50%	90.45%	91.46%	49.67%	58.91%	53.90%	45.09%	53.48%	48.93%
SciDr-in	90.56%	63.40%	74.53%	78.13%	54.76%	64.34%	97.47%	92.47%	94.90%	94.47%	89.63%	**91.98%**	77.08%	57.36%	65.77%	68.75%	51.16%	58.66%
SciDr-in with External data	91.91%	76.12%	83.26%	85.55%	70.86%	77.50%	97.58%	91.78%	94.59%	93.81%	88.24%	90.94%	86.36%	58.91%	70.04%	81.81%	55.81%	66.35%

Table 1: In-expansion techniques Precision, Recall, and F1-measures for acronym and pair extraction and for the average of the biomedical datasets, SciAI dataset, and User Generated dataset.

5.1 A Benchmark of Out-expansion Techniques

Section 5.1.1 describes the datasets used in this benchmark. Section 5.1.2 explains the steps used to prepare those datasets. Section 5.1.3 lists the out-expansion techniques included in the benchmark, grouped by type. Finally, Section 5.1.4 describes the metrics to evaluate those out-expansion techniques.

5.1.1 Datasets The datasets included in our out-expansion benchmark are:

MSH dataset [33] contains biomedical document abstracts from the MEDLINE (Medical Literature Analysis and Retrieval System Online) corpus used in Li et al. [42], Prokofyev et al. [62]. This dataset was automatically annotated using citations from MEDLINE and the ambiguous terms with MeSH headings identified in the Metathesaurus[11]. We use the original texts and the revised labels from Li et al. [42];

SciWISE dataset consists of document abstracts of the Physics dataset used in Li et al. [42] and Prokofyev et al. [62]. This dataset was annotated by human experts, and it includes expansions either containing at least 2 words or a single word with at least 14 characters.

CSWiki (Computer Science Wikipedia) dataset created in Thakker et al. [77] contains documents from different fields that contain acronyms used in computer science. Expansions were extracted by parsing the content of English Wikipedia disambiguation pages of acronyms used in computer science (e.g., https://en.wikipedia.org/wiki/PDF_(disambiguation)).

SciAD This dataset was prepared for the out-expansion SDU@ AAAI-21 competition [79]. It is based on the SciAI in-expansion dataset, described in Section 4.1.1. We use the revised version[12] created by Egan and Bohannon [20] who removed duplicate sentences from the original training and validation sets.

5.1.2 Data Preparation The data preparation steps are roughly the same for each out-expansion technique:

(1) **Dataset Splitting:** We split each dataset into *train* and *test* sets (respectively 70% and 30% of the documents of the original dataset). We then apply 5-fold cross validation on the train dataset in order to tune the hyperparameters of each out-expansion technique. The hyperparameter-tuned technique is then tested on the yet unseen 30% of the data.
(2) **Expansion Consolidation:** For the expansions of acronym A in each dataset, we apply an approximate duplicate detection process that groups expansion strings that correspond to the same expansion meaning. For example, *portable document format* and *Portable-Document-Formats* are two distinct strings that refer to the same real expansion. As criteria, we consider two expansions to be equal if their lower case versions without dashes have an edit-distance less than 3 or if the first 4 characters of each word are equal. Equal expansions are consolidated by mapping them all to the most frequent expansion.
(3) **Expansion Removal:** When testing the accuracy of out-expansion techniques on some document d, we associate any acronym A in the document with its in-expansion $In(A)$, if present. Then, we replace all occurrences of the in-expansion $In(A)$ in text by A alone.
(4) **Tokenization:** We apply the word tokenization from the Natural Language Toolkit (NLTK) [8] to obtain only alphanumeric tokens. Additionally, we remove stop words using NLTK and numeric tokens;
(5) **Token Normalization:** We transform each token into its stem, e.g., probable, probability, and probabilities all map to probabl. We use the Porter Stemmer algorithm from NLTK.

The preparation of the MSH and SciWISE datasets follows the preprocessing reported in Li et al. [42], so we apply all the preparation steps above except token normalization. The five steps are consistent with the pre-processing steps used in Thakker et al. [77] for the CSWiki dataset. For SciDr-out and MadDog-out, we apply only the first three steps, because these techniques replace the last two steps with steps that depend on the language models of the neural networks they use.

5.1.3 Out-expansion Techniques This benchmark includes the following groups of out-expansion techniques:

Classical Techniques: We use two baselines: **Random** which randomly assigns a possible expansion to an acronym; and **Most Frequent** which always selects the most frequent expansion found in our training data as measured by the number of occurrences in distinct documents. We use the Cosine similarity (**Cossim**) with the Classic Context Vector (**CCV**) [42], Document Context Vector (**DCV**) - variant of Classic for each document, Surrounding Based Embedding (**SBE**) [42], and **Thakker** et al. [77].

[11] https://www.nlm.nih.gov/research/umls/knowledge_sources/metathesaurus
[12] https://github.com/PrimerAI/sdu-data

Sentence-oriented Techniques: We include related work techniques that expect a sentence as input (instead of a document) and adapt them as described in the AcX overview (Section 3.3). These include Unsupervised Abbreviation Disambiguation (**UAD**) [16], MadDog [78] out-expander (**MadDog-out**), and SciDr [72] out-expander (**SciDr-out**). We also use **SciDr-out with External Data** consisting of the Wikipedia pages that contain an expansion found in the training data.

Representator Techniques: We include **Cossim** with the document representation techniques described in Section 3.2, that we have adapted from natural language processing: Term Frequency-Inverse Document Frequency (**TF-IDF**), Latent Dirchlet Allocation (**LDA**), **Doc2Vec** , and Sentence Bidirectional Encoder Representations from Transformers (**SBERT**). We used SBERT model *all-mpnet-base-v2*, the top performing model in Sentence Similarity tasks (14 datasets)[13]. all-pnet-base-v2[14] is based on MPNet model [74] that outperforms BERT and RoBERTA in both quality and speed. *all-mpnet-base-v2* was trained on one billion sentences pairs from a diverse set of data sources.

Classification Techniques: We created a complete new class of out-expansion techniques that use the outputs of a representator as features for a Machine Learning classifier, specifically, Random Forests (**RF**), Logistic Regression (**LR**), and Support Vector Machines (**SVM**). Each acronym has its own classifier trained with the features of the documents that contain an expansion for the acronym (e.g., acronym PDF will have a random forest RandFor(PDF) based on documents that contain an in-expansion for PDF). Based on the features of a target document, the classifier will choose the appropriate expansion as explained in Section 3.3.

