Les Cahiers du GERAD ISSN: 0711–2440

# ACAS: A comprehensive framework for automatic abstract screening in systematic literature reviews

J. I. Serrato-Fonseca, A. M. Anaya-Arenas, J. Ortmann, A. Ruiz

G–2024–53

Septembre 2024

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2024
– Bibliothèque et Archives Canada, 2024

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2024
– Library and Archives Canada, 2024

**GERAD** HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

**Tél. : 514 340-6053**
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

# ACAS: A comprehensive framework for automatic abstract screening in systematic literature reviews

**Julia Isabel Serrato-Fonseca** [a, c, e]

**Ana María Anaya-Arenas** [a, c, e]

**Janosch Ortmann** [a, d, e, f]

**Angel Ruiz** [b, c]

[a] *Department of Analytics, Operations and IT, ESG-UQAM, Université du Quebec à Montréal, Montréal (Qc), Canada, H2X 3X2*

[b] *Department of Operations and Decisions Systems, Faculty of Business Administration, Université Laval, Québec (Qc), Canada, G1V 0A6*

[c] *CIRRELT, Université de Montréal, Montréal, Montréal (Qc), Canada, H3T 1J4*

[d] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[e] *CRI2GS, Université du Québec à Montréal, Montréal (Qc), Canada, H2X 3X2*

[f] *CRM, Université de Montréal, Montréal (Qc), Canada, H3T 1J4*

serrato_fonseca.julia_isabel@courrier.uqam.ca
anaya-arenas.ana_maria@uqam.ca
ortmann.janosch@uqam.ca
angel.ruiz@fsa.ulaval.caa

**Septembre 2024**
**Les Cahiers du GERAD**
**G–2024–53**

**Abstract :** When performing a Systematic Literature Review (SLR), the Abstract Screening Process (ASP) can be a very consuming and laborious task, especially when researchers retrieve a significant number of citations after running queries in scientific databases. By casting the ASP as a supervised machine learning (classification) problem, we propose a machine learning-based framework that generalises from a small subset of manually classified abstracts that greatly reduces the researchers' workload. Crucially, our approach explains the judgements that must be made and proposes different approaches from which researchers can choose, detailing the advantages and drawbacks of each choice. Our approach seeks a careful balance between ease of use and defence against overfitting on the one hand and lack of complexity on the other. A key innovation is the use of the acceptance sampling method from industrial engineering to evaluate the risk of non-detected positive instances rejected by our tool.

Our methods are implemented in a tool we call *AI-driven Comprehensive Abstract Selection* (ACAS). Implemented in Python, the ACAS tool is easy to use and freely available to researchers via GitHub. It can either be used out-of-the box or calibrated according to the researchers' needs. Numerical results on a real-life case study (a SLR on Operations Research methods in healthcare) shows a reduction of workload by 86.32% (corresponding to 522.46 hours of work), while offering a similar error risk as a human expert reviewer.

**Keywords :** Abstract Screening, citation screening, Systematic Literature Reviews, natural language processing, machine learning

# 1 Introduction

*Systematic Literature Reviews* (*SLR*) encompass the process of synthesizing research in a systematic, transparent, and reproducible manner to do a critical assessment and evaluation of all research studies that address a particular research question on a research topic (Tranfield et al., 2003). The stages for performing SLR are presented in Figure 1.

**Stage I: Planning the review**
Phase 0: Identification for the need for a review
Phase 1: Preparation of a proposal for a review
Phase 2: Development of a review protocol
**Stage II: Conducting a review**
Phase 3: Identification of research
Phase 4: Selection of studies
Phase 5: Study quality assessment
Phase 6: Data extraction and monitoring process
Phase 7: Data synthesis
**Stage III: Reporting and dissemination**
Phase 8: The report and recommendations
Phase 9: Getting evidence into practice

**Figure 1: Stages of a SLR adapted from NHS Centre for Reviews and Dissemination (2001) by Tranfield et al. (2003)**

The first stage comprehends the planning of the review by formulating research questions and designing the methodology that will be followed. The second stage starts with the collection and finding of citations for the review, by usually executing queries in scientific databases. The definition of sensitive and precise queries is required. Sensitive queries lead to retrieve assure that the highest possible proportion of all relevant papers are found. However, it will also yield to a high number of irrelevant studies. Moreover, a precise query leads to identify papers that meet exactly the inclusion criteria of the research, but it may miss several relevant papers (Gough et al., 2007). The second stage also involves the screening of retrieved citations to identify those of potential interest, according to inclusion and exclusion criteria previously defined in stage one. The second stage ends with the coding, synthesis and analysis of the citations selected (to be included in the review), leading to the final report of the SLR in stage three. Without doubt, the quality of the synthesis produced in a SLR is influenced by the papers that are selected. Hence, researchers need to be clear and explicit in the process they follow.

This paper contributes to phase four of the SLR process. In the *selection of studies* phase, the screening process is applied to the retrieved documents. The screening process usually start by researchers reading the title and abstract of each retrieved document to decide, by the application of the defined inclusion and exclusion criteria, if the document is of their interest or not. We will refer to this as the *Abstract Screening Process* (ASP) in the sequel. An ASP can be very time-consuming because the number of retrieved documents can be very large (Olorisade et al., 2016). Moreover, the proportion of relevant documents is often relatively small, so the screening process must be conducted thoroughly to ensure that none of the relevant documents is misclassified and therefore discarded. To reduce the workload and the time invested, while keeping the risk of error in the process as low as possible, different approaches have been recently developed for automatizing ASP in SLR using Machine Learning (ML) techniques. However, to the best of our knowledge, none of previous papers describing ML tools for ASP proposes a detailed framework allowing other researchers to replicate it on other instances. As transparency in the process is imperative when performing a SLR, as requested in the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) checklist, researchers need to provide details on the methodology followed, including on the automation tools that were used (Page et al., 2021).

The aim of this paper is to help filling this gap in the literature by proposing a general framework that could be followed by any researcher that seeks to automize the ASP in a SLR. More precisely, this paper provides three key contributions:

- We propose the *AI-driven Comprehensive Abstract Selection* tool (ACAS), a fully integrated ML-based abstract classification tool that researchers can use off-the-shelf or personalise according to their needs. Beyond the classification itself, the tool provides a set of decision rules that can be used to justify inclusion and exclusion decisions for the SLR.
- We provide detailed instructions and guidelines to use and implement ACAS in other SLR, focusing on the main challenges encountered in the process of building the classification tool and recommended key techniques to solve them. As part of our methodological approach, we implement a statistic quality control tool as a novel strategy to estimate the number of ignored relevant documents, given to researchers a way to make a more confident and informed decision in the ASP.
- Using a case study of a specific SLR process in healthcare logistics, we provide evidence on how the proposed framework does as well as human experts in the ASP, with a maximum of 10% error rate in the classification (Wang et al., 2020), but with an overall workload saving of 86.32%, or 522.5 of working hours out of 605.3 (based on a conservative hypothesis of four minutes per paper based on our own experience).

The rest of the paper is structured as follows: we give background on abstract screening in systematic literature reviews and on supervised machine learning in Section 2. Section 3 contains precise problem statement, while Section 4 is dedicated to a literature review on the stated problem. In section 5 we describe our methodology and illustrate it with the help of a case study of a SLR in healthcare logistics. Section 6 concludes the paper and discusses limitations of this research.

# 2 Background

In this section we state the abstract screening problem (ASP) in systematic literature reviews (SLR) and explain how it can be formulated as a supervised machine learning problem. We also list the main challenges that practitioners will typically face.

## 2.1 ASP in SLR

As motivated in the introduction, the ASP consists of assigning to each element of a large set of candidate papers a decision to either include or not the paper in the SLR. A key challenge lies in the potentially very large number of candidate papers to be screened manually. For example, in our case study there were over 9000 citations that corresponded to the search criteria. Manually screening all the candidate papers would be extremely time consuming and expensive. Furthermore, even if we were given unlimited resources, the error rate of the decision-maker rises with the workload (see for example Xie & Salvendy, 2000). On the other hand, spreading the work across several decision-makers would lead to the risk of inconsistency. Thus, we propose to reformulate the ASP as a supervised machine learning problem.

