PLOD: An Abbreviation Detection Dataset for Scientific Documents

Leonardo Zilio[1], Hadeel Saadany[1], Prashant Sharma[2],
Diptesh Kanojia[1,3], Constantin Orăsan[1,3]
[1]Centre for Translation Studies, University of Surrey, United Kingdom.
[2]Hitachi CRL, Japan.
[3]Surrey Institute for People-centred AI, University of Surrey, United Kingdom.
[1,3]{l.zilio, h.saadany, d.kanojia, c.orasan}@surrey.ac.uk, [2]prashaantsharmaa@gmail.com

Abstract

The detection and extraction of abbreviations from unstructured texts can help to improve the performance of Natural Language Processing tasks, such as machine translation and information retrieval. However, in terms of publicly available datasets, there is not enough data for training deep-neural-networks-based models to the point of generalising well over data. This paper presents PLOD, a large-scale dataset for abbreviation detection and extraction that contains 160k+ segments automatically annotated with abbreviations and their long forms. We performed manual validation over a set of instances and a complete automatic validation for this dataset. We then used it to generate several baseline models for detecting abbreviations and long forms. The best models achieved an F1-score of 0.92 for abbreviations and 0.89 for detecting their corresponding long forms. We release this dataset along with our code and all the models publicly in this Github Repository.

Keywords: abbreviation detection, PLOS journals, abbreviation dataset

1. Introduction

A pervasive characteristic of scientific reports and research papers is their frequent use of abbreviations (Wu et al., 2011). For submitting to various journals, authors are also required to submit an abbreviation glossary, *i.e.*, a list of short forms with their expanded long forms. Such a glossary is essential for the reader to understand the domain-specific terminology used in the reported work.

From a linguistic point of view, there are different typologies of abbreviations, and often authors disagree in relation to a common classification system (Fabijanić, 2015). As Tchiotashvili et al. (2021) explain, there are authors advocating for the distinction between initialisms and acronyms[1], and others that defend a separation between shortening abbreviations and initial abbreviations. Considering that a typology of abbreviations is not our focus in this paper, we will use the terms "short form", "abbreviation" and "abbreviated token" interchangeably and as umbrella terms to denote any token(s) that can be expanded into a longer token or into a sequence of tokens that corresponds to its long form. As such, unless explicitly stated, we will not differentiate between, for instance, abbreviations and acronyms, and nor does our dataset.

From a Natural Language Processing (NLP) point of view, abbreviations are problematic for automatic processing, and the presence of short forms might hinder the machine processing of unstructured text. For example, a machine translation system may not provide a suitable translation for such tokens. Abbreviated tokens can pose a problem for almost any NLP system, because they often contain important information, such as names of diseases, drug names, or common procedures which must be recognisable in the translated document. The performance of information retrieval can also be affected in terms of both precision and recall due to incorrect abbreviation expansion (Toole, 2000). Therefore, the detection and extraction of accurate abbreviated tokens and their corresponding long forms is an important task that can significantly impact NLP systems' output.

Any NLP system which attempts to extract such information from unstructured text faces several challenges, because abbreviations:

- are domain-specific (*e.g.*, BMI can mean "Body Mass Index" or "Bilinear Matrix Inequalities", depending on the domain or the context);

- often have ambiguous connotations (*e.g.*, CI can mean "conditional independence", "confidence interval", or "compound interest"; all from the same domain);

- can often contain sub-abbreviations which are not fully expanded in the immediate context (*e.g.*, NMT might only contain the expansion "Neural MT", where MT can be found expanded earlier in the document);

- can have multiple letters that are a part of the same

---

[1]According to those authors, initialisms and acronyms are both short forms that are created using the initial letter(s) from a sequence of tokens, and are differentiated according to how they are pronounced.

word (*e.g.*, subsequence kernel (SSK) or maximum entropy (MaxEnt)); and

- can often appear in a text unaccompanied by their respective long forms.

The challenges discussed here also show that rule-based approaches will fail to perform well at this task as they will try to generalise over a pattern or a regular expression to detect abbreviations from a text. Multiple outliers cannot be detected with the help of such approaches. Therefore, it is important to create robust NLP systems that can detect and extract abbreviations with their corresponding long forms. The detection of short and long forms can help automate glossary generation for researchers. Similarly, it can help the expansion of the abbreviations in a free text, thus enabling downstream NLP tasks like Machine Translation (MT) or Information Retrieval (IR) to perform better.

In this paper, we describe our efforts to collect a large dataset of abbreviations from online open-source journals and create baseline NLP systems for the task of abbreviation detection. We present our new PLOD dataset for abbreviation detection and provide a detailed analysis of its main features. We also describe steps for dataset crawling and cleaning, and for its manual and automatic validation. With the help of various publicly available language models, we perform fine-tuning to create baseline models for abbreviation detection based on the PLOD dataset. We also test our baseline models performance on another publicly available acronym-extraction dataset to show their efficacy on this sister task. Our contributions with this paper are summarised below:

- We present PLOD, a large dataset for the detection and extraction of short and long forms.

- We provide several pre-trained baseline models that are readily available to use.