Combination of Representator Techniques: The final type of out-expansion techniques that we assembled consists of combining two representators' outputs, namely the Doc2Vec with a Context Vector (either Classic or Document), as input to predictors: **CCV + Doc2Vec** and **DCV + Doc2Vec**. Combinations are constructed by concatenating the outputs together into a single feature vector.

Ensembler Techniques: We support two ensembler techniques: **Hard** voting where each technique votes for its preferred expansion regardless of its confidence; and **Soft** voting that takes the averages of confidences per expansion. The confidences are normalized at the individual technique level in such a way that their sum is 1. For the experiments, we assembled the following 7 out-expansion techniques: Cossim with CCV, Cossim with TF-IDF, Cossim with Do2Vec, SVM with Doc2Vec, Cossim with SBERT, SVM with SBERT, and SciDr-out.

5.1.4 Performance Metrics Our benchmark uses the following metrics:

Out-expansion accuracy: is the accuracy of predicting the right expansion for a given acronym in a textual document. Intuitively, this is the fraction of acronym-expansions that are correctly predicted. Accuracy is also used in previous out-expansion works [16, 42, 77] and analogous benchmarks, e.g., for Word-Sense-Disambiguation [64]. Note that an acronym may appear many times in the same document and many times across documents. In our measure, if A is in k documents, it is counted k times, but if A is present j times in the same document, it is counted only once in that document.

Out-Expansion macro averages: Recently, Veyseh et al. [78][80] started using a different set of metrics that we have implemented and measured for completeness. Those metrics are macro-averages of Precision, Recall and F1-measures for acronym-expansions pairs. So, we calculate precision, recall, and F1-measure independently for each acronym-expansion in the training data.

Representator execution time: is the execution time to create representations of training documents.

Average execution time per document: is the average execution time to predict expansions for acronyms in a document.

5.2 Out-expansion Experimental Results

Setup. For out-expansion on the benchmark presented in Section 5.1, we ran the experiments on a GoogleCloud platform[15] machine with the following specifications: Intel Broadwell CPU platform with 8 cores, 30GB to 80GB of RAM (Random Access Memory). For SBERT, MadDog-out, SciDr-out, and LUKE half of a Tesla K80 GPU board was used.

To reduce the duration of experiments, we first find the representator's hyperparameters using cosine similarity (a parameter-less metric). Next, we find the best out-expansion predictor model hyperparameters.

Results. Table 2 reports the out-expansion accuracy and macro F1-measure to predict the expansions of acronyms in a document for each dataset; and the average document processing times. The *Technique Group* column identifies the out-expansion group that the technique belongs to, as organized in Section 5.1.3 (e.g., Classical). The *Predictors* column identifies the out-expansion predictor technique (e.g., Cossim or an ML classifier) that takes a given document representation to predict an expansion (e.g., Cossim). The *Representators* column indicates the technique used to generate a document representation (e.g., Doc2Vec). We did not run SciDr-out with External Data on CSWiki dataset because the external data (i.e., Wikipedia data) would overlap with CSWiki itself. The execution time of each ensemble technique is just the additional time required to decide on an expansion given the input predictions and confidence measures.

In these out-expansion experiments, we measure the accuracy and macro F1 only on the acronym-expansions pairs whose acronym is ambiguous (i.e., have at least two expansions in the training data) and whose in-expansions are in the training data.

[13] https://www.sbert.net/docs/pretrained_models.html#model-overview
[14] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[15] https://cloud.google.com/

Out-expansion Technique			ScienceWISE		MSH		CSWiki		SciAD		Average		
Technique Group	**Predictors**	**Repre-sentators**	**Acc**	**MaF1**	**Acc**	**MaF1**	**Acc**	**MaF1**	**Acc**	**MaF1**	**Acc**	**MaF1**	**Exec Times**
Classical	Random		47.72%	46.10%	47.04%	45.49%	14.54%	14.21%	33.06%	32.22%	35.59%	34.51%	**0.00**
	Most Frequent		70.52%	49.31%	50.30%	32.32%	47.76%	20.37%	69.08%	37.64%	59.41%	34.91%	**0.00**
	Cossim	CCV	91.34%	80.72%	97.62%	97.65%	77.96%	65.01%	92.28%	89.03%	89.80%	83.10%	0.07
		DCV	89.51%	78.69%	96.18%	96.07%	78.59%	65.86%	93.67%	87.08%	89.49%	81.92%	0.15
		SBE	88.07%	75.89%	95.84%	95.24%	74.60%	63.30%	86.50%	80.76%	86.25%	78.80%	0.05
	Thakker		87.77%	77.38%	92.53%	91.68%	73.16%	63.86%	84.36%	73.21%	84.46%	76.53%	3.79
Entity Disam.	LUKE		83.42%	57.33%	67.47%	58.95%	52.65%	46.60%	50.68%	42.53%	63.55%	51.35%	17.07
Sentence- -Oriented	UAD		43.69%	46.73%	93.92%	92.55%	12.94%	11.60%	34.98%	45.75%	46.38%	49.16%	0.01
	MadDog-out		89.13%	68.84%	94.09%	93.16%	57.03%	47.71%	87.38%	73.23%	81.91%	70.73%	0.37
	SciDr-out		88.22%	77.45%	97.23%	96.76%	**84.19%**	**72.67%**	**94.48%**	88.94%	91.03%	83.96%	1.28
	SciDr-out with External Data		89.89%	77.86%	97.58%	97.22%	N/A	N/A	**94.71%**	**89.42%**	N/A	N/A	N/A
Repre- sentator	Cossim	TF-IDF	91.26%	81.82%	97.62%	97.57%	77.80%	65.36%	91.79%	83.48%	89.62%	82.06%	2.53
		LDA	85.56%	73.94%	93.81%	93.28%	71.89%	60.49%	84.56%	73.39%	83.95%	75.28%	0.02
		Doc2Vec	92.86%	83.14%	98.33%	98.07%	77.16%	65.25%	92.05%	82.96%	90.10%	82.35%	0.10
		SBERT	**94.83%**	**85.32%**	**98.78%**	**98.80%**	81.47%	67.67%	**94.19%**	89.76%	**92.32%**	**85.39%**	0.30
Classi- fication	RF	TFIDF	70.82%	52.03%	84.53%	76.78%	32.11%	23.14%	87.64%	68.90%	68.77%	55.21%	23.79
		LDA	70.75%	54.13%	95.64%	92.84%	67.57%	50.78%	82.32%	61.33%	79.07%	64.77%	1.20
		Doc2Vec	79.18%	61.34%	96.58%	95.37%	66.29%	41.55%	84.39%	62.43%	81.61%	65.17%	1.36
	LR	TFIDF	71.05%	54.47%	93.41%	88.29%	71.89%	45.59%	80.63%	55.06%	79.24%	60.85%	11.84
		LDA	71.13%	51.49%	88.66%	80.02%	71.73%	48.77%	80.08%	55.16%	77.90%	58.86%	0.02
		Doc2Vec	88.83%	78.35%	**98.87%**	98.72%	76.68%	57.97%	90.75%	77.95%	88.78%	78.25%	0.11
	SVM	TFIDF	81.84%	62.13%	94.71%	91.27%	77.16%	53.54%	91.01%	78.24%	86.18%	71.29%	3.23
		LDA	78.88%	59.80%	93.64%	91.16%	71.89%	51.11%	85.59%	70.63%	82.50%	68.18%	0.02
		Doc2Vec	89.67%	79.31%	**98.93%**	98.79%	77.00%	58.70%	91.56%	80.88%	89.29%	79.42%	0.10
		SBERT	93.01%	83.91%	**98.87%**	**98.84%**	**82.43%**	64.44%	92.34%	86.53%	91.66%	83.43%	0.29
Combi- nation of Repre- sentators	Cossim	CCV + Doc2Vec	90.27%	79.04%	98.19%	97.95%	77.16%	65.25%	86.92%	82.82%	88.14%	81.27%	0.33
		DCV + Doc2Vec	90.27%	79.01%	98.33%	98.10%	77.16%	65.19%	92.05%	82.96%	89.45%	81.32%	1.65
	SVM	CCV + Doc2Vec	89.97%	80.44%	**98.95%**	**98.84%**	77.00%	58.70%	80.73%	76.06%	86.66%	78.51%	0.39
		DCV + Doc2Vec	89.67%	79.35%	**98.95%**	98.83%	77.00%	58.70%	90.20%	75.90%	88.95%	78.20%	1.68
Ensemblers	Hard		**94.15%**	**86.95%**	**99.60%**	**99.58%**	**84.19%**	78.32%	**96.59%**	**91.88%**	93.63%	**89.18%**	0.00
	Soft		93.62%	85.00%	99.38%	99.41%	**86.26%**	**79.33%**	95.94%	91.04%	**93.80%**	88.70%	0.00