## 2.2 Supervised machine learning

*Machine learning* (ML) is a computational approach that enables information systems to automatically learn and improve from experience without being explicitly programmed for specific tasks (Alpaydin, 2020; James et al., 2023). ML is a subset of artificial intelligence and focuses on the development of algorithms that can analyse and make predictions based on data. The essence of machine learning lies in its ability to process large volumes of data and learn patterns or characteristics from this data to make informed predictions or decisions on new, unseen data. The methodologies employed in machine learning are broadly classified into supervised learning, where the algorithm learns from a labeled dataset (Alpaydin, 2020; Shalev-Shwartz & Ben-David, 2014); unsupervised learning, which deals with unlabeled data (Bishop, 2006); and reinforcement learning (Sutton & Barto, 2018), where an agent learns to make decisions by interacting with an environment. These approaches have been instrumental in advancing various scientific and industrial applications, from natural language processing and computer vision to predictive analytics in finance and healthcare.

As we will see further below, the branch of *supervised machine learning* is ideally adapted to the abstract selection problem. It focuses on the development of predictive models using labelled datasets (James et al., 2023; Mohri et al., 2018). In this framework, each instance in the training dataset is composed of input features and an associated target output. The primary goal of supervised learning is to construct a model that can predict the output for new, unseen instances based on the learned patterns from the training data. In supervised machine learning, there are two types of variables: the *features* $x_1, \ \ldots, \ x_n$ (also called *predictors* or sometimes *independent variables*) which are typically known in advance, and the *target variable* (sometimes also called *dependent variable*) $Y$ that needs to be predicted. The fundamental assumption of supervised ML is that there exists a functional relationship (up to a random error) between the target variable and the predictors. Mathematically speaking this means that there exists an (unknown) function $f$ such that

$$Y = f(X_1, ..., X_n) + \ \epsilon,$$

where $\epsilon$ is a random variable representing the error. To give a concrete example, suppose that we would like to estimate the market value of a house in terms of its habitable surface area, the number of bedrooms, the type of heating (such as gas, electric etc.), and whether there is easy access to public transport. In that case, the target variable $Y$ is the market value of the house and there are four predictors (or features) $X_1, X_2, X_3, X_4$, which correspond to the surface area, number of bedrooms, the type of heating, and the availability of nearby public transport. Note that the predictors can be quantitative (the first three) or qualitative (the forth one).

The supervised ML task is further divided according to the type of the target variable. If $Y$ is quantitative, one speaks of a *regression* task, whereas predicting a qualitative target variable is called *classification*. Thus, the previous example was a regression. For an example of a classification task, consider developing a supervised ML model to predict whether a patient has a non-small cell lung cancer (NSCLC), based on medical test results and health indicators. This is in fact a *binary classification* problem: the target variable can only take two values, namely whether the patient has the disease (positive) or not (negative).[1]

In supervised ML, the task is to "learn" the function $f$ using a labelled data set, called the *training set*, i.e. a collection of data points, each with associated input features and corresponding output labels (or target values). The main purpose of the training set is to provide the machine learning model with examples from which it can learn the underlying patterns or relationships between the input features and the output labels. This process of learning from the training set is referred to as *training* the model (James et al., 2023). The training set is composed of pairs of values for the predictors $X$ (features) and the target variable $Y$. In the first example of predicting the market value of a house, the training set would consist of a list of houses for which both the values of the predictors (surface area, number of bedrooms, availability of nearby public transport and type of heating) and the actual market value as measured, for example, by a sale transaction are known. For the example of diagnosing NSCLC, the training set would consist of medical test results and health indicators as well as the diagnosis (positive or negative) for a list of patients.

[1] In many application, the decision-maker may decide whether the he problem is considered to be a classification or a regression problem. For instance, they might try to predict whether a patient is running a fever (a binary classification problem), or simply to predict the patient's temperature (a regression task).

In supervised learning, the model uses the training set in a process called *fitting* or *training* to learn the unknown function that maps input features to the target variable. This can be done through various algorithms, which iteratively adjust the model parameters to minimise the difference between the predicted outputs and the actual labels in the training set. This difference is called the *training error*. The choice of algorithm depends on the nature of the problem (e.g., classification or regression) and the type of model being trained. After training, the model is evaluated on a separate *test set*, for which the true values of the target variable are also know. The difference between predicted and actual values on the test set is called the *test error*, and this quantity is used to determine how well the model generalises from its training. Once validated, the ML model is ready to be deployed on unseen data, that is, on data points where the values for the predictors but not those for the target variable are known.

Since there are numerous models for regression and classification tasks, metrics to evaluate these are required. We will be concerned with a binary classification problem, so we present only evaluation metrics for such a task and refer to (James et al., 2023) for a more general treatment.

Using the terminology of diagnosing NSCLC from above, a data point can have a positive or negative sample, and the model can assign a positive or negative classification. Thus, there are four different cases: 1) the model correctly assigns a positive (*true positive, TP*), or 2) a negative classification (*true negative*, *TN*); 3) the model can assign a positive classification to a sample with a negative label (*false positive, FP*), or conversely 4) the model can assign a negative classification to a positively labelled sample (*false negative*, *FN*). This information can be represented in a *confusion matrix*, in which the number of each of the four cases is given. In terms of quantitative measures, *accuracy* computes the proportion of correctly labelled samples as $\frac{(TP+TN)}{N}$ (where $N$ denotes the total number of samples), the *specificity* denotes the proportion of negative labels that were correctly identified: $\frac{TN}{(TN+FP)}$, *sensitivity* (or also referred as *recall*) computes the proportion of correctly identified true positive labels: $\frac{TP}{(TP+FN)}$, and *precision* describes the proportion of identified positives that are actually correct: $\frac{TP}{(TP+FP)}$. Table 1 presents the confusion matrix for a classification model.

**Table 1: A confusion matrix**

| | | "True label" | |
|---|---|---|---|
| | | positive | negative |
| Model classification | positive | True positive (**TP**) | False positive (**FP**) |
| | negative | False negative (**TN**) | True negative (**TN**) |

# 3 Problem statement

The ASP can now be framed as a supervised ML problem, more precisely a binary classification task where each abstract is labeled either "yes" (potentially included in an SLR) or "no" (not included). This formulation presents researchers with a sequence of challenges encountered at different stages of the machine learning process. When approaching ASP as a supervised ML problem, researchers are faced with six main challenges. We state these not in order of importance or difficulty, since this may vary from case to case, but rather in chronological order when they are encountered.

**Challenge I:** is the creation of a labelled dataset. While acquiring a large volume of unlabelled data, such as the abstracts of candidate papers, is relatively straightforward and cost-effective, the process of manually labelling this data to create a training set is resource-intensive and costly. The labelled dataset is essential for training the model; thus, this step represents a significant bottleneck in the setup phase.

**Challenge II:** The second challenge arises from the imbalanced nature of the dataset, with a typical skew towards negative instances (abstracts not suitable for inclusion). This imbalance can bias the ML model towards predicting the majority class, leading to a high rate of false negatives in the context of ASP.

**Challenge III:** Next, researchers must convert the unstructured text data of abstracts, which vary in length and format, into a structured form. This involves extracting a fixed number of features ($X_1,\ X_2,\ \ldots, X_n$) from the text, which can be used as predictors in the machine learning model. The process of *feature extraction* is crucial, as it must capture the essential information that influences the classification decision.

**Challenge IV:** Reproducibility is another critical challenge. The selection rules derived from the machine learning model must be transparent and comprehensible to non-experts, including editors and peer researchers. This requires the model to not only be accurate but also to provide explanations for its decisions that can be understood and verified by those outside the machine learning field. This is especially true in the case of ASP, as the transparency in the process and the reproducibility of the process is key in a SLR.

