Manuscript drafts

Gerbrich Ferdinands

22/04/2020

Keywords: Human-Machine interaction, active learning, systematic reviews, text classification

Introduction

Methods

Results

Discussion

Introduction

Systematic reviews are top of the bill in building evidence in research. A systematic review brings together all studies relevant to answer a specific research question [1]. Systematic reviews inform practice and policy [2] and are key in developing clinical guidelines [3]. However, systematic reviews are costly because they involve the manual screening of thousands of titles and abstracts, identifying publications relevant to answering the research question. An experienced reviewer takes on average 30 seconds to screen one title and abstract, whereas an inexperienced reviewer takes even longer [4]. Conducting a systematic review typically requires over a year of work by a team of researchers [5].

Moreover, systematic reviewers are often bound to a limited budget and timeframe. Currently, the demand for systematic reviews exceeds the available time and resources by far [6]. Especially when the need for guidelines is urgent - such as in the context of the current COVID-19 crisis - it is almost impossible to provide a review that is both timely and comprehensive. To ensure a timely review, reducing workload in systematic reviews is imperative.

With advances in Artificial Intelligence (AI), there has been wide interest in tools to reduce workload in systematic reviews [7]. Various learning models have been proposed, aiming to predict whether a given publication is relevant or irrelevant to the systematic review. Findings suggest that such models potentially reduce workload with 30-70% at the cost of losing 5% of relevant publications (95% recall) [8].

Screening prioritization is a well-established approach in increasing efficiency in title and abstract screening [9,10]. In screening prioritization, the learning model reorders publications to be screened by their likeliness to be relevant. The model presents the reviewer with the publications which are most likely to be relevant first, thereby expediting the process of finding all of the relevant publications. Such an approach allows for substantial time-savings in the screening process. Reviewing relevant publications early facilitates a faster transition of those publications to the next steps in the review process [9]. Additionally, several studies report increasing efficiency beyond saving time [8].

Recent studies have demonstrated the effectiveness of screening prioritization by means of active learning models [4,11,11–14]. Active learning is when the model can iteratively improve its predictions by allowing the model to choose the data from which it can learn [15]. Active learning has proven to be an efficient strategy in large datasets where labels are scarce, which makes title and abstract screening an ideal candidate for such models. When applied in screening prioritization, the reviewer screens publications that are presented by an active learning model. Subsequently, the active learning model learns from the reviewers' decision ('relevant', 'irrelevant') and uses this knowledge in selecting the next publication to be screened by the reviewer.

Although prior studies on the application of active learning models in title and abstract screening have addressed important issues [4,11,12,16], the complex nature of the field is making it difficult to draw overarching conclusions about best practice [8]. First, whilst there exists a scala of classification techniques, research to date has not yet evaluated techniques other than Support Vector Machines (SVM) [4,11,11,12]. Second, Miwa et al. [12] were the only study so far to make a direct comparison of model performance across research areass and found that active learning was more difficult on data from the social sciences compared to data from the medical sciences. The lack of replication on data from varying research contexts makes it impossible to draw conclusions about the general effectiveness of such techniques [8,17]. The question remains how active learning models for screening prioritization perform across different (1) classification techniques and (2) review contexts. Hence, additional evaluations of active learning models are required.

The purpose of the current paper is to increase the evidence base of active learning models for reducing workload in title and abstract screening in systematic reviews. Combining latest insights from this area, we propose seven different active learning models for the purpose of identifying relevant publications in systematic review datasets. The models were chosen to maximize the number of identified relevant publications, while minimizing the number of publications needed to screen. Working towards a general consensus in this emerging field, model performance was asessed by conducting a retrospective simulation on six systematic review datasets. Datasets were collected from the fields of medicine, software engineering, psychology, behavioural public administration, and virology to assess generalizability of the models across research contexts. The models, datasets and simulations are implemented in a pipeline of active learning for prioritization

screening, called `ASReview` [18]. `ASReview` is an open source and generic tool such that users can adapt and add modules as they like, encouraging fellow researchers to replicate findings from previous studies. All scripts and data used are openly published to facilitate usability and acceptability of AI-assisted title and abstract screening in the field of systematic review.

The remaining part of this paper is organized as follows. The Technical Details section will cover the workings of active learning models for study selection in systematic reviews on a conceptual level. The method section describes ... The results section reports ... The discusssion section summarises the findings, comments on them and summarises the main findings, discusses discuss limitations, draw conclusion and

Methods

Task Description

The screening process of a systematic review starts with all publications obtained in the search. The task is to identify which of these publications are relevant, by screening the publications at the title and abstract level. In active learning for screening prioritiztaion, the screening process proceeds as follows:

- Start with the set of all publications x with an unknown label, $\mathcal{U}$.
- The reviewer provides a label for a few publications $x \in \mathcal{U}$, creating an set of labeled publications $\mathcal{L}$. The publications can be labeled as relevant x_R or irrelevant x_I.
- The active learning cycle starts:
    1. A classifier is trained on the labeled publications, $C = \texttt{train}(\mathcal{L})$
    2. The classifier predicts labels for all unlabelled publications, $C(\mathcal{U})$
    3. Based on the predictions by C, the model selects the most relevant publication $x \in \mathcal{U}$
    4. The model asks the reviewer to screen this publication, $x_?$
    5. The reviewer screens the publication and provides a label, x_R or x_I
    6. The labeled publication is added to the training data, $\mathcal{L}$
    7. Back to step 1.
- In this active learning cycle, the model can incrementally improve its predictions on the remaining unlabeled publications. The relevant publications are identified as early in the process as possible. The reviewer and the model keep interacting until the reviewer decides to stop or until all publications in **U** have been labelled.