Table 2: Out-expansion accuracy (Acc) and macro F1-measure (MaF1). Values marked as bold indicate the best Acc obtained by an individual technique and by an ensembler, respectively in that dataset. A technique T1 is considered better than T2 if a non-parametric significance test (based on shuffling[27]) indicates that the difference in their means has a p-value < 0.05. Thus, even though each column has a highest mean value for some technique H which will be bolded, the value of a technique T will also be bolded if H is no better than T based on the p-value criterion. We apply the same p-value criteria to bold ensemblers on all datasets, except on ScienceWISE where we apply the statistical test to each ensembler against Cossim SBERT (the best technique on ScienceWISE).

The best individual techniques (average above 89% of accuracy) in descending order are: Cossim with SBERT, SVM with SBERT, SciDr-out, Cossim with CCV, Cossim with TF-IDF, Cossim with DCV, Cossim with Doc2Vec alone or with DCV, and SVM with Doc2Vec. Regarding statistical significance, Cossim with SBERT is the best for SciWISE. For MSH, SVM with Doc2vec combined with either CCV or DCV score higher accuracy. However, they are not statistically significantly better than: SVM with either Doc2Vec or SBERT, Cossim with SBERT, and LR with Doc2Vec. SciDr-out achieves higher accuracy for CSWiki, but is not statistically better than SVM with SBERT. Finally, for SciAD, SciDr-out with external data scores higher accuracy but not statistically significantly better than: SciDr-out and Cossim with SBERT.

Interpretation: An important question in interpreting these numerical results is to understand why some techniques are better than others.

For out-expansion, the best approaches SciDr-out and Cossim/SVM with SBERT are based on language models trained on large data collections, but that does not tell the whole story. SciDr-out uses the particularly effective strategy of predicting the expansion span from the list of possible expansions passed as input. Further, SciDr-out is an ensemble of models trained in a 5-fold cross-validation setting. SBERT augments transformer language models to sentence similarity tasks using a siamese architecture that generates embeddings for each sentence and is trained to maximize similarity. Those embeddings turn out to be very informative regarding the context for documents: both Cossim or SVM combined with SBERT obtained on average the highest accuracy among individual techniques.

While LUKE's transformer language model enables the creation of entity embeddings, the results are not the best for acronyms, even with fine-tuning. One reason is that each entity is referenced frequently (over 600 times on the average [87]). Acronym/expansion pairs are referenced less than twice on the average.

Independently of which technique is best, we should note that each of the top techniques, except SciDr-out, gives a confidence score. For some of the best techniques SBERT, Doc2Vec, TFIDF, and CCV, the confidence score has a positive correlation with accuracy, though the correlation is modest (under 0.5). This low positive correlation is reflected in our results for ensemble techniques. The soft ensemble technique (in which each underlying technique's weight is monotonic with its confidence) does well thanks to the positive correlation. On the other hand, hard voting ensemble techniques (in which each underlying technique votes for its preferred expansion regardless of confidence) perform even better, suggesting that the "wisdom of crowds" effect is stronger than using confidences. A deeper look at ensemble techniques for acronym expansion is a subject for future work.

Representators and document processing execution times. The CCV and DCV representators take the least time (average **2s**) closely followed by TF-IDF (average **18s**). The most expensive models are SciDr-out (**14ks-66ks**) followed by LUKE (**1Ks-13ks**) and MadDog-out (**566s-10ks**) which use either language models or neural networks.

Among these best techniques, Cossim with CCV is the fastest for all datasets, able to process input documents in less than **0.07** seconds on dataset average. However, SVM with Doc2Vec is the fastest for MSH and SciWISE. The slowest among the best is Cossim with TF-IDF (average **2.5s**), followed by SciDr-out (**1.3s** for base and **2.3s** with external data). These differences are statistically significant.

The extended paper contains fine-grained execution time values per dataset, the correlation between confidence and accuracy, and further qualitative analysis[16].

In summary:

- If neither training time nor document processing time is of major concern and especially if GPU processing is available, then use either a **Hard** ensembler (best but slowest), **SciDr-out** (best with more domain data) or **Cossim/SVM** with **SBERT** (fastest and close to best).
- Otherwise, use **Cossim** with **CCV**, which requires almost no training time (less than 5s) and is the fastest in testing time among the best set of techniques.