**Challenge V:** The machine learning model must be trained to have sufficient discriminatory power to make accurate yes/no labeling decisions. This entails selecting the appropriate machine learning algorithm and tuning it to handle the nuances and complexities of the data while maintaining the ability to discern between the classes effectively.

**Challenge VI:** Finally, the different consequences of false positives and false negatives must be considered. In the context of ASP, false positives (papers incorrectly selected for inclusion) are less detrimental since researchers can perform further manual screening. However, false negatives (relevant papers incorrectly excluded) are more problematic as they result in possibly important papers being lost for the SLR. Thus, the model needs to be particularly sensitive to minimizing false negatives.

# 4 Existing frameworks and review of relevant works in the literature

We contribute to the literature with an original methodology for the ASP that tackles specifically the challenges we enumerated before. We provide a reproducible methodology that focuses on implementation in real-life examples, inspired both from the existing literature and the difficulties we encountered when performing a SLR of our own. As per the literature, we base our contribution in the results of the research and analysis of 52 papers found in the recent reviews proposed by O'Mara-Eves et al. (2015); and van Dinter et al. (2021).

The existing literature in automated ASP can be split into two groups of papers: Group **G1** are those studies that extract their own data set (such as Dalal et al., 2013). In studies belonging to G1, the authors perform a keyword-based search in literature catalogues such as SCOPUS and extract all references that correspond to the search.

On the other hand, Group **G2** are those studies who rely on open access benchmark data sets to train and test ML models and compare their results with previous studies. These models were not developed with the intention of implementing them on a new SLR. Therefore, they do not encounter several of the challenges we identified. Specifically, Challenges I and II, and to some extent also IV and VI do not occur in a controlled setting of using instances from benchmark data sets with known value of the target variable.

For instance, Matwin et al. (2010), Kim & Choi (2012), Jonnalagadda & Petitti (2013) and Olorisade et al. (2019) used the benchmark data set of Cohen et al. (2006) whereas Miwa et al. (2014) used the benchmark data set in Wallace, Small et al. (2010) and Wallace, Trikalinos et al. (2010). Some studies, such as Kontonatsios et al. (2020), evaluated their methodology with both benchmark data sets.

When we revised the related literature, we encountered an opportunity for developing an easy understanding reproducible methodology for ML non-experts. We analyzed the main steps of previous ML contributions for the ASP as a starting point. In the following section we outline the general methodology observed among studies. Then, we discuss how the different studies handled the challenges presented in Section 2.3.

## 4.1 Shared process among studies

We read and analysed the 52 references we found on the subject. As a result, we identified the following shared process.

**Step 1:** Most studies consider the first step of the ASP to be *collecting citations* for constructing the classification model. As presented in the introduction of this section, there are two ways of doing this: examples from the group G1 of studies building an ASP for a new SLR typically proceed by running searches in scientific databases, or by employing other citation extraction techniques. For studies applying a model to an existing data set (G2), this step is not necessary because a well-curated data set already exists. In both cases, the number of samples included in the data set can vary from about hundred up to tens of thousands of citations.

**Step 2:** For G1, the sampled citations must be *labelled* manually as positive (include) or negative (discard), based on the researchers' predefined inclusion and exclusion criteria. Again, this step does not apply to studies from G2 working on a pre-existing labelled data set.

**Step 3:** The labelled sample of citations undergoes a *vectorization* process and is then partitioned into training and testing sets. This relates to challenge III: the data set consists of unstructured data (text of variable length) and must be turned into a vector of fixed size. This process is called *vectorization* and involves two parts known in the literature as *feature extraction* or *feature induction*, and *feature representation.*

In the feature extraction process, as described by Ouhbi et al. (2016), the titles and abstracts of the citations are tokenized. Next, common English stop words and non-alphabetical characters are removed using lists like the one provided by Carroll et al. (1971). Then, the words are *stemmed* by removing their morphological affixes. These stems conform the features in the classification model.

Afterwards *feature representation* takes place, where the previously extracted features are transformed into a numerical format so a ML algorithm can process them. One of the most common feature representation approaches is a *bag-of-words* (BOW), which involves "vectorizing the citations by transforming them into high-dimensional points that are represented in sequences of numbers" (Marshall & Wallace, 2019). The BOW represents the frequency of occurrence of each word in each text. The presence or absence of each word in the citation can be indicated (weighted) either binarily, by its frequency (commonly referred as Term Frequency or TF), or even using more complex approaches (e.g. Term Frequency–Inverse Document Frequency or TF-IDF as in García Adeva et al., 2014). BOW is also referred to as *uni-gram*, because it is a special case of variation of the *N-gram* approach (Jurafsky & Martin, 2014) that considers sequences of up to $N$ words, where $N$ is an input parameter chosen by the study authors. Some studies, such as Gonzalez-Toral et al. (2019), used more complex methods such as word embeddings or attention-based algorithms.

Finally, the collection of resulting vectors, called the *vectorized dataset*, is partitioned into training and testing sets. The proportion of the test set varied considerably between the different studies.

**Step 4:** Next, ML algorithms are trained and tested. The most common classification models used are Support Vector Machines (SVM) (e.g. Ros et al., 2017) and Naïve Bayes (NB) (e.g. Almeida et al., 2016).

**Step 5:** Finally, the performance of the classification models is assessed. We observed that there is not really a consensus in the selection of the performance measures among the studies. Recall, Precision, general Accuracy, F-measures, were within the most popular choices as in Ros et al.

(2017), along with AUC score as in Gonzalez-Toral et al. (2019), and Work Saved over Sampling (WSS, a measure to balance between very high recall and optimal precision) as in Kontonatsios et al. (2020).

## 4.2 Approaches in the literature for addressing the challenges

To finalize this section, we present the approaches that have been proposed in the literature to tackle the challenges identified in Section 2.3.

### 4.2.1 Challenge I (dataset creation)

One of the important questions when developing a ML model for the ASP of a SLR concerns sampling a database and labelling the citations. Recall that this only applies to the group G1 of studies creating their own data set. For instance, Bannach-Brown et al. (2019) applied a sampling strategy: approximately 10% of the 70 365 references produced after running the query on scientific databases were uniformly randomly selected for training, while the trained ML algorithm was used for classifying the rest of the citation corpus. However, we found that almost all studies in G1 did not employ any sampling strategy. Instead, they labeled all the citations obtained from running queries in scientific databases or using other citation retrieving approaches. Their contribution to the literature focused more on testing their developed classifiers than on applying a methodology in an actual SLR. For example, Rubio & Gulo (2016) ran a queried “Smart Grid Multi-Agent Systems” in three different repositories: IEEE Xplore, ACM Digital and Springer. They retrieved 790 citations, manually labelled all of them, and used 70% of the data set for training, and 30% for testing. Another example is Kim & Choi (2014), who selected 19 SLRs, obtained the citations they referenced, and then expert reviewers labelled all of them as included or excluded based on their relevance to specific topics and common exclusion criteria. They reserved a portion of the datasets for testing, using the rest for creating various training sets by considering different combinations of included and excluded citations, which were then used for training SVMs.