The rest of this paper is organised as follows: in Section 2 we discuss existing efforts for the extraction of abbreviations; Section 3 describes the methodology applied for the creation and validation of the new dataset and also presents some of its main statistics at the end; Section 4 contains an extrinsic evaluation of the resource, where we created several baseline models to test the dataset for automatically detecting short and long forms; in Section 5 we detail the results of each baseline model; finally, Section 6 briefly summarises what was achieved with this research and presents an overview of future steps.

2. Related Work

For many years, researchers have employed machine-learning-based methods to detect abbreviations from generic English texts. Toole (2000) introduces a hybrid two-stage approach for the identification and expansion of abbreviations based on a dataset from the Air Safety Reporting System (ASRS) database. The author proposes various features and utilises a binary decision tree to model the characteristics of an abbreviation. Similarly, Vanopstal et al. (2010) use Support Vector Machine (SVM) to classify abbreviations based on various features and on a dataset in the medical domain. Abbreviation detection has been more popular in the clinical and the medical domains as a lot of unstructured free text is prevalent in these areas which also contains multiple abbreviations. Xu et al. (2009) also propose a decision-tree-based approach for the classification of abbreviations from clinical narratives. Another research paper (Wu et al., 2011) shows a more exhaustive comparison of various machine-learning-based methods like decision tree, random forests, and SVM and utilise over 70 patient discharge summaries to perform the task. Kreuzthaler et al. (2016) use an unsupervised learning approach to detect abbreviations in clinical narratives, and show a decent performance on a small German language dataset (1696 samples). More recently, the CLEF shared task for short form normalisation propelled the efforts in this area (Wu et al., 2013). Another recent approach to detect abbreviations in clinical text utilises a semi-supervised learning approach to do the task (Vo and Cao, 2019) for a clinical text dataset (Moon et al., 2014).

The more specific task of acronym extraction, where the focus lies on abbreviations formed by initial letter(s), has also been of interest to the NLP community and has been performed for different domains in English. Early approaches for this task were primarily rule-based (Taghva and Gilbreth, 1999; Yeates, 1999; Park and Byrd, 2001; Larkey et al., 2000; Schwartz and Hearst, 2002), but there are instances of machine learning being used for the task (Nadeau and Turney, 2005; Kuo et al., 2009).

Recently, various deep learning-based approaches have been used for acronym extraction (Rogers et al., 2021; Li et al., 2021). Similarly, Zhu et al. (2021; Kubal and Nagvenkar (2021) use the fine-tuning approaches based on the recent transformer-based architectures. Ehrmann et al. (2013) show how acronym recognition patterns initially developed for medical terms can be adapted to the more general news domain. Their efforts led to automatically merging long-form variants referring to the same short form, while maintaining non-related long forms separately. Their work is based on the algorithm developed by Schwartz and Hearst (2002), but they perform the task of acronym extraction for 22 languages.

In fact, the acronym extraction and disambiguation shared task (Veyseh et al., 2022b) has encouraged more participants in the area while also releasing a large-scale dataset for multilingual and multi-domain acronym extraction (Veyseh et al., 2022a). However, none of the abbreviation datasets discussed can be considered significantly large for deep-learning-based approaches to generalise well enough and show decent task performance. With this work, we release a much

larger dataset containing tagged abbreviations and their corresponding long forms. Although we do not make an explicit distinction, our dataset does contain, among the abbreviations, acronyms, so that the models that we present here can also help in the more specific task of acronym extraction. With the help of fine-tuning, our evaluation also shows that this dataset can help extract abbreviations and acronyms with a decent performance.

Journal	Publication Period	Number of Files
PLOS Biology	2003-present	6,072
PLOS Medicine	2004-present	4,494
PLOS Computational Biology	2005-present	8,473
PLOS Genetics	2005-present	9,251
PLOS Pathogens	2005-present	9,148
PLOS Clinical Trials*	2006-2007	68
PLOS ONE	2006-present	257,854
PLOS Neglected Tropical Diseases	2007-present	9,388
PLOS Currents	2009-2018	697

*Later merged with PLOS ONE.

Table 1: PLOS Journals publication period and number of files.

3. Proposed Resource

In this section, we discuss the new PLOD dataset that we built from research articles published in PLOS Journals[2]. We first describe the corpus that was used and the methodology for collecting data from the journals. We then describe some automatic and manual methods applied for cleaning and validating the dataset. We conclude the section with statistics of the resource we developed.

3.1. Dataset Description and Creation

The PLOD dataset was extracted from open access articles published in PLOS journals. The articles from these journals are freely distributed along with the PMC Open Access Subset[3] and can be downloaded from their FTP server[4]. The corpus contains several journals, mostly from the Biomedical domain, since 2003. All articles from these journals are written in English. Table 1 presents the areas, publication time span and number of files that we have for each PLOS journal in the corpus[5].

[2]https://plos.org/.
[3]https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/.
[4]https://ftp.ncbi.nlm.nih.gov/pub/pmc/oa_bulk/.
[5]The data used in this research was downloaded on 16 October 2021.