6 End-to-end Benchmark and Evaluation

The end-to-end benchmark described in Section 6.1 is a set of documents together with human-annotated acronyms, whether those acronyms correspond to in-expansions or out-expansions.

[16]temporarily located at: web.tecnico.ulisboa.pt/ist164790/acx_extended.pdf

6.1 Benchmark Datasets, Algorithms, and Performane Metrics

6.1.1 Datasets The end-to-end benchmark uses (i) a *training* dataset consisting of documents from Wikipedia (ii) a *testing* dataset consisting of a disjoint set of Wikipedia articles (briefly described in Section 4.1.1). Those documents came from the Wikipedia dump of March 1, 2020[17]. These were converted to pure text using WikiExtractor [4].

We preprocessed all the documents using all the steps described in Section 5.1.2 for all out-expansion techniques except MadDog-out which uses its own preprocessing techniques.

6.1.2 End-to-end systems We use: (i) the end-to-end MadDog System (**MadDog-sys**) and (ii) various ***pipelines* of AcX** consisting of an in-expansion technique followed by an out-expansion technique possibly with machine learning (see Figure 1). An example of a pipeline would be the SH in-expander, Doc2Vec, and SVMs. The pipelines we test consist of combinations of the most practical (accurate and fastest) techniques for in-expansion and out-expansion as determined by the benchmarks in Sections 4.2 and 5.2. Specifically, AcX pipelines use either the **MadDog-in** or the **SH** technique as in-expanders to identify acronyms and expansions in input documents. For out-expansion, AcX pipelines include one of the following combinations of out-expansion techniques, i.e., a predictor (Section 3.3) with a representator (Section 3.2): (i) **Cossim** with **SBERT**; (ii) **SVM** with **SBERT**; (iii) **Cossim** with **CCV**; (iv) **SVM** with **Doc2vec**.

6.1.3 Performance Metrics Similarly to Section 4.1.3, we evaluate MadDog-sys, different pipelines of AcX, and human annotators listed in Section 6.1.2 in terms of Precision (**P**), Recall (**R**) and F1-Measure (**F1**). In contrast to Section 4.1.3, we evaluate all acronym-expansions pairs, whether they come from in-expansions or out-expansions.

We also measure training and per test document execution times.

6.2 Results on End-to-end Experiments

Setup. For these experiments, we used a virtual machine with the following specifications: AMD EPYC Processor with 16 cores and 256GB of RAM (Random Access Memory). For SBERT, the virtual machine specifications were: 5 cores of an Intel Xeon Gold 6126 Processor, 40GB of RAM and a NVIDIA GeForce RTX 2080 Ti.

Results. Table 3 presents the results for the AcX system running each one of the different pipelines mentioned in Section 6.1.2, the MadDog-sys[18], and the results for the student annotators. The AcX pipeline composed by MadDog-in, SVM with SBERT obtains the best results with precision (61.32%) and F1-measure (54.97%). However, based on the F1-measure, this is not statistically significantly better (i.e., P-value above 0.05) than SH and SVM with SBERT. The best system pipeline takes **2s** on average to process a document. Our best AcX pipeline obtains better results for all measures than the MadDog-sys (**+20%** of F1) and is faster (**2s** to **1084s**).

[17]https://dumps.wikimedia.org/enwiki

[18]https://archive.org/details/MadDog-models

AcX (pipelines)						
In-exp	**Out-exp Predictor**	**Repre-sentators**	**P**	**R**	**F1**	**Exec Times**
SH	Cossim	CCV	51.43%	45.62%	48.35%	21.31
		SBERT	53.10%	47.10%	49.92%	**0.15**
	SVM	SBERT	57.27%	**50.80%**	**53.84%**	2.37
Mad-Dog-in	Cossim	CCV	53.12%	43.15%	47.62%	17.45
		SBERT	55.17%	44.81%	49.46%	0.33
	SVM	Doc2Vec	59.12%	48.02%	53.00%	1.12
		SBERT	**61.32%**	**49.81%**	**54.97%**	2.29
MadDog-sys			37.85%	29.14%	32.93%	1084.92
Student annotators			88.36%	76.41%	81.95%	N/A

Table 3: End-to-end system quality metrics and average execution times to process a document in seconds. Values marked as bold indicate the best obtained in that metric. A technique T1 is considered better than T2 if a non-parametric significance test (based on shuffling[27]) indicates that the difference in their means has a p-value < 0.05. Thus, even though each column has a highest mean value for some technique H, the value of a technique T will be bolded if H is no better than T based on the p-value criterion.

Best AcX pipeline analysis. AcX precision is low, mostly because it incorrectly extracts words as acronyms (329 in total). Some are small words like "and" and "not" or codes like ZAB and ZAU (airports). Conversely, it fails to extract measurement units (e.g., m for meter, g for gram) and some common language abbreviations (e.g., Micro, "etc"). By contrast, AcX provides the correct expansion for the acronyms that newcomers to a field may not know, e.g., CAS - Computer Algebra System and SLS - SoftLanding Linux system.
Comparison with human performance. Compared with human annotators, our best AcX pipeline (MadDog-in and SVMs with SBERT) is around 27% lower in Precision, Recall, and F1-Measure. So, there is a lot of room for improvement. On the other hand, AcX is rapid (**2s**) and can help a newcomer in a field.
An example application of AcX. Consider one of the documents out of the 163 at random whose original page is here https://en.wikipedia.org/wiki/CC_(complexity). Our best AcX pipeline identified the following acronym-expansion pairs: CC - comparator circuits; CCVP - comparator circuit value problem; AC - alternating current; NC - nick's class; and NL - national league. However, it failed to identify CC-complete, and P. We can see that CC, CCVP, NC, and AL are correct and NL is incorrect. With a different Pipeline consisting of Doc2Vec instead of SBERT, AL is incorrect, but NL is correct.
In summary: The best AcX pipeline consists of **MadDog-in**, with **SVM** and **SBERT**.

7 Error Analysis

We studied how out-expansion errors for known expansions (i.e., expansions in documents of the training set) relate to the following properties: (i) the number of appearances of a particular acronym A, (ii) the length of acronym A, (iii) the fraction of appearances of a given expansion e of A and (iv) the total number of occurrences of expansion e for acronym A.