### 4.2.2 Challenge II (imbalanced dataset)

The *class imbalance problem* refers to the fact that a very large percentage of the retrieved citations will not meet the inclusion criteria and will be excluded from the SLR, leading to a disproportion of classes: there are many more negative than positive labels, which has performance repercussions. Indeed, if we expect the ML model to recognize positive instances, we must expose it to enough positive instances during its training. This is the case for both the G1 and G2 of papers. For instance, some of the synthetic datasets contain only 4.51% positive instances (Cohen et al., 2006), and this ratio drops below 1% in others (Gonzalez-Toral et al., 2019; Yu et al., 2008). To address this, undersampling, weighting and oversampling techniques can be applied (O’Mara-Eves et al., 2015). *Undersampling* means excluding some of the negatively labelled instances from the dataset (Miwa et al., 2014; Shemilt et al., 2014; Wallace, Small, Brodley, Lau, & Trikalinos, 2012). *Weighting* grants greater weight to positive instances than to the negative ones when training the classification models. In Timsina et al. (2016) an *oversampling* approach is explored, where the number of instances in the smaller class is increased by adding several copies of selected instances, possibly distorted by noise (this is called creating *synthetic examples*). While this technique reduces the class imbalance without reducing the sample size, it has other downsides. For example, it does not address the problem that there are only few examples from the smaller class that the model can learn from. Moreover, if synthetic data is created, this introduces extra noise and possibly unrealistic examples. Oversampling may also distort the distribution of the predictors in the oversampled class.

### 4.2.3 Challenge III (unstructured data)

The next main challenge in this process is to convert unstructured data to adequate predictors variables. The most common feature extraction approaches we observed were BOW and N-grams. The word list used often included stems obtained from citations (abstract, title and keywords) and vocabulary thesauruses (i.e. lists of terms relevant to the subject area) such as MeSH, UMLS, and/or MEDLINE terms. In medical related benchmark data sets, authors commonly incorporate MeSH terms (from PubMed) and Unified Medical Language System (UMLS) terms when building their BOW. On the other hand, MEDLINE is the National Library of Medicine's (NLM) bibliographic database which focuses on life sciences, especially on biomedicine. Some studies used alternative feature extraction approaches like *SemRep*, which is a tool specifically designed for biomedical text analysis. It is a UMLS-based program that extracts from sentences three-part propositions, called semantic predications. Whereas the use of these methodologies is a huge advantage for medical related fields, this strategy is not replicable to other scientific domains.

#### 4.2.4 Challenge IV (reproducibility)

To facilitate the reproduction of methodologies proposed in the literature, some works have used visual representation tools to help readers follow their workflows. For example, Cawley et al. (2020) provided a visual summary of their process, and Kontonatsios et al. (2020) presented a diagram of their framework's architecture. In addition, some works provided comprehensive details for reproducibility, such as the datasets they used, the proportion of positives and negatives, feature extraction methods, classification models, and performance measures. However, some works lacked specificity in aspects like preferred feature representation techniques and cross-validation approaches. Not explicitly addressing these details may lead to multiple interpretations and reduce reproducibility.

#### 4.2.5 Challenge V (machine learning model)

Choosing the best model and specifying suitable hyperparameters is an important aspect of machine learning. Works like Kim & Choi (2014), Hashimoto et al. (2016) and Olorisade et al. (2019) used variations of Support Vector Machines, citing previous studies that achieved good results when automizing the ASP. Other works, such as Sellak et al. (2015), who used a Bit-priori Association Classification Algorithm (BACA), Kontonatsios et al. (2017), who employed active learning and label propagation methodologies, and Gonzalez-Toral et al. (2019), who utilized semantic similarity models for clustering and ranking, chose these approaches because they had not been previously used for automating the ASP. Finally, works like (Wallace, Trikalinos et al., 2010), Matwin et al. (2010), and Frunza et al. (2011) justified using models like complement Naïve Bayes and Support Vector Machines with an active learning framework, arguing that these methods are effective for handling imbalanced datasets.

Regarding hyperparameter selection the most common approach has been grid optimization as in and Bekhuis et al. (2014), Ros et al. (2017) and Olorisade et al. (2019).

#### 4.2.6 Challenge VI (different consequences of false positives and false negatives)

In the context of ASP, False Positives (irrelevant citations misclassified as relevant) are less detrimental since researchers can perform further manual screening. However, False Negatives (relevant citations misclassified as irrelevant) are more problematic as they result in possibly important papers being lost for the SLR (Wallace, Small, Brodley, Lau, Schmid et al., 2012). Wallace, Small, Brodley, Lau, Schmid et al. (2012) conclude that the ML model used for the abstract selection needs to be particularly attentive to minimizing False Negatives.

Accordingly, evaluation measures that prioritize True Positives (relevant citations classified correctly) are common in the literature. For instance, Bekhuis & Demner-Fushman (2010) stated in that the evaluation measures of their choice were mean Recall, mean Precision, and the F1- measure (the harmonic mean of Recall and Precision, see Section 2); while Cohen et al. (2006) proposed and used *Work Saved over Sampling at 95% Recall* (WSS@95%). The latter estimates the advantage of a given classification model in terms of time saved (i.e. papers not read) over uniformly randomly sampling

95% of the papers. WSS@95% is frequently used in the literature, see for example Olorisade et al. (2019) and Kontonatsios et al. (2020). However, Cohen et al. (2006) themselves pointed out the simplifying assumptions under which the metric was conceived, and more recent works have made the same critique such as Kusa et al. (2023).

# 5 Methodology and case study

We now present our methodological approach, which is subdivided into five main steps: data set creation, feature engineering, training, cut-off calibration and final validation. We also propose guidelines related to what to try when the classifier's performance is initially not acceptable, and we provide insights on how the tool's behaviour should respond to these changes. Figure 2 sketches the complete framework.

As we describe our methodology, we illustrate the choices made along the way with a case study, namely on the application on operations research methodologies to problems in healthcare logistics.

## 5.1 Creation of the data set

Following a pre-selected set of queries designed for a SLR in the field of Operations Research applied to healthcare (summarised in Appendix A1.), more than 9 thousand citations were retrieved from SCOPUS database. These papers are used to form the full dataset, from which, 171 papers were randomly sampled for inclusion in the training set and manually classified. The size of this sample set is a necessary trade-off proposed to address Challenge I, namely between having enough samples to draw statistically significant conclusions and the effort needed to manually label citations. The task of manually classifying 171 references is manageable, while the sample is large enough to be representative of the overall population of papers.

Due to the class imbalance expressed in Challenge II, only 11 papers (about 6.5%) were given a positive classification. As discussed in Section 3, neither under- nor oversampling are satisfactory responses in our situation. Instead, we propose a new approach, where a further 56 papers that were known to the authors to be relevant to the SLR were added to the training set, bringing the total size of the training set to 227, of which 67 had a positive (relevant to the SLR) classification. In this way, extra information is injected in the training set by exhibiting entirely new positive instances to the model.

## 5.2 Feature engineering

Challenge III consists of converting unstructured data – the paper abstracts – into a set of vectors, the predictors. This process, known as text vectorization, is an aspect of natural language processing (NLP; see for example Manning & Schütze, 1999) and involves converting the abstract text into numerical vectors that can be fed into a supervised ML model as predictors. We have chosen to use the N-grams vectorization technique (see the literature review) more specifically uni-, bi- and trigrams. This method contrasts with other vectorization techniques, such as bag-of-words explained in Section 4.1, which disregard word order and context, or more sophisticated models like word embeddings, which require substantial computational resources and large datasets to capture semantic relationships effectively.

Our choice to employ N-grams for vectorizing the abstracts of academic papers is motivated by a trade-off between capturing the syntactic structure of language and computational efficiency. This balance is crucial when dealing with the dense and complex language often found in academic writing, where the precise arrangement of words can significantly alter meanings and implications. Using N-grams allows us to handle domain-specific terminology and phrases common in academic literature, facilitating the identification of key concepts and themes within a field. Unlike methods such as bag-of-words, N-grams captures word order and syntactic structure, while requiring much less powerful computational resources and being easier to interpret than word embeddings like Word2Vec (Mikolov, Chen et al., 2013; Mikolov, Sutskever et al., 2013) or GloVe (Pennington et al., 2014), and deep learning models such as transformers. The choice of limiting the context length $N$ to 3 represents the same trade-off: the N-gram approach considers context, so N should be not "too" small. On the other hand, the number of possible N-grams (and hence the number of features) grows exponentially with N, leading to the problem of curse of dimensionality. As a trade-off between these two issues, we selected uni-, bi- and trigrams, which is a standard choice in the literature.