The full corpus contains 305,445 files (31GB) divided into several types of articles. We used only the main category, Research Articles, which accounts for 283,874 files. All articles are presented in XML format, and most of them contain a section called "Abbreviations". We used Python's BeautifulSoup package to process the XML files and, for each file, extracted short and long forms from the "Abbreviations" section and then processed all $< p >$ tags from the XML structure. The textual content inside $< p >$ tags was split into sentences using a simple regular expression, for the sake of brevity, and then each sentence was analysed to match for the occurrence of any short forms present in the paper's glossary. Where an abbreviation was found, we also looked for its long form in the same segment/textual extract. During this annotation process, we identified the indexes of the beginning and ending character in the segment. In the search for abbreviations, we used the exact same case format as it appeared in the "Abbreviations" section of the article, using word boundaries as delimitation (*e.g.*, punctuation, brackets, spaces, apostrophes). However, for the long forms, we converted both the textual extract and the long form to lower case. We also allowed for certain plural forms in the abbreviations and in the last token of the long form (*e.g.*, addition of lower cased "s"), but these were not included in the annotation process (*i.e.*, the extracted indexes and forms do not reflect these extra characters).

This resulted in a huge collection of 1,348,357 segments with annotated abbreviations. Although most papers have a glossary with abbreviations and long forms, we only found 13,883 articles in which long forms appeared together with abbreviations. This means that all of the 1.3M+ extracted segments have abbreviations, but not all of them contain long forms. When we filtered the textual extracts that had at least one long form corresponding to one of the abbreviations present in the segment, we ended up with 162,658 segments. Considering only these textual extracts that contain both short and long forms, we had a vocabulary of 56,810 unique combinations of abbreviations with their corresponding long form. This dataset was then further validated and cleaned, as we discuss in the next subsection.

3.2. Dataset Cleaning and Validation

Since the process of collecting abbreviations and long forms, and annotating them in textual extracts was done automatically, we did some manual checks to validate the data. In this section we describe this process of validation, elaborating on the different steps that were taken to improve the quality of our released dataset.

One of the first steps that we took was to go through 500 random examples to check overall issues with the data. During this process, we identified **two main issues**: **one-character abbreviations** were resulting in several lines where, even though both the abbreviation and long form were present in the segment, they were

Journal	Number of Segments	Annotated Abbreviations	Annotated Long Forms
PLOS Biology	50975	165099	97002
PLOS Medicine	33036	83549	54237
PLOS Computational Biology	2124	4380	2540
PLOS Genetics	2740	5659	3152
PLOS Pathogens	2394	6225	2814
PLOS Clinical Trials	325	709	410
PLOS ONE	69217	183358	106031
PLOS Neglected Tropical Diseases	121	287	165
Total	160932	449266	266351

Table 2: Number of annotated segments, abbreviations and long forms per journal in the PLOD dataset.

not connected as a textual co-reference (see Example 1); and there were **missing annotations** of abbreviations and/or long forms, either because they were not coded in the "Abbreviations" section of the article or because they were written differently in the text (see Example 1).

- Example 1: The reaction of an **oligonucleotide substrate** bearing a **S** *P*-phosphorothioate at the cleavage site (*SSp*, Table 1) also experiences Cd2+ stimulation with the **WT** ribozyme.

In Example 1, it is possible to see that although the abbreviated token **S** and its long form **oligonucleotide substrate** are present in the segment, they are not being used as co-referents in this particular textual extract. We also see in the same example that SSp and P were not identified as abbreviations, because they were not present in the article's "Abbreviations" section.

To solve the first issue, after further investigation of other similar cases, we decided to filter one-character abbreviations out of the dataset, as they were indeed a source of many issues. This resulted in the removal of 705 unique long forms from the dataset, totalling 3,877 occurrences across 1,698 segments (the filtered dataset at this point has 160,969 segments). As it will be explained in Sections 4 and 5, we conducted our experiments using both PLOD_{Unfiltered Dataset} and PLOD_{Filtered Dataset} (which does not have any annotated one-character abbreviations). As for the second issue, where there is missing annotation, we decided not to act upon it, and we accepted that there will be some segments where some of the abbreviations (with or without their long forms) and/or long forms are not identified.

In a second step of validation, we used spaCy[6] to create a language model specific for the annotated long forms. In this language model, any token that was not a stop-word was replaced by either a placeholder for punctuation or for content word[7]. This step reduced the amount

of different long forms to 3,592, which allowed us to identify some oddly formed sequences, and also long forms that were too long, and possibly wrong. Based on this analysis, we annotated each segment with a number indicating whether it contains a long form that either begins or ends with stop-words or is very long (*i.e.*, longer than 12 tokens), or both. This generated an extra annotation on 5,671 segments, which were not automatically excluded from the dataset, but are readily identifiable because of this annotation. In terms of the long forms that were very long, we did perform a validation among all long forms that had more than 12 tokens, and, among the 344 unique combinations of abbreviation and long form, only 17 were not correct, totalling 36 instances in the dataset. The 22 segments that contained these incorrect instances were completely removed from the final dataset. After this validation, the longest valid long form contains 26 tokens: *Multicenter, Randomized, Parallel Group Efficacy and Safety Study for the Prevention of Venous Thromboembolism in Hospitalized Acutely Ill Medical Patients Comparing Rivaroxaban with Enoxaparin*; and its associated abbreviation is *MAGELLAN*.