The data sources are the out-expansion and end-to-end benchmarks. For out-expansion, we also considered the dataset domain. We collect these results in a set of decision trees[19]. Each leaf of each decision tree holds the F1 score **value** for acronym-expansions having the properties indicated by the path to that leaf. Here is a summary of the patterns found in the decision trees:

- If the expansion e is very infrequent for A (below 2% of acronym occurrences) and the number of occurrences of A is low, the F1 score is low or very low (well under 0.2). There is, however, a boost of the F1 score for the SVM with SBERT technique when the acronym length is at least 3.
- When expansion e appears at least half of the time for acronym A, but acronym A occurs less than a dozen times, then the F1 score is decent (around 0.5).
- Finally, if the expansion count of e for acronym A is high and expansion e is a majority expansion for A, then F1 is very high (often more than 0.9).

Those patterns are generalizable to the best out-expansion techniques. The F1 score is largely independent of the dataset domain.

8 Conclusions and Future Work

The AcX system synthesizes and extends the best of previous work on acronym expansion. We have found:

- In-expansion rule-based techniques (SH and MadDog-in) usually work best and require little execution time.
- For out-expansion, SciDr-out and Cossim or SVMs with SBERT usually work best , followed by Cossim and SVMs with either CCV or Doc2Vec.

There are five data and software products of our work that future researchers can either extend or use as a basis of comparison.

(1) The first human-annotated dataset for end-to-end acronym expander systems.
(2) Three benchmarks to evaluate: (i) in-expansion techniques, (ii) out-expansion techniques, (iii) the combination in an end-to-end setting.
(3) The end-to-end AcX system is available publicly and can be applied to arbitrary languages, and can incorporate new in- and out-expansion techniques.

Future Work

Because the automated techniques in the state-of-the-art fall well below human-level accuracy levels, there is a large margin for improvement. Some promising avenues for improvements include: (i) more accurate in-expansion (e.g., additional acronym-expansion extraction patterns), (ii) new context representation techniques, and (iii) an extensive study of ensemble techniques.

With respect to the AcX system, we will add an Application Programming Interface (API) so text analytics systems (e.g., entity disambiguation or sentiment analysis) can benefit from acronym expansion. Finally, because our platform easily extends to other languages (e.g., our Portuguese extension was done by a high school student), we plan to create AcX pipelines, benchmarks, and perform end-to-end experiments for a variety of natural languages.

[19]Temporary location: https://amsuni-my.sharepoint.com/:f:/g/personal/j_p_pereira_uva_nl/EjzkFRe9POVHlKJvXIkWDmcBzqOf23_pb8L1gnSSYXcu2A?e=964CbX

Acknowledgments

Pereira's work was supported by national funds through FCT (*Fundação para a Ciência e a Tecnologia*), under the PhD Scholarship SFRH/BD/135719/2018. Furthermore, Pereira and Galhardas' work was supported by national funds through FCT under the project UIDB/50021/2020.

Shasha's work has been partly supported by (i) the New York University Abu Dhabi Center for Interacting Urban Networks (CITIES), funded by Tamkeen under the NYUAD Research Institute Award CG001 and by the Swiss Re Institute under the Quantum Cities initiative, (ii) NYU WIRELESS, (iii) U.S. National Science Foundation grants 1934388, 1840761, and 1339362, and (iv) INRIA.

The server virtual machine used to run the experiments was supported by BioData.pt – *Infraestrutura Portuguesa de Dados Biológicos*, project 22231/01/SAICT/2016, funded by Portugal 2020. This material is based upon work supported by Google Cloud. This work was carried out on the Dutch national e-infrastructure with the support of SURF Cooperative.

Finally, we would like to thank the reviewers for several excellent suggestions.

References

[1] ABBREX. 2011. ABBREX - The Abbreviation Expander. http://abbrex.com/

[2] Khaled Abdalgader and Andrew Skabar. 2012. Unsupervised Similarity-based Word Sense Disambiguation Using Context Vectors and Sentential Word Importance. *ACM Transactions on Speech and Language Processing* 9, 1 (2012), 2–21.

[3] Hiroko Ao and Toshihisa Takagi. 2005. ALICE: an algorithm to extract abbreviations from MEDLINE. *Journal of the American Medical Informatics Association* 12, 5 (2005), 576–586.

[4] Giuseppe Attardi. 2015. WikiExtractor. https://github.com/attardi/wikiextractor.

[5] S Azimi, H Veisi, and R Amouie. 2019. A method for automatic detection of acronyms in texts and building a dataset for acronym disambiguation. In *Iranian Conference on Signal Processing and Intelligent Systems*. 1–4.

[6] Adrian Barnett and Zoe Doubleday. 2020. Meta-Research: The growth of acronyms in the scientific literature. *eLife* 9 (jul 2020), e60080. https://doi.org/10.7554/eLife.60080

[7] Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: Pretrained Language Model for Scientific Text. In *Empirical Methods in Natural Language Processing*.

[8] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural Language Processing with Python* (1st ed.). O'Reilly Media, Inc.

[9] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3 (March 2003), 993–1022.

[10] Robert Bossy, Louise Deléger, Estelle Chaix, Mouhamadou Ba, and Claire Nédellec. 2019. Bacteria Biotope at BioNLP Open Shared Tasks 2019. In *Workshop on Biomedical Natural Language Processing Open Shared Tasks*. Association for Computational Linguistics, Hong Kong, China, 121–131. https://doi.org/10.18653/v1/D19-5719

[11] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32.

[12] Jean Charbonnier and Christian Wartena. 2018. Using Word Embeddings for Unsupervised Acronym Disambiguation. In *International Conference on Computational Linguistics*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2610–2619. https://www.aclweb.org/anthology/C18-1221

[13] Zheng Chen, Suzanne R Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew G Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. CUNY-BLENDER TAC-KBP2010 Entity Linking and Slot Filling System Description. *Theory and Applications of Categories* (2010).

[14] Daphné Chopard and Irena Spasić. 2019. A Deep Learning Approach to Self-expansion of Abbreviations Based on Morphology and Context Distance. In *Statistical Language and Speech Processing*. 71–82. https://doi.org/10.1007/978-3-030-31372-2_6

[15] Manuel R. Ciosici and Ira Assent. 2018. Abbreviation Expander - a Web-based System for Easy Reading of Technical Documents. In *Computational Linguistics: System Demonstrations*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 1–4. https://www.aclweb.org/anthology/C18-2001

[16] Manuel R. Ciosici, Tobias Sommer, and Ira Assent. 2019. Unsupervised Abbreviation Disambiguation Contextual disambiguation using word embeddings. *Computing Research Repository* arXiv:1904.00929 (2019). arXiv:1904.00929 http://arxiv.org/abs/1904.00929 version 2.