The process of vectorizing using N-grams consists of two steps: stemming and the creation of the N-grams. Before stemming a pre-processing step took place where unnecessary characters, such as punctuation, numbers, or special characters were removed, as well as the most common English words (such as articles and prepositions). The actual stemming consists of reducing words to their roots by removing their morphological affixes: for instance, "scheduling", "reschedule" and "schedule" were all reduced to the stem "schedul". Furthermore, to avoid including an excessive number of features, only stems with more than three characters and had a frequency of five or more were considered.

## 5.3 Model selection

Model selection and training, responding to challenges IV and V, consisted of two steps: train-test-split and training. For the first step, we randomly selected 70% of the instances (158) in the vectorized sample for training and assigned the remaining 69 instances (30%) to the test set. For model selection we chose two different candidate models: a Logistic Regression (LR) and a Decision Tree (DT). We have made this selection because LR and DT are both explainable models, allowing a user of the tool to understand which N-grams contributed to the classification decision and why. Moreover, these models require neither kernel selection decisions nor major hyperparameter tuning requirements, reducing computational burden and potential sources for error. Once trained, we choose the model according to their performance on the test set.

The results produced by the two classification models (LR and DT) are presented in Table 2. The LR model performed very well (97% accuracy on the test set with high sensitivity and specificity) and the DT model performed quite well (78% test accuracy), with a low generalisation error and an absence of the usual signs of overfitting. In addition, we ensured that the model demonstrated sample stability by performing a traditional 10-fold cross-validation and confirming that more than 90% accuracy was achieved. Due to the superior predictive performance of the LR model, it was chosen for further analysis.

**Table 2: Confusion matrices of the two classification models**

| | **LR model** | | **DT model** | |
|---|---|---|---|---|
| | **Actual positive** | **Actual negative** | **Actual positive** | **Actual negative** |
| **Testing set** | | | | |
| Predicted positive | 20 | 1 | 8 | 2 |
| Predicted negative | 1 | 47 | 13 | 46 |
| **Training set** | | | | |
| Predicted positive | 46 | 0 | 45 | 0 |
| Predicted negative | 0 | 112 | 1 | 112 |

## 5.4 Cut-off calibration: acceptance sampling plan

The actual output of most binary classification models, including those considered is a numerical quantity between 0 and 1, representing the probability predicted by the model of a positive classification. Thus, if the output is close to 1, the model predicts that the instance likely belongs to the positive class and if the output probability is close to 0, the model predicts that the instance likely belongs to the negative class.

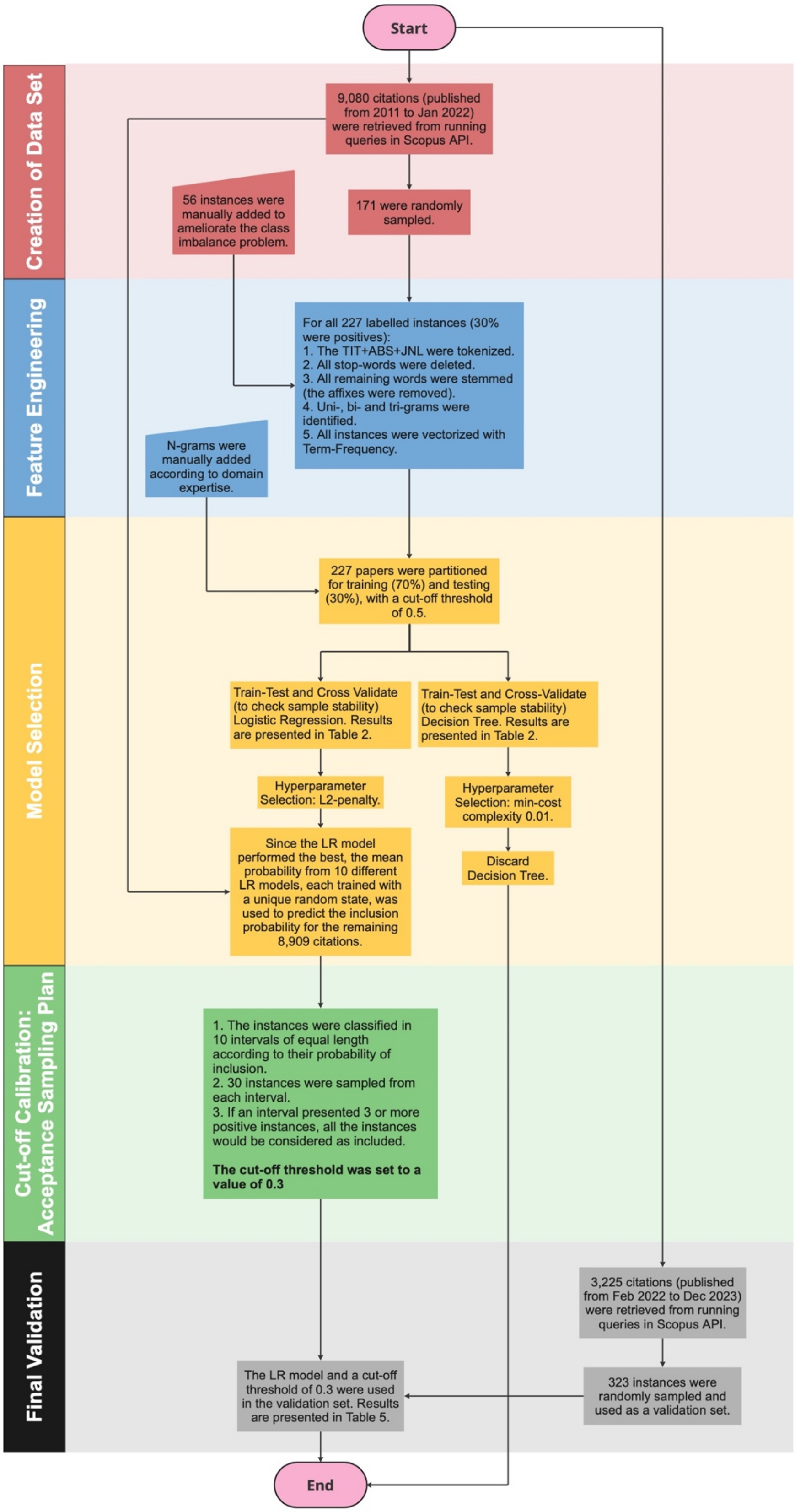
Start
Creation of Data Set
9,080 citations (published from 2011 to Jan 2022) were retrieved from running queries in Scopus API.
56 instances were manually added to ameliorate the class imbalance problem.
171 were randomly sampled.
Feature Engineering
For all 227 labelled instances (30% were positives):
1. The TIT+ABS+JNL were tokenized.
2. All stop-words were deleted.
3. All remaining words were stemmed (the affixes were removed).
4. Uni-, bi- and tri-grams were identified.
5. All instances were vectorized with Term-Frequency.
N-grams were manually added according to domain expertise.
Model Selection
227 papers were partitioned for training (70%) and testing (30%), with a cut-off threshold of 0.5.
Train-Test and Cross Validate (to check sample stability) Logistic Regression. Results are presented in Table 2.
Train-Test and Cross-Validate (to check sample stability) Decision Tree. Results are presented in Table 2.
Hyperparameter Selection: L2-penalty.
Hyperparameter Selection: min-cost complexity 0.01.
Since the LR model performed the best, the mean probability from 10 different LR models, each trained with a unique random state, was used to predict the inclusion probability for the remaining 8,909 citations.
Discard Decision Tree.
Cut-off Calibration: Acceptance Sampling Plan
1. The instances were classified in 10 intervals of equal length according to their probability of inclusion.
2. 30 instances were sampled from each interval.
3. If an interval presented 3 or more positive instances, all the instances would be considered as included.
The cut-off threshold was set to a value of 0.3
Final Validation
3,225 citations (published from Feb 2022 to Dec 2023) were retrieved from running queries in Scopus API.
323 instances were randomly sampled and used as a validation set.
The LR model and a cut-off threshold of 0.3 were used in the validation set. Results are presented in Table 5.
End

**Figure 2: Proposed framework for the development and implementation of a ML tool for ASP**

To address Challenge VI and to convert the numerical output into a classification, a *cut-off* is used: if the predicted probability is greater than or equal to the cut-off, the model classifies the instance as belonging to the positive class (in our case, that the article is relevant to the SLR). Otherwise, the model classifies the instance as belonging to the negative class (not relevant to the SLR). Adjusting the cut-off threshold affects the sensitivity (true positive rate) and specificity (true negative rate) of the model: a lower threshold means the model is more sensitive (produces more true positives) but decreases specificity (more false positives), and vice versa. This is because lowering the threshold makes the model more likely to predict positive class labels. Moreover, Bannach-Brown et al. (2019) conclude that choosing a cut-off threshold is an important yet open question in the field of information retrieval.