We also applied a similar validation based on the length of the abbreviations. By going through 141 instances of abbreviations that were longer than 15 characters (with spaces), we were able to identify 11 incorrect abbreviations, all of which with 19+ characters, which led to the removal of 15 segments from the final dataset. This helped us remove instances that had up to 145 characters and were clearly an error in the glossaries of the research papers from the PLOS corpus. After this validation of the long abbreviations, the longest abbreviation in the dataset has a total of 33 characters: *pos regul transcr RNA pol II prom*; and it stands for *positive regulation of transcription from RNA pol II promoter*.

After these two validation steps, we analysed a sample of a thousand random segments from the dataset, and there we observed that 5.5% of the segments presented wrong annotation (*i.e.*, at least one long form did not have its abbreviation as a co-reference, as we show in Example 2, where *paroxon* and *Pxn* are, respectively, long form and abbreviation denoting the same sub-

---

[6]https://spacy.io/

[7]For the purpose of this simple language model, numbers were replaced with the same placeholder used for content words.

(a) Length of long forms in tokens.

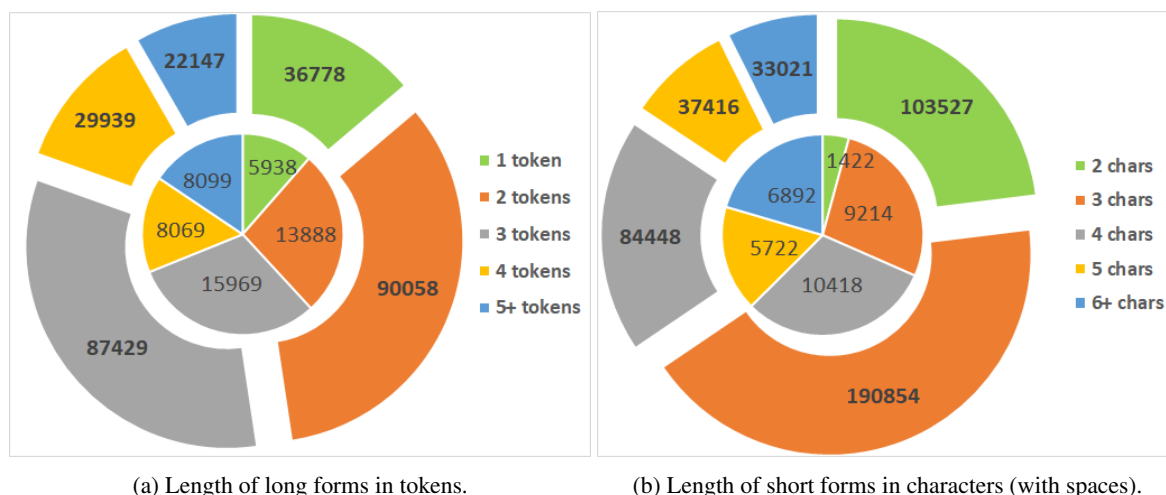(b) Length of short forms in characters (with spaces).

Figure 1: Distribution of short and long forms by length in the PLOD dataset. (Internal pie charts show number of unique forms, while the external, exploded doughnut charts display the total frequency.)

stance, but they are not co-referents in the segment), and 26.7% had missing annotation (*i.e.*, at least one abbreviation and/or one long form were missing from the annotated data, as we presented before in Example 1).

- Example 2: Km value of *paraoxon* towards selected **rh-PON1** mutants was nearly same while Kcat values differs however, correlates with the *Pxn*-hydrolyzing activity of the mutants as in Fig 2.

Although there are these few issues with the dataset, it does present very useful information for abbreviation and long form detection (which is the main focus of this paper), as we will show with an extrinsic evaluation in Sections 4 and 5, but it can also be used for typological studies related to abbreviation description. The dataset is mainly focused on biomedical texts, but it contains a representation of different types of abbreviations that can be further studied from a linguistic point of view. Besides the usual abbreviations composed by the initial letter(s) from the long form, such as **CI** (confidence interval) and **GFP** (green fluorescent protein), there are also those formed by suppressing some letters from one word, such as **TB** (tuberculosis) and **IFN** (interferon). Some short forms contain other abbreviations inside them and have a mix of upper- and lower-cased letters, such as **siRNA** (small interfering RNA) and **qPCR** (quantitative PCR), while others include parts that are not abbreviated, such as **cryo-EM** (cryo-electron microscopy). These are only a few examples of correctly annotated abbreviations that co-occur with their respective long forms in the dataset, and there are many more types and forms that can be further studied, for instance, from a Corpus Linguistics point of view. More information about quantities and types of long forms and abbreviations will be discussed in the next subsection.

3.3. Dataset Statistics

In this section, we describe the main statistics of the dataset that we are releasing. These statistics refer to the final PLOD_Filtered Dataset after all the data was removed in the validation steps described in the previous subsection.