[17] Nigel Collier and Jin-Dong Kim. 2004. Introduction to the Bio-entity Recognition Task at JNLPBA. In *International Joint Workshop on Natural Language Processing in Biomedicine and its Applications*. Geneva, Switzerland, 73–78. https://aclanthology.org/W04-1213

[18] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Computing Research Repository* arXiv:1810.04805 (2018). https://arxiv.org/abs/1810.04805

[20] Nicholas Egan and John Bohannon. 2021. Primer AI's Systems for Acronym Identification and Disambiguation. In *Workshop on Scientific Document Understanding co-located with 35th AAAI Conference on Artificial Inteligence*. CEUR-WS.org. http://ceur-ws.org/Vol-2831/paper30.pdf

[21] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A Library for Large Linear Classification. *Journal of Machine Learning Research* 9 (jun 2008), 1871–1874.

[22] Shicong Feng, Yuhong Xiong, Conglei Yao, Liwei Zheng, and Wei Liu. 2009. Acronym Extraction and Disambiguation in Large-Scale Organizational Web Pages. In *Conference on Information and Knowledge Management* (Hong Kong, China). Association for Computing Machinery, New York, NY, USA, 1693–1696. https://doi.org/10.1145/1645953.1646206

[23] Nicholas FitzGerald, Dan Bikel, Jan Botha, Daniel Gillick, Tom Kwiatkowski, and Andrew McCallum. 2021. MOLEMAN: Mention-Only Linking of Entities with a Mention Annotation Network. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, Online, 278–285. https://doi.org/10.18653/v1/2021.acl-short.37

[24] Michael R. Glass, Md. Faisal Mahbub Chowdhury, and Alfio Massimiliano Gliozzo. 2017. Language Independent Acquisition of Abbreviations. *Computing Research Repository* arXiv:1709.08074 (2017). arXiv:1709.08074 http://arxiv.org/abs/1709.08074 version 1.

[25] Aitor Gonzalez-Agirre, Montserrat Marimon, Ander Intxaurrondo, Obdulia Rabal, Marta Villegas, and Martin Krallinger. 2019. PharmaCoNER: Pharmacological Substances, Compounds and proteins Named Entity Recognition track. In *Workshop on Biomedical Natural Language Processing Open Shared Tasks*. Association for Computational Linguistics, Hong Kong, China, 1–10. https://doi.org/10.18653/v1/d19-5701

[26] Phil Gooch. 2012. BADREX: In situ expansion and coreference of biomedical abbreviations using dynamic regular expressions. *Computing Research Repository* arXiv:1206.4522 (2012). arXiv:1206.4522 http://arxiv.org/abs/1206.4522 version 1.

[27] Phillip I Good. 2006. *Resampling Methods: A Practical Guide to Data Analysis*. Birkhäuser Basel. https://doi.org/10.1007/0-8176-4444-X

[28] Richard D Hipp. 2020. SQLite. https://www.sqlite.org/

[29] Rezarta Islamaj Doğan, Donald C Comeau, Lana Yeganova, and W John Wilbur. 2014. Finding abbreviations in biomedical literature: three BioC-compatible modules and four BioC-formatted corpora. *Database: the journal of biological databases and curation* 2014 (2014).

[30] Kayla Jacobs, Alon Itai, and Shuly Wintner. 2020. Acronyms: identification, expansion and disambiguation. *Annals of Mathematics and Artificial Intelligence* 88, 5 (2020), 517–532.

[31] A Jain, S Cucerzan, and Saliha Azzam. 2007. Acronym-Expansion Recognition and Ranking on the Web. *IEEE International Conference on Information Reuse and Integration* (2007), 209–214.

[32] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. 2014. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated.

[33] Antonio J Jimeno-Yepes, Bridget T McInnes, and Alan R Aronson. 2011. Exploiting MeSH indexing in MEDLINE to generate a data set for word sense disambiguation. *BMC Bioinformatics* 12, 1 (2011), 1–14.

[34] Karen Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation* 28 (1972), 11–21.

[35] İlknur Karadeniz, Ömer Faruk Tuna, and Arzucan Özgür. 2019. BOUN-ISIK Participation: An Unsupervised Approach for the Named Entity Normalization and Relation Extraction of Bacteria Biotopes. In *Workshop on Biomedical Natural Language Processing Open Shared Tasks*. Association for Computational Linguistics, Hong Kong, China, 150–157. https://doi.org/10.18653/v1/D19-5722

[36] Klaus Krippendorff. 2018. *Content analysis: An introduction to its methodology*. Sage publications.

[37] Cheng-Ju Kuo, Maurice HT Ling, Woody Lin, and Chun-Nan Hsu. 2009. BIOADI: A machine learning approach to identifying abbreviations and definitions in biological literature. *BMC bioinformatics* 10 Suppl 15 (12 2009), S7.

[38] Matt J Kusner, Yu Sun, Nicholas I Kolkin, and Kilian Q Weinberger. 2015. From Word Embeddings to Document Distances. In *International Conference on International Conference on Machine Learning*. JMLR.org, 957–966.

[39] John D Lafferty, Andrew McCallum, and Fernando C N Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *International Conference on Machine Learning*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 282–289. http://dl.acm.org/citation.cfm?id=645530.655813

[40] Quoc V. Le and Tomas Mikolov. 2014. Distributed Representations of Sentences and Documents. In *International Conference on Machine Learning* (Beijing, China), Vol. 32. 1188–1196.

[41] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. 2017. End-to-end Neural Coreference Resolution. In *Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 188–197. https://doi.org/10.18653/v1/D17-1018

[42] Chao Li, Lei Ji, and Jun Yan. 2015. Acronym Disambiguation Using Word Embedding. In *AAAI Conference on Artificial Intelligence* (Austin, Texas). 4178–4179.

[43] Yang Li, Bo Zhao, Ariel Fuxman, and Fangbo Tao. 2018. Guess Me if You Can: Acronym Disambiguation for Enterprises. In *Annual Meeting of the Association for Computational Linguistics*, Vol. 1: Long Papers. Association for Computational Linguistics, Melbourne, Australia, 1308–1317. https://doi.org/10.18653/v1/P18-1121

[44] Nicholas B Link, Sicong Huang, Tianrun Cai, Jiehuan Sun, Kumar Dahal, Lauren Costa, Kelly Cho, Katherine Liao, Tianxi Cai, and Chuan Hong. 2022. Binary acronym disambiguation in clinical notes from electronic health records with an application in computational phenotyping. *International Journal of Medical Informatics* 162 (2022), 104753. https://doi.org/10.1016/j.ijmedinf.2022.104753

[45] Jie Liu, Caihua Liu, and Yalou Huang. 2017. Multi-granularity sequence labeling model for acronym expansion identification. *Information Sciences* 378 (2017), 462 – 474.