We propose a novel approach to this question inspired by *acceptance sampling*, a statistical quality control technique used in industrial engineering for determining if a production lot within a sequence meets the required quality standards to satisfy both the customer and the manufacturer. Given the characteristics of our problem, we are going to approach it with a *single sampling by attributes* plan, where lots can be classified as either accepted or rejected (see for example Schilling & Neubauer, 2017). In the simple sampling acceptance plan, each lot is sampled, the items in the sample are inspected and if the number of faults in the sample exceeds a threshold, the whole lot is rejected. Otherwise, the lot is accepted.

As in the classical industrial engineering theory, both the buyer (customer) and the seller (or manufacturer) of the product specify acceptable proportions of defects in the overall (unknown) population. The buyer will not accept more than a proportion $p_1$ of defects, whereas the seller will not wish to destroy the lot unless at least a proportion of $p_2$ products are defective. The values $p_1$ and $p_2$ represent the *acceptance quality level* (AQL) and *lot tolerance percent defective* (LTPD) respectively. Afterwards, a sample of $n$ items is considered and tested. The lot will be rejected if, and only if, the proportion of defective items in the sample is larger than $c$. Now $n$ and $c$ are negotiated and chosen such that the risks for the buyer and the seller to have unacceptable levels of defects are $\alpha$ and $\beta$ respectively. By modelling the lots as independent and identically distributed samples, Dodge & Romig (1944) develop a model that allows to compute $n$ and $c$ in terms of the user-determined parameters $\alpha$, $\beta$, $p_1$, and $p_2$. In particular, $p_1$ and $p_2$ follow a hypergeometric distribution.

In our analogy, *defective* samples correspond to positive labels, i.e. papers that are to be considered in the literature review. Therefore, *rejecting a lot* means that the number of positive labels (defects) are beyond the control value $c$, and the lot then needs to be fully explored or analyzed. On the same way, *accepting a lot* (a group of papers) means to exclude it without a detailed inspection. We wish to limit both the risk of accepting an irrelevant paper and rejecting a pertinent one. Thus, $p_1$ one sets the largest proportion of positive instances in a lot that should still lead to exclusion, while $p_2$ is the minimum proportion of positives at which an interval should be screened. However, for a SLR, the impact of rejecting a relevant paper is much higher than that of accepting an irrelevant one (due to the possibility of further screening), so the value of $\beta$ must be set higher than that of $\alpha$.

Traditionally, if the LR model provides a probability of being relevant (positive instance) of 0.5 or more, then the set of papers should be included in the review. On the contrary, if the probability is less than 0.5, they should be excluded from the review without further inspection. We however propose to apply the acceptance sampling plan to the papers with a probability lower than 0.5 to validate if they should indeed be excluded from the SLR.

To apply acceptance sampling to our case, we split the papers into intervals $i$ giving the predicted probability to be relevant to the study according to be relevant to the study the LR model, and this for instances with a reported probability of 0.5 or less. Each interval has then $N_i$ instances (citations) and the intervals will be treated as independent lots to be tested. The accepting sampling plan requires sampling of $n$ instances from each lot to be manually screened by the researcher to determine the number $d_i$ of defects (papers to be included). The entire batch is then considered to be defective (i.e. a candidate for inclusion) if $d_i > c$.

Based on standard values in the literature (Dodge & Romig, 1944) and recalling the higher impact of false negatives than false positives, we defined our quality level expectations to at least $\alpha = 0.001$ and $\beta = 0.1$. Then, given our context, where encountering a higher proportion of positives is ideal, unlike in an industrial setting where finding few defects is preferable, we propose using the following parameters: $p_1 = 0.01$ and $p_2 = 0.2$. Using these input values, we obtained $n = 30, c = 3$ from the acceptance sampling plan. Therefore, 30 citations were randomly sampled from each of the first five intervals, then manually screened. The actual number of positive instances found in each sample is presented in Table 3.

**Table 3: Number of positive instances found within a sample of size n=30 taken from the first five intervals**

| Interval | Number of positive instances |
|---|---|
| [0.00 – 0.10) | 0 |
| [0.10 – 0.20) | 2 |
| [0.20 – 0.30) | 1 |
| [0.30 – 0.40) | 5 |
| [0.40 – 0.50) | 4 |

Given that the screening of the samples drawn from intervals [0.4, 0.5) and $[0.3, 0.4)$ yield more than 3 positive citations, the final cut-off threshold was set to 0.3. The true acceptance rate of the seven included intervals was calculated (all the citations in these intervals were manually screened), leading to Table 4, which presents for each interval, the number of citations in the interval (instances screened), the number of citations that are indeed positive and should be included in the review and the calculated the acceptance rate (included over screened). Overall, 13.68% of the instances in the full dataset were kept for further analysis.

**Table 4: Final percentage of positive instances in each interval and included instances**

| Interval | Total citations | Included citations | 100* (Included/screened) |
|---|---|---|---|
| [0.00 – 0.10) | 7003 | - | - |
| [0.10 – 0.20) | 550 | - | - |
| [0.20 – 0.30) | 284 | - | - |
| [0.30 – 0.40) | 173 | 15 | 8.33% |
| [0.40 – 0.50) | 140 | 25 | 20.71% |
| [0.50 – 0.60) | 133 | 21 | 18.04% |
| [0.60 – 0.70) | 116 | 29 | 25.0% |
| [0.70 – 0.80) | 116 | 49 | 42.24% |
| [0.80 – 0.90) | 125 | 45 | 36.0% |
| [0.90 – 1.00] | 440 | 283 | 64.54% |

## 5.5 Final validation

In the previous steps, we have developed two convincing classification models. However, as formulated in Challenge IV, reproducibility (i.e., the ability of the model to perform adequately on completely new data) is a key requirement. To shed some light on this matter we have sampled a further, independent, validation set to test the out-of-sample performance. To be sure that no information leak occurs, we have launched a database search with the same query, and retrieved papers published between February 2022 and December 2023 (recall that the original search covered only papers up to end of January 2022). The new search to this date range led us to obtain 3225 new citations, from which 323 (10%) papers were randomly sampled and manually classified by the researchers. We found that our chosen LR model performed well on this unseen data, with an overall accuracy of 90% and a specificity (probability of discarding a true positive) of 98.5%. The confusion matrix is shown in Table 5.

For comparison, the validation performance of the DT model, discarded in the previous step due to its inferior performance compared to the LR model, is shown in Table 6. It is worth noting that

Table 5: Validation performance of the LR model

| | LR model | |
|---|---|---|
| | **Actual positive** | **Actual negative** |
| **Validation set** | | |
| Predicted positive | 23 | 28 |
| Predicted negative | 4 | 268 |

the expectation that the DT model performs worse is confirmed with respect to accuracy, specificity and sensitivity.