Table 2 presents information regarding annotated segments and the amount of abbreviations and long forms divided by journal from the original PLOS corpus. It also shows the distribution of the extracted information over the different subject areas covered by the PLOS journals. As it can be seen in comparison to Table 1, the journal *PLOS Currents* is missing, as no segment was extracted from its files.

In Figure 1a, it is possible to have an idea of the distribution of long forms in terms of token length (split by space). Most of the long forms in the PLOD dataset have between 2 and 3 tokens, and, even though the number of unique 1- and 4-token-long long forms is representative, these forms are less frequently repeated in the texts.

A similar distribution can be seen in Figure 1b. It shows a higher concentration of 2- and 3-character abbreviations in terms of frequency. However, there is also an even higher number of unique 4-character abbreviations that are not as frequently used as the shorter abbreviations.

Another interesting figure from the dataset is that the number of unique long forms is 18k+ larger than the number of unique abbreviations. This serves as an indicative of the ambiguity among the existing short forms.

After all the validation steps, the PLOD dataset was ready for an extrinsic evaluation. We then moved on to an experiment for detecting abbreviations and long forms using several pre-trained language models. The setup for this experiment is explained in the following section.

684

	PLOD_test-unfiltered						SDU@AAAI-22 Shared Task_train + dev					
	Abbreviations			Long-forms			Abbreviations			Long-forms		
	P	R	F	P	R	F	P	R	F	P	R	F
ALBERT_base	0.845	0.898	0.871	0.758	0.812	0.784	0.682	0.638	0.659	0.462	0.154	0.231
BERT_base-cased	0.855	0.906	0.880	0.781	0.826	0.803	0.691	0.650	0.670	0.461	0.151	0.228
DeBERTa_base	0.877	0.910	0.893	0.817	0.874	0.845	0.682	0.638	0.659	0.462	0.154	0.231
DistillBERT_base	0.845	0.900	0.872	0.772	0.798	0.785	0.700	0.641	0.670	0.467	0.139	0.214
MPNet_base	0.846	0.899	0.872	0.782	0.823	0.802	0.691	0.606	0.645	0.466	0.145	0.221
RoBERTa_base	0.860	0.919	0.889	0.805	0.862	0.833	**0.707**	**0.641**	**0.672**	0.516	0.163	0.248
ALBERT_large	0.895	0.920	0.907	0.848	0.898	0.872	0.476	0.607	0.534	0.397	0.160	0.228
RoBERTa_large	**0.911**	**0.935**	**0.922**	**0.876**	**0.921**	**0.898**	0.515	0.650	0.575	**0.423**	**0.191**	**0.264**
BERT_large-cased	0.899	0.928	0.913	0.866	0.909	0.887	0.532	0.645	0.583	0.362	0.173	0.234

Table 3: Results of the fine-tuning-based abbreviation detection task where **Unfiltered** data was used for training and testing. The table also shows results where we used the same trained models, but tested them on the SDU Shared Task dataset.

4. Experiment Setup for Evaluation

In this section, we describe the experimental procedures for generating baseline models for detecting abbreviations and long forms. This methodology also serves as an extrinsic evaluation of the PLOD dataset. We used a customised NER pipeline from spaCy v3.2 [8] that utilises transformers for performing a sequence labelling task to detect abbreviations and long forms. SpaCy-transformer interoperates with PyTorch [9] and the HuggingFace transformers library [10], allowing us to access a series of pre-trained models based on state-of-the-art transformer architectures that were applied for generating our baseline models. In order to perform training with spaCy's pipeline, we annotated the PLOD dataset with an I-O-B scheme, where abbreviations were annotated as B-AB (i.e. Begin ABbreviation), and the words which were a part of the long forms were assigned B-LF (i.e. Begin Long Form) at the beginning, and I-LF (i.e. Inside Long Form) in the middle and end. This resulted in a one-token-per-line training file with the I-O-B annotation which amounted to 7,150,008 annotated tokens. We release the I-O-B-annotated dataset via a GitHub repository [11] along with the same dataset in the TSV format for researchers who wish to reproduce our experiment. We randomised and split our dataset into 70% instances for training, 15% for validation, and the remaining 15% as test data. To perform comparative evaluation, we trained models on both filtered and unfiltered data (as discussed in Section 3.2).

We utilised the following pre-trained Language Models (LMs) for the task of abbreviation detection: RoBERTa (Liu et al., 2019), BERT (Devlin et al., 2019), ALBERT (Lan et al., 2019), DeBERTa (He et al., 2020), DistilBERT (Sanh et al., 2019). For RoBERTa, BERT, and ALBERT, we used both *base* and *large* variants in our experiment. This resulted in an extensive extrinsic evaluation that was performed with the help of *nine LM variants*, and with different datasets. We trained all our models with a batch size of 128 and a hidden-layer size of 64. We used a spaCy Span-Get function which transforms each batch into a list of span objects for each sentence to be processed by the transformer. This technique helps with long sentences by cutting them into smaller sequences before running the transformer and allows for overlapping of the spans to cover both left and right context. We set the span window to 128 tokens and the stride to 96 to allow for overlapping of token windows. For tokenisation we used spacy.Tokenizer.v1. For optimisation, we used Adam optimiser with an initial learning rate of 0.00001 and initial warm-up steps set to 250, with up to a total of 20000 steps. We also chose 2 Maxout units (Goodfellow et al., 2013) as an activation function to calculate the maximum of the inputs. These architecture parameters were chosen because they have performed well for NER tasks [12]. The results obtained with the help of our models are presented in Tables 3 and 4.