[46] Robert L Logan IV, Andrew McCallum, Sameer Singh, and Dan Bikel. 2021. Benchmarking Scalable Methods for Streaming Cross Document Entity Coreference. In *Annual Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, Online, 4717–4731. https://doi.org/10.18653/v1/2021.acl-long.364

[47] Pengcheng Lu and Massimo Poesio. 2021. Coreference Resolution for the Biomedical Domain: A Survey. In *Workshop on Computational Models of Reference, Anaphora and Coreference*. https://doi.org/10.48550/ARXIV.2109.12424

[48] Xinyin Ma, Yong Jiang, Nguyen Bach, Tao Wang, Zhongqiang Huang, Fei Huang, and Weiming Lu. 2021. MuVER: Improving First-Stage Entity Retrieval with Multi-View Entity Representations. In *Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic, 2617–2624. https://doi.org/10.18653/v1/2021.emnlp-main.205

[49] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *Computing Research Repository* arXiv:1301.3781 (2013). https://arxiv.org/abs/1301.3781 version 3.

[50] Sungrim Moon, Bridget McInnes, and Genevieve B Melton. 2015. Challenges and practical approaches with word sense disambiguation of acronyms and abbreviations in the clinical domain. *Healthcare Informatics Research* 21, 1 (jan 2015), 35–42. https://doi.org/10.4258/hir.2015.21.1.35

[51] Sungrim Moon, Serguei Pakhomov, and Genevieve B Melton. 2012. Automated disambiguation of acronyms and abbreviations in clinical texts: window and training size considerations. *AMIA Annual Symposium proceedings* 2012 (2012), 1310–1319.

[52] Andrea Moro and Roberto Navigli. 2015. SemEval-2015 Task 13: Multilingual All-Words Sense Disambiguation and Entity Linking. In *International Workshop on Semantic Evaluation*. Association for Computational Linguistics, Denver, Colorado, 288–297. https://doi.org/10.18653/v1/S15-2049

[53] Andrea Moro, Alessandro Raganato, and Roberto Navigli. 2014. Entity Linking meets Word Sense Disambiguation: a Unified Approach. *Transactions of the Association for Computational Linguistics* 2 (2014), 231–244. https://doi.org/10.1162/tacl_a_00179

[54] Roberto Navigli. 2009. Word Sense Disambiguation: A Survey. *Comput. Surveys* 41, 2, Article 10 (Feb. 2009), 69 pages. https://doi.org/10.1145/1459352.1459355

[55] Vincent Ng. 2017. Machine Learning for Entity Coreference Resolution: A Retrospective Look at Two Decades of Research. In *AAAI Conference on Artificial Intelligence*, Vol. 31. https://ojs.aaai.org/index.php/AAAI/article/view/11149

[56] Sergeui Pakhomov, Ted Pedersen, and Christopher G Chute. 2005. Abbreviation and acronym disambiguation in clinical discourse. *AMIA Annual Symposium proceedings* 2005 (2005), 589–593.

[57] Youngja Park and Roy J. Byrd. 2001. Hybrid Text Mining for Finding Abbreviations and their Definitions. In *Empirical Methods in Natural Language Processing*. https://www.aclweb.org/anthology/W01-0516

[58] Eleni Partalidou, Despina Christou, and Grigorios Tsoumakas. 2021. Improving Zero-Shot Entity Retrieval through Effective Dense Representations. *Computing Research Repository* arXiv:2103.04156 (2021). https://arxiv.org/abs/2103.04156

[59] Rebecca Passonneau. 2006. Measuring Agreement on Set-valued Items (MASI) for Semantic and Pragmatic Annotation. In *International Conference on Language Resources and Evaluation*. European Language Resources Association (ELRA), Genoa, Italy.

[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[61] Anja Pilz and Gerhard Paaß. 2011. From Names to Entities Using Thematic Context Distance. In *Conference on Information and Knowledge Management*. Association for Computing Machinery, New York, NY, USA, 857–866. https://doi.org/10.1145/2063576.2063700

[62] Roman Prokofyev, Gianluca Demartini, Alexey Boyarsky, Oleg Ruchayskiy, and Philippe Cudré-Mauroux. 2013. Ontology-Based Word Sense Disambiguation for Scientific Literature. In *European Conference on Advances in Information Retrieval* (Moscow, Russia). Springer-Verlag, Berlin, Heidelberg, 594–605. https://doi.org/10.1007/978-3-642-36973-5_50

[63] J Pustejovsky, J Castaño, B Cochran, M Kotecki, and M Morrell. 2001. Automatic extraction of acronym-meaning pairs from MEDLINE databases. *Studies in Health Technology and Informatics* 84, Pt 1 (2001), 371–375.

[64] Alessandro Raganato, Jose Camacho-Collados, and Roberto Navigli. 2017. Word Sense Disambiguation: A Unified Evaluation Framework and Empirical Comparison. In *Conference of the European Chapter of the Association for Computational Linguistics s*. Association for Computational Linguistics, Valencia, Spain, 99–110. https://www.aclweb.org/anthology/E17-1010

[65] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Hong Kong, China, 3982–3992. https://doi.org/10.18653/v1/D19-1410

[66] Saneesh Mohammed N and K A Abdul Nazeer. 2013. An improved method for extracting acronym-definition pairs from biomedical Literature. In *2013 International Conference on Control Communication and Computing (ICCC)*. 194–197.

[67] Ariel S Schwartz and Marti A Hearst. 2003. A simple algorithm for identifying abbreviation definitions in biomedical text. In *Pacific Symposium on Biocomputing*. 451–462.

[68] Özge Sevgili, Alexander Panchenko, and Chris Biemann. 2019. Improving Neural Entity Disambiguation with Graph Embeddings. In *Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*. Association for Computational Linguistics, Florence, Italy, 315–322. https://doi.org/10.18653/v1/P19-2044

[69] Wei Shen, Yuhan Li, Yinan Liu, Jiawei Han, Jianyong Wang, and Xiaojie Yuan. 2021. Entity Linking Meets Deep Learning: Techniques and Solutions. *IEEE Transactions on Knowledge and Data Engineering* (2021). https://doi.org/10.1109/TKDE.2021.3117715

[70] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Transactions on Knowledge and Data Engineering* 27, 2 (2015), 443–460. https://doi.org/10.1109/TKDE.2014.2327028

[71] Utpal Kumar Sikdar and Björn Gambäck. 2017. A Feature-based Ensemble Approach to Recognition of Emerging and Rare Named Entities. In *Workshop on Noisy User-generated Text*. Association for Computational Linguistics, Copenhagen, Denmark, 177–181. https://doi.org/10.18653/v1/W17-4424

[72] Aadarsh Singh and Priyanshu Kumar. 2021. SciDr at SDU-2020: IDEAS–Identifying and Disambiguating Everyday Acronyms for Scientific Domain. In *Workshop on Scientific Document Understanding co-located with 35th AAAI Conference on Artificial Inteligence*. CEUR-WS.org. http://ceur-ws.org/Vol-2831/paper31.pdf

[73] Sunghwan Sohn, Donald C Comeau, Won Kim, and W John Wilbur. 2008. Abbreviation definition identification based on automatic precision estimates. *BMC bioinformatics* 9, 1 (2008), 402.