According to Wang et al. (2020) a human reader can aspire to an overall accuracy of approximately 90% when performing a classification for an ASP. In other words, our LR model performs roughly at the same level as a human reader. On the other hand, it saves an inordinate amount of work: only 1203 of the overall 9080 papers to be classified were screened by a human reader, which corresponds to an overall workload saving of 86.32%, or 522.5 hours of work.

Table 6: Validation performance of the DT model

| | DT model | |
|---|---|---|
| | **Actual positive** | **Actual negative** |
| **Validation set** | | |
| Predicted positive | 19 | 38 |
| Predicted negative | 8 | 258 |

# 6 Discussion and conclusion

We introduced AI-driven Comprehensive Abstract Selection (ACAS), a machine learning based tool to help decision makers choose which papers to include in their systematic literature review (SLR). It is designed to produce a full pipeline for the abstract selection process (ASP), from data set creation via feature extraction to generating new inclusion or exclusion decisions.

The tool is available on GitHub.[2] It can be used off-the-shelf and does not require any prerequisite knowledge about machine learning. At the same time, ACAS is completely open source and has been designed to allow authors to modify the choices we made to their specific needs.

In this paper, we have discussed the key challenges that authors have faced and provided guidelines how to address them. In this way, authors can personalise our ACAS tool. We have illustrated the utility of ACAS on a case study, a SLR of the utility of operations research in healthcare. Our results show that by using ACAS, we were able to reduce the workload by 86.32% (corresponding to 522.46 hours of work), while maintaining error rates comparable to that of human experts.

Our approach is deliberately based on explainability and reproducibility. For example, our feature extraction method is based on n-grams rather than autoregressive or bidirectional encoders. Similarly, we have chosen explainable machine learning models (logistic regression and classification trees). We have developed a novel way convert the probabilistic forecasts of the models into class assignments (i.e. choosing a cut-off value), inspired a classical idea from industrial engineering: an acceptance sampling plan.

By using our tools and our guidelines, authors will be able to provide a reproducible, explainable set of decision rules to justify the inclusion and exclusion decisions for their SLR.

[2] https://github.com/janoschortmann/abstract-screening

Finally, it is important to consider the limitations of this study. It is assumed that the manual labelling of the citations is done correctly, adhering to the inclusion and exclusion criteria established by the researchers. The same statement applies to the manually added terms in the n-grams list. Additionally, it is assumed that uni-, bi-, and tri- grams are sufficient to capture the necessary information to describe the content of the citations.

# Appendix

## A1 Query ran in Scopus API

"TITLE-ABS-KEY ("Healthcare Logistics" OR "Healthcare Distribution" OR "Healthcare Optimization" OR "Healhcare Operations" OR "Healthcare Supply Chain" OR "Healthcare Planning" OR "Healthcare Simulation" OR "Healthcare Scheduling" OR "Healthcare Mathematical Modeling" OR "Hospital Logistics" OR "Hospital Distribution" OR "Hospital Optimization" OR "Hospital Operations" OR "Hospital Supply Chain" OR "Hospital Planning" OR "Hospital Simulation" OR "Hospital Scheduling" OR "Hospital Mathematical Modeling" OR "Patient Logistics" OR "Patient Distribution" OR "Patient Optimization" OR "Patient Operations" OR "Patient Supply Chain" OR "Patient Planning" OR "Patient Simulation" OR "Patient Scheduling" OR "Patient Mathematical Modeling" OR "Home healthcare")"

## References

Almeida, H., Meurs, M.-J., Kosseim, L., & Tsang, A. (2016). Data Sampling and Supervised Learning for HIV Literature Screening. IEEE Transactions on NanoBioscience, 15(4), 354–361. https://doi.org/10.1109/TNB.2016.2565481

Alpaydin, E. (2020). Introduction to machine learning. MIT Press.

Bannach-Brown, A., Przybyła, P., Thomas, J., Rice, A. S. C., Ananiadou, S., Liao, J., & Macleod, M. R. (2019). Machine learning algorithms for systematic review: reducing workload in a preclinical review of animal studies and reducing human screening error. Systematic Reviews, 8(1), 23. https://doi.org/10.1186/s13643-019-0942-7

Bekhuis, T., & Demner-Fushman, D. (2010). Towards automating the initial screening phase of a systematic review. Studies in Health Technology and Informatics, 160(Pt 1), 146–150.

Bekhuis, T., Tseytlin, E., Mitchell, K. J., & Demner-Fushman, D. (2014). Feature Engineering and a Proposed Decision-Support System for Systematic Reviewers of Medical Evidence. PLoS ONE, 9(1), e86277. https://doi.org/10.1371/journal.pone.0086277

Bishop, C. M. (2006). Pattern recognition and machine learning. Springer.

Carroll, J. B., Davies, P., Richman, B., & Davies, P. (1971). The American Heritage word frequency book. Houghton Mifflin Boston.

Cawley, M., Beardslee, R., Beverly, B., Hotchkiss, A., Kirrane, E., Sams, R., Varghese, A., Wignall, J., & Cowden, J. (2020). Novel text analytics approach to identify relevant literature for human health risk assessments: A pilot study with health effects of in utero exposures. Environment International, 134, 105228. https://doi.org/10.1016/j.envint.2019.105228

Cohen, A. M., Hersh, W. R., Peterson, K., & Yen, P.-Y. (2006). Reducing Workload in Systematic Review Preparation Using Automated Citation Classification. Journal of the American Medical Informatics Association, 13(2), 206–219. https://doi.org/10.1197/jamia.M1929

Dalal, S. R., Shekelle, P. G., Hempel, S., Newberry, S. J., Motala, A., & Shetty, K. D. (2013). A Pilot Study Using Machine Learning and Domain Knowledge to Facilitate Comparative Effectiveness Review Updating. Medical Decision Making, 33(3), 343–355. https://doi.org/10.1177/0272989X12457243

Dodge, H. F., & Romig, H. G. (1944). Sampling inspection tables: single and double sampling.

Frunza, O., Inkpen, D., Matwin, S., Klement, W., & O'Blenis, P. (2011). Exploiting the systematic review protocol for classification of medical abstracts. Artificial Intelligence in Medicine, 51(1), 17–25. https://doi.org/10.1016/j.artmed.2010.10.005

García Adeva, J. J., Pikatza Atxa, J. M., Ubeda Carrillo, M., & Ansuategi Zengotitabengoa, E. (2014). Automatic text classification to support systematic reviews in medicine. Expert Systems with Applications, 41(4), 1498–1508. https://doi.org/10.1016/j.eswa.2013.08.047

Gonzalez-Toral, S., Freire, R., Gualan, R., & Saquicela, V. (2019). A ranking-based approach for supporting the initial selection of primary studies in a Systematic Literature Review. 2019 XLV Latin American Computing Conference (CLEI), 1–10. https://doi.org/10.1109/CLEI47609.2019.235079

Gough, D., Oliver, S., & Thomas, J. (2007). An introduction to systematic reviews. Sage.

Hashimoto, K., Kontonatsios, G., Miwa, M., & Ananiadou, S. (2016). Topic detection using paragraph vectors to support active learning in systematic reviews. Journal of Biomedical Informatics, 62, 59–65. https://doi.org/10.1016/j.jbi.2016.06.001

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). An introduction to statistical learning: With applications in python. Springer Nature.

Jonnalagadda, S., & Petitti, D. (2013). A new iterative method to reduce workload in systematic review process. International Journal of Computational Biology and Drug Design, 6(1/2), 5. https://doi.org/10.1504/IJCBDD.2013.052198

Jurafsky, D., & Martin, J. H. (2014). Speech and Language Processing. Pearson Education.