After training a total of 18 models both on filtered and unfiltered data, we tested them on the test splits generated from PLOD and also on the English data provided by Veyseh et al. (2022b) (SDU@AAAI-22 Shared Task). Since the labelled test data from this shared task has not yet been released, we combined both the train and validation sets released by the organisers as a combined test set. With this set of experiments, we aimed to explore the efficacy of our models on a test set that is from a different domain and that contains data from other sources. Please note that the shared task data belongs to the task of Acronym Detection (AD) and our models do not make any specific distinction for acronyms, as they were trained on all sorts of short

---

[8] spaCy Transformers.
[9] PyTorch
[10] HuggingFace
[11] PLOD Dataset Github repository.

[12] spaCy: Facts and Figures

	PLOD$_{\text{test-filtered}}$						SDU@AAAI-22 Shared Task$_{\text{train + dev}}$					
	Abbreviations			**Long-forms**			**Abbreviations**			**Long-forms**		
	P	R	F	P	R	F	P	R	F	P	R	F
ALBERT$_{\text{base}}$	0.842	0.899	0.870	0.734	0.819	0.774	0.716	0.629	0.670	0.485	0.146	0.225
BERT$_{\text{base-cased}}$	0.853	0.902	0.877	0.766	0.834	0.799	0.723	0.628	0.672	0.471	0.150	0.228
DeBERTa$_{\text{base}}$	0.852	0.937	0.893	0.803	0.881	0.840	0.691	0.606	0.645	0.466	0.145	0.221
DistillBERT$_{\text{base}}$	0.842	0.904	0.872	0.763	0.805	0.783	0.709	0.642	0.674	0.456	0.140	0.215
MPNet$_{\text{base}}$	0.852	0.888	0.870	0.777	0.824	0.800	0.711	0.586	0.642	0.472	0.147	0.224
RoBERTa$_{\text{base}}$	0.857	0.918	0.886	0.798	0.867	0.832	**0.728**	**0.643**	**0.683**	**0.520**	**0.169**	**0.255**
ALBERT$_{\text{large}}$	0.840	0.918	0.877	0.770	0.830	0.799	0.532	0.651	0.585	0.373	0.174	0.237
RoBERTa$_{\text{large}}$	**0.906**	**0.935**	**0.920**	**0.874**	**0.925**	**0.898**	0.502	0.645	0.564	0.427	0.181	0.254
BERT$_{\text{large-cased}}$	0.892	0.931	0.911	0.858	0.912	0.884	0.532	0.651	0.585	0.373	0.174	0.237

Table 4: Results of the fine-tuning-based abbreviation detection task where **Filtered** data was used for training and testing. The table also shows results where we used the same trained models, but tested them on the SDU Shared Task dataset.

forms.

5. Results and Discussion

Based on the experiment setup discussed in Section 4, we performed the evaluation of the PLOD dataset in various scenarios. Initially, we performed fine-tuning with the **PLOD$_{\text{Unfiltered Dataset}}$**, and utilised nine variants of pre-trained LMs for the task. As can be seen in Table 3, our models are able to achieve a decent performance on both test sets, both in terms of precision and recall. We observed that the RoBERTa models seem to outperform the others, with the highest F1-scores in all of the cases. We also note that the RoBERTa$_{\text{large}}$ model shows significantly higher precision and recall values of 0.911 and 0.9335, with an F1-score of 0.922.

However, when testing this model trained on our data with the train+dev set of the SDU Shared Task dataset, we see a drop in performance. We attribute this drop in performance to various reasons: (1) There were spurious annotations in the SDU dataset which had been pointed out to the task organisers earlier this year when the dataset was released; (2) the domains used in the SDU dataset are 'legal' and 'scientific' whereas our dataset is mostly based on the Biomedical domain; (3) the drop in performance of long forms, specifically, can be attributed to incorrect classification of some tokens in the long forms which consist of many tokens. When the models trained on the PLOD$_{\text{Unfiltered Dataset}}$ are tested with the **SDU shared task data**, however, the RoBERTa$_{\text{base}}$ model seems to detect abbreviations better (0.672) than the RoBERTa$_{\text{large}}$ model; but for long forms RoBERTa$_{\text{large}}$ model outperforms (0.264) every other model.

In Table 4, we present the results based on the **PLOD$_{\text{Filtered Dataset}}$**. It seems that the RoBERTa$_{\text{large}}$ model and the RoBERTa$_{\text{base}}$ model again perform the task with significantly higher F1-score than others. We observed a similar performance on both the datasets, and each language model is performing better individually when trained on the unfiltered datasets. Our filtra-

tion process takes out many data points from the dataset which impacts the performance of the task. Also, on both the datasets BERT$_{\text{large-cased}}$ models also shows a comparable performance with minor differences from our best performing model.