[74] Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. 2020. MPNet: Masked and Permuted Pre-training for Language Understanding. *Advances in Neural Information Processing Systems* 33 (2020), 16857–16867.

[75] Mark Stevenson, Yikun Guo, Abdulaziz Al Amri, and Robert Gaizauskas. 2009. Disambiguation of Biomedical Abbreviations. In *Workshop on Current Trends in Biomedical Natural Language Processing*. Association for Computational Linguistics, USA, 71–79.

[76] Bilyana Taneva, Tao Cheng, Kaushik Chakrabarti, and Yeye He. 2013. Mining Acronym Expansions and Their Meanings Using Query Click Log. In *International Conference on World Wide Web*. Association for Computing Machinery, New York, NY, USA, 1261–1272. https://doi.org/10.1145/2488388.2488498

[77] Aditya Thakker, Suhail Barot, and Sudhir Bagul. 2017. Acronym Disambiguation: A Domain Independent Approach. *Computing Research Repository* arXiv:1711.09271 (2017). https://arxiv.org/abs/1711.09271 version 3.

[78] Amir Pouran Ben Veyseh, Franck Dernoncourt, Walter Chang, and Thien Huu Nguyen. 2021. MadDog: A Web-based System for Acronym Identification and Disambiguation. In *European Chapter of the Association for Computational Linguistics*.

[79] Amir Pouran Ben Veyseh, Franck Dernoncourt, Thien Huu Nguyen, Walter Chang, and Leo Anthony Celi. 2021. Acronym Identification and Disambiguation Shared Tasks for Scientific Document Understanding. In *Workshop on Scientific Document Understanding co-located with 35th AAAI Conference on Artificial Inteligence*. CEUR-WS.org. http://ceur-ws.org/Vol-2831/paper33.pdf

[80] Amir Pouran Ben Veyseh, Franck Dernoncourt, Quan Hung Tran, and Thien Huu Nguyen. 2020. What Does This Acronym Mean? Introducing a New Dataset for Acronym Identification and Disambiguation. In *International Conference on Computational Linguistics*.

[81] Yogarshi Vyas and Miguel Ballesteros. 2021. Linking Entities to Unseen Knowledge Bases with Arbitrary Schemas. In *North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Online, 834–844. https://doi.org/10.18653/v1/2021.naacl-main.65

[82] Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable Zero-shot Entity Linking with Dense Entity Retrieval. In *Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online, 6397–6407. https://doi.org/10.18653/v1/2020.emnlp-main.519

[83] Yonghui Wu, Joshua C Denny, S Trent Rosenbloom, Randolph A Miller, Dario A Giuse, Lulu Wang, Carmelo Blanquicett, Ergin Soysal, Jun Xu, and Hua Xu. 2017. A long journey to short abbreviations: developing an open-source framework for clinical abbreviation recognition and disambiguation (CARD). *Journal of the American Medical Informatics Association* 24 (2017), 79–86. https://doi.org/10.1093/jamia/ocw109

[84] Yonghui Wu, Jun Xu, Yaoyun Zhang, and Hua Xu. 2015. Clinical Abbreviation Disambiguation Using Neural Word Embeddings. In *Workshop on Biomedical Natural Language Processing*. Association for Computational Linguistics, Beijing, China, 171–176. https://doi.org/10.18653/v1/W15-3822

[85] Vikas Yadav and Steven Bethard. 2018. A Survey on Recent Advances in Named Entity Recognition from Deep Learning models. In *International Conference on Computational Linguistics*. Association for Computational Linguistics, Santa Fe, New Mexico, USA, 2145–2158. https://aclanthology.org/C18-1182

[86] Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. 2020. LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention. In *Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Online, 6442–6454. https://doi.org/10.18653/v1/2020.emnlp-main.523

[87] Ikuya Yamada, Koki Washio, Hiroyuki Shindo, and Yuji Matsumoto. 2019. Global Entity Disambiguation with BERT. *Computing Research Repository* arXiv:1909.00426 (2019). https://arxiv.org/abs/1909.00426 version 3.

[88] Zonghai Yao, Liangliang Cao, and Huapu Pan. 2020. Zero-shot Entity Linking with Efficient Long Range Sequence Modeling. In *Findings of the Association for Computational Linguistics: Empirical Methods in Natural Language Processing*. 2517–2522.

[89] Anna Yarygina and Natalia Vassilieva. 2012. High-recall Extraction of Acronym-definition Pairs with Relevance Feedback. In *Joint Extending Database Technology and International Conference on Database Theory Workshops*. ACM, New York, NY, USA, 21–28. https://doi.org/10.1145/2320765.2320781

[90] Hong Yu, Won Kim, Vasileios Hatzivassiloglou, and John Wilbur. 2006. A Large Scale, Corpus-Based Approach for Automatically Disambiguating Biomedical Abbreviations. *ACM Transactions on Information Systems* 24, 3 (jul 2006), 380–404. https://doi.org/10.1145/1165774.1165778

[91] Danqing Zhu, Wangli Lin, Yang Zhang, Qiwei Zhong, Guanxiong Zeng, Weilin Wu, and Jiayu Tang. 2021. AT-BERT: Adversarial Training BERT for Acronym Identification Winning Solution for SDU@ AAAI-21. In *Workshop on Scientific Document Understanding co-located with 35th AAAI Conference on Artificial Inteligence*. CEUR-WS.org. http://ceur-ws.org/Vol-2831/paper28.pdf

[92] Stefan Zwicklbauer, Christin Seifert, and Michael Granitzer. 2016. Robust and collective entity disambiguation through semantic embeddings. In *Special Interest Group in Information Retrieval*. Association for Computing Machinery, New York, NY, USA, 425–434. https://doi.org/10.1145/2911451.2911535