Kim, S., & Choi, J. (2012). Improving the Performance of Text Categorization Models used for the Selection of High Quality Articles. Healthcare Informatics Research, 18(1), 18. https://doi.org/10.4258/hir.2012.18.1.18

Kim, S., & Choi, J. (2014). An SVM-based high-quality article classifier for systematic reviews. Journal of Biomedical Informatics, 47, 153–159. https://doi.org/10.1016/j.jbi.2013.10.005

Kontonatsios, G., Brockmeier, A. J., Przybyła, P., McNaught, J., Mu, T., Goulermas, J. Y., & Ananiadou, S. (2017). A semi-supervised approach using label propagation to support citation screening. Journal of Biomedical Informatics, 72, 67–76. https://doi.org/10.1016/j.jbi.2017.06.018

Kontonatsios, G., Spencer, S., Matthew, P., & Korkontzelos, I. (2020). Using a neural network-based feature extraction method to facilitate citation screening for systematic reviews. Expert Systems with Applications: X, 6, 100030. https://doi.org/10.1016/j.eswax.2020.100030

Kusa, W., Lipani, A., Knoth, P., & Hanbury, A. (2023). An analysis of work saved over sampling in the evaluation of automated citation screening in systematic literature reviews. Intelligent Systems with Applications, 18, 200193. https://doi.org/10.1016/j.iswa.2023.200193

Manning, C. D., & Schütze, H. (1999). Foundation of Statistical Natural Language Processing. The MIT Press.

Marshall, I. J., & Wallace, B. C. (2019). Toward systematic review automation: a practical guide to using machine learning tools in research synthesis. Systematic Reviews, 8(1), 163. https://doi.org/10.1186/s13643-019-1074-9

Matwin, S., Kouznetsov, A., Inkpen, D., Frunza, O., & O'Blenis, P. (2010). A new algorithm for reducing the workload of experts in performing systematic reviews. Journal of the American Medical Informatics Association, 17(4), 446–453. https://doi.org/10.1136/jamia.2010.004325

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). Distributed Representations of Words and Phrases and their Compositionality.

Miwa, M., Thomas, J., O'Mara-Eves, A., & Ananiadou, S. (2014). Reducing systematic review workload through certainty-based screening. Journal of Biomedical Informatics, 51, 242–253. https://doi.org/10.1016/j.jbi.2014.06.005

Mohri, M., Rostamizadeh, A., & Talwalkar, A. (2018). Foundations of Machine Learning. MIT Press.

Olorisade, B. K., Brereton, P., & Andras, P. (2019). The use of bibliography enriched features for automatic citation screening. Journal of Biomedical Informatics, 94, 103202. https://doi.org/10.1016/j.jbi.2019.103202

Olorisade, B. K., de Quincey, E., Brereton, P., & Andras, P. (2016). A critical analysis of studies that address the use of text mining for citation screening in systematic reviews. Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering, 1–11. https://doi.org/10.1145/2915970.2915982

O'Mara-Eves, A., Thomas, J., McNaught, J., Miwa, M., & Ananiadou, S. (2015). Using text mining for study identification in systematic reviews: a systematic review of current approaches. Systematic Reviews, 4(1), 5. https://doi.org/10.1186/2046-4053-4-5

Ouhbi, B., Kamoune, M., Frikh, B., Zemmouri, E. M., & Behja, H. (2016). A hybrid feature selection rule measure and its application to systematic review. Proceedings of the 18th International Conference on Information Integration and Web-Based Applications and Services, 106–114. https://doi.org/10.1145/3011141.3011177

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., . . . Moher, D. (2021). The PRISMA 2020 statement: an updated guideline for reporting systematic reviews. BMJ, n71. https://doi.org/10.1136/bmj.n71

Pennington, J., Socher, R., & Manning, C. (2014). Glove: Global Vectors for Word Representation. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1532–1543. https://doi.org/10.3115/v1/D14-1162

Ros, R., Bjarnason, E., & Runeson, P. (2017). A Machine Learning Approach for Semi-Automated Search and Selection in Literature Studies. Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering, 118–127. https://doi.org/10.1145/3084226.3084243

Rubio, T. R. P. M., & Gulo, C. A. S. J. (2016). Enhancing academic literature review through relevance recommendation: Using bibliometric and text-based features for classification. 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), 1–6. https://doi.org/10.1109/CISTI.2016.7521620

Schilling, E. G., & Neubauer, D. V. (2017). Acceptance Sampling in Quality Control,Third Edition. Chapman and Hall/CRC. https://doi.org/10.4324/9781315120744

Sellak, H., Ouhbi, B., & Frikh, B. (2015). Using rule-based classifiers in systematic reviews. Proceedings of the 17th International Conference on Information Integration and Web-Based Applications & Services, 1–5. https://doi.org/10.1145/2837185.2837279

Shalev-Shwartz, S., & Ben-David, S. (2014). Understanding machine learning: From theory to algorithms. Cambridge university press.

Shemilt, I., Simon, A., Hollands, G. J., Marteau, T. M., Ogilvie, D., O'Mara-Eves, A., Kelly, M. P., & Thomas, J. (2014). Pinpointing needles in giant haystacks: use of text mining to reduce impractical screening workload in extremely large scoping reviews. Research Synthesis Methods, 5(1), 31–49. https://doi.org/10.1002/jrsm.1093

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT Press.

Timsina, P., Liu, J., & El-Gayar, O. (2016). Advanced analytics for the automation of medical systematic reviews. Information Systems Frontiers, 18(2), 237–252. https://doi.org/10.1007/s10796-015-9589-7

Tranfield, D., Denyer, D., & Smart, P. (2003). Towards a Methodology for Developing Evidence-Informed Management Knowledge by Means of Systematic Review. British Journal of Management, 14(3), 207–222. https://doi.org/10.1111/1467-8551.00375

van Dinter, R., Tekinerdogan, B., & Catal, C. (2021). Automation of systematic literature reviews: A systematic literature review. Information and Software Technology, 136, 106589. https://doi.org/10.1016/j.infsof.2021.106589

Wallace, B. C., Small, K., Brodley, C. E., Lau, J., Schmid, C. H., Bertram, L., Lill, C. M., Cohen, J. T., & Trikalinos, T. A. (2012). Toward modernizing the systematic review pipeline in genetics: efficient updating via data mining. Genetics in Medicine : Official Journal of the American College of Medical Genetics, 14(7), 663–669. https://doi.org/10.1038/gim.2012.7

Wallace, B. C., Small, K., Brodley, C. E., Lau, J., & Trikalinos, T. A. (2012). Deploying an interactive machine learning system in an evidence-based practice center. Proceedings of the 2nd ACM SIGHIT Symposium on International Health Informatics - IHI '12, 819. https://doi.org/10.1145/2110363.2110464

Wallace, B. C., Small, K., Brodley, C. E., & Trikalinos, T. A. (2010). Active learning for biomedical citation screening. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '10, 173. https://doi.org/10.1145/1835804.1835829

Wallace, B. C., Trikalinos, T. A., Lau, J., Brodley, C., & Schmid, C. H. (2010). Semi-automated screening of biomedical citations for systematic reviews. BMC Bioinformatics, 11(1), 55. https://doi.org/10.1186/1471-2105-11-55

Wang, Z., Nayfeh, T., Tetzlaff, J., O'Blenis, P., & Murad, M. H. (2020). Error rates of human reviewers during abstract screening in systematic reviews. PLOS ONE, 15(1), e0227742-. https://doi.org/10.1371/journal.pone.0227742

Xie, B., & Salvendy, G. (2000). Review and reappraisal of modelling and predicting mental workload in single- and multi-task environments. Work & Stress, 14(1), 74–99. https://doi.org/10.1080/026783700417249

Yu, W., Clyne, M., Dolan, S. M., Yesupriya, A., Wulf, A., Liu, T., Khoury, M. J., & Gwinn, M. (2008). GAPscreener: an automatic tool for screening human genetic association literature in PubMed using the support vector machine technique. BMC Bioinformatics, 9, 205. https://doi.org/10.1186/1471-2105-9-205