When the models trained on the PLOD$_{\text{Filtered Dataset}}$ are tested with the **SDU shared task data** (Table 4), however, it can be seen that the results are more homogeneous in terms of model performance. The RoBERTa$_{\text{base}}$ model shows a much better performance for both long forms and abbreviations, attaining F1-scores of 0.255 and 0.683, respectively. We also observe that these models have higher precision values compared to the other models, especially RoBERTa$_{\text{large}}$, which leads to the next part of our discussion: confusion matrices. Since RoBERTa$_{\text{large}}$ performs well on our dataset, we show the confusion matrices obtained by fine-tuning on both Filtered and Unfiltered datasets in Figure 2.

Based on Figure 2a, we infer that there are a large number of abbreviations (AC tag) wrongly classified with the 'O' tag (6.8%, *i.e.,* 9631 tokens), when the Filtered dataset is used. However, using the Unfiltered dataset (Figure 2b, this number is reduced to 3.7%, *i.e.,* 5158 tokens. The overall accuracy of correctly classified AC tags is 93% and, for long forms, it is at 94.6%. The misclassified number of long forms stands at 2.2%, *i.e.,* ∼ 25k tokens, which is also a large number in a real-world scenario. This clearly indicates that there is a need to improve the model performance before we apply it to a real-world Biomedical domain scenario. Again, on the Unfiltered dataset, these language models show a better performance and the misclassified long-form tokens are reduced to 1%, *i.e.,* ∼ 11k tokens.

From the tables and confusion matrices above, we can conclude that overall, RoBERTa models perform the best for the task of abbreviation detection. However, given the current results, we also plan to conduct further experiments which use an ensemble approach with multiple models.
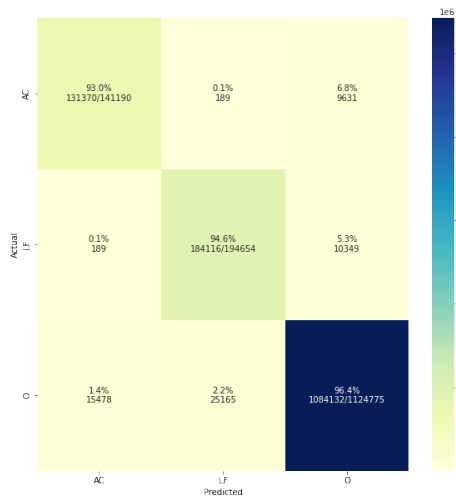
(a) Confusion Matrix on PLOD$_{\text{test-filtered}}$
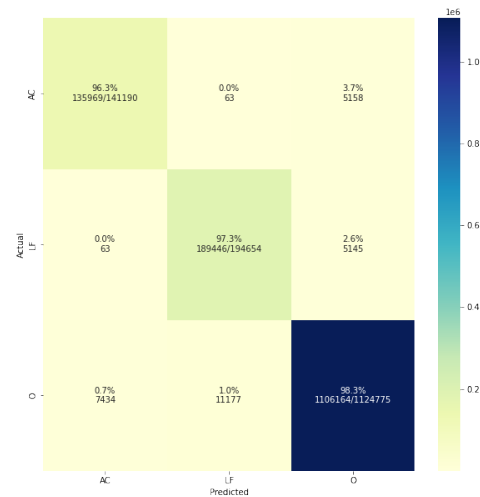
(b) Confusion Matrix on PLOD$_{\text{test-unfiltered}}$

Figure 2: Confusion Matrices over test set predictions from our best models, *i.e.,* RoBERTa$_{\text{large}}$

6. Conclusion and Future Work

In this study, we motivated the importance of abbreviation detection as an NLP task in the scientific domain and discussed the challenges one can encounter while trying to perform this task. We collected a large number of abbreviations and their corresponding long forms from open-sourced PLOS Journals and described the data collection process in detail. With some efforts towards the validation of this data, we were able to identify problems and further filter the dataset. Based on an unfiltered and a filtered version of this dataset, we performed an extensive evaluation of the abbreviation detection task by utilising various pre-trained language models. These models were not only tested on our test data but also on the SDU@AAAI-22 acronym detection shared task dataset. By analysing the results, we showed how some state-of-the-art transformer models fare at this task. With the hopes that these models might be of importance to the NLP community, we release them publicly along with the code and the raw datasets (both filtered and unfiltered).

In the future, we plan to extend this dataset with additional sources, which can be added to our data. We also plan to extend our experiments further with an ensemble approach, which can utilise various language models to perform the detection of abbreviations and their corresponding long forms.

Acknowledgement

Bibliographical References

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Ehrmann, M., Della Rocca, L., Steinberger, R., and Tanev, H. (2013). Acronym recognition and processing in 22 languages. *arXiv preprint arXiv:1309.6185*.

Fabijanić, I. (2015). A dictionary of abbreviations in linguistics: Towards a bilingual, specialized, single-field, explanatory dictionary. *Planning non-existent dictionaries*, page 113.

Goodfellow, I., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. (2013). Maxout networks. In *International conference on machine learning*, pages 1319–1327. PMLR.

He, P., Liu, X., Gao, J., and Chen, W. (2020). Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Kreuzthaler, M., Oleynik, M., Avian, A., and Schulz, S. (2016). Unsupervised abbreviation detection in clinical narratives. In *proceedings of the clinical natural language processing workshop (ClinicalNLP)*, pages 91–98.

Kubal, D. R. and Nagvenkar, A. (2021). Effective ensembling of transformer based language models for acronyms identification. In *SDU@ AAAI*.

Kuo, C.-J., Ling, M. H., Lin, K.-T., and Hsu, C.-N.

(2009). Bioadi: a machine learning approach to identifying abbreviations and definitions in biological literature. In *BMC bioinformatics*, volume 10, pages 1–10. BioMed Central.

Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*.

Larkey, L. S., Ogilvie, P., Price, M. A., and Tamilio, B. (2000). Acrophile: an automated acronym extractor and server. In *Proceedings of the fifth ACM conference on Digital libraries*, pages 205–214.

Li, F., Mai, Z., Zou, W., Ou, W., Qin, X., Lin, Y., and Zhang, W. (2021). Systems at sdu-2021 task 1: Transformers for sentence level sequence label. In *SDU@ AAAI*.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Moon, S., Pakhomov, S., Liu, N., Ryan, J. O., and Melton, G. B. (2014). A sense inventory for clinical abbreviations and acronyms created using clinical notes and medical dictionary resources. *Journal of the American Medical Informatics Association*, 21(2):299–307.

Nadeau, D. and Turney, P. D. (2005). A supervised learning approach to acronym identification. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 319–329. Springer.

Park, Y. and Byrd, R. J. (2001). Hybrid text mining for finding abbreviations and their definitions. In *Proceedings of the 2001 conference on empirical methods in natural language processing*.

Rogers, W., Rae, A. R., and Demner-Fushman, D. (2021). Ai-nlm exploration of the acronym identification shared task at sdu@ aaai-21. In *SDU@ AAAI*.

Sanh, V., Debut, L., Chaumond, J., and Wolf, T. (2019). Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Schwartz, A. S. and Hearst, M. A. (2002). A simple algorithm for identifying abbreviation definitions in biomedical text. In *Biocomputing 2003*, pages 451–462. World Scientific.

Taghva, K. and Gilbreth, J. (1999). Recognizing acronyms and their definitions. *International Journal on Document Analysis and Recognition*, 1(4):191–198.

Tchiotashvili, D., Garibashvili, M., and Geldiashvili, N. (2021). For the classification of business-related abbreviations and acronyms. *ECONOMICS, BUSINESS AND ADMINISTRATION*, (I):66–79.

Toole, J. (2000). A hybrid approach to the identification and expansion of abbreviations. In *RIAO*, pages 725–736. Citeseer.

Vanopstal, K., Desmet, B., and Hoste, V. (2010). Towards a learning approach for abbreviation detection and resolution. In *LREC*.

Veyseh, A. P. B., Meister, N., Yoon, S., Jain, R., Dernoncourt, F., and Nguyen, T. H. (2022a). MACRONYM: A Large-Scale Dataset for Multilingual and Multi-Domain Acronym Extraction. In *arXiv*.

Veyseh, A. P. B., Meister, N., Yoon, S., Jain, R., Dernoncourt, F., and Nguyen, T. H. (2022b). Multilingual Acronym Extraction and Disambiguation Shared Tasks at SDU 2022. In *Proceedings of SDU@AAAI-22*.

Vo, C. and Cao, T. (2019). Incremental abbreviation detection in clinical texts. In *2019 Joint 8th International Conference on Informatics, Electronics Vision (ICIEV) and 2019 3rd International Conference on Imaging, Vision Pattern Recognition (icIVPR)*, pages 280–285.

Wu, Y., Rosenbloom, S. T., Denny, J. C., Miller, R. A., Mani, S., Giuse, D. A., and Xu, H. (2011). Detecting abbreviations in discharge summaries using machine learning methods. In *AMIA Annual Symposium Proceedings*, volume 2011, page 1541. American Medical Informatics Association.

Wu, Y., Tang, B., Jiang, M., Moon, S., Denny, J. C., and Xu, H. (2013). Clinical acronym/abbreviation normalization using a hybrid approach. In *CLEF (Working Notes)*.

Xu, H., Stetson, P. D., and Friedman, C. (2009). Methods for building sense inventories of abbreviations in clinical notes. *Journal of the American Medical Informatics Association*, 16(1):103–108.

Yeates, S. A. (1999). Automatic extraction of acronyms from text. In *New Zealand Computer Science Research Students' Conference*, pages 117–124. Citeseer.

Zhu, D., Lin, W., Zhang, Y., Zhong, Q., Zeng, G., Wu, W., and Tang, J. (2021). At-bert: Adversarial training bert for acronym identification winning solution for sdu@ aaai-21. *arXiv preprint arXiv:2101.03700*.