Technical details

A more detailed account of the active learning models is given in the following section. The structure and functions of the key components of the models will be introduced to clarify the choices made in the design of the current study.

Classification To make predictions on the unlabeled publications, a classifier is trained on features from the set of previously labeled publications. A technique widely used in classification tasks is the Support Vector Machine (SVM). SVMs separate the data into classes by finding a multidimensional hyperplane [19,20]. SVMs have been proven to be effective in active learning models for screening prioritization [[11]; Miwa2014]. Moreover, SVMs are the currently the only classifier implemented in ready-to-use software tools

implementing active learning for screening prioritization (Abstrackr [21], Colandr [22], FASTREAD [11], Rayyan [23], and RobotAnalyst [24]).

Whilst performance several classification techniques have been investigated in the AI-aided title and abstract screening field in general [SOURCE], the relatively new subfield of active learning for screening prioritization has not yet studied the performance of classifiers other than SVMs [4,11,11–14]. The current study aims to address this gap by exploring performance of three classifiers besides SVM:

- L2-regularized Logistic Regression (LR) models the probabilities describing the possible outcomes by a logistic function. The L2 penalty is imposed on the coefficients to reduce the number of features upon which the given solution is dependent [25].
- Naive Bayes (NB) is a supervised learning algorithm often used in text classification. Based on Bayes' Theorem, with the 'naive' assumption that all features are independent given the class value [26].
- Random Forests (RF) is a supervised learning algorithm where a large number of decision trees are fit on bootstrapped samples of the original data. All trees cast a vote on the class, which are aggregated into a class prediction for each instance [27].

These three classification techniques were selected because they are all widely adopted in previous studies on text classifcation [source]. Moreover, the amount of processing power is relatively low. They can run on a personal computer.

Class imbalance problem There are two classes in the dataset: relevant and irrelevant publications. Typically, only a fraction of the publications in the data belong to the relevant class. This poses a problem for training a classifier as there are far fewer examples of relevant than irrelevant publications to train on [8]. Moreover, classifiers are well-suited to separate data into classes, but not to correctly identifying one class [4]. This is evident in the case of a dataset where only one percent of publications are relevant. A model would achieve 99% accuracy when classifying all publications as irrelevant, even though none of the relevant papers would have been identified. Therefore, the class imbalance problem causes the classifier to miss relevant publications.

Previous studies have addressed the class imbalance problem by rebalancing the training data in different ways [8]. To decrease the class imbalance in the training data, the models in the current study rebalance the training set by Dynamic Supersampling (DS). DS decreases the number of irrelevant publications in the training data, whereas the number of relevant publications are increased (by copy) such that the size of the training data remains the same. The ratio between relevant and irrelevant publications is not fixed, but dynamically updated and depends on the size of the training data, the total number of publications, and the ratio between total number of relevant and irrelevant publications.

Word embeddings To predict publication class, the classifier uses information from the publications in the dataset. Examples of such information are titles and abstracts. However, a model cannot predict class from the titles and abstracts as they are; their textual content needs to be repressented numerically. The textual information needs to be mapped to feature vectors. This process of numerically representing textual content is called 'word embeddings'.

A classical example of word embeddings is a 'bag of words' (bow) representation. For each text in the data set, the number of occurrences of each word is stored. This leads to n features, where n is the number of distinct words in the texts [25]. The bag-of-words method is simplistic and will highly value often occuring but otherwise meaningless words such as "and". A more sophisticated approach is Term-frequency Inverse Document Frequency (TF-IDF). TF-IDF circumvents this problem by adjusting the term frequency in a text with the inverse docuement frequency, the frequency of a given word in the entire data set [28]. A downside of TF-IDF and other bow methods is that they do not take into account the ordering of the words, thereby ignoring semantics. An example of an approach that aims to overcome this weakness is Doc2Vec (D2V), capable of grasping the relations between words by learning to predict the words in the texts [29].

Miwa et al. found that active learning was more difficult on data from the social sciences compared to data the medical sciences and were able to link this difficulty to a natural difference in text complexity between these research areas [12]. As the study by Miwa et al. adopted a bow approach [12], we hypothesize that a more sophisticated word embeddings strategy has the potential to bridge the performance gap between these research areas.

Query strategy The active learning model can adopt different strategies in selecting the next publication to be screened by the reviewer. A strategy mentioned before is selecting the publication with the highest probability of being relevant. In the active learning literature this is referred to as certainty-based active learning [15]. Another well-known strategy is uncertainty-based active learning, where the instances that will be presented next will be those instances on which on which the model's classifications are the least certain, i.e. close to 0.5 probability [15]. Traditionally, this strategy trains the most accurate model because the model can learn the most from instances it is uncertain about.

Even though uncertainty-based active learning generally leads to a more accurate model in the end, certainty-based active learning is the preferred strategy for the task at hand. The main reason is that this strategy is far better suited to the goal of prioritizing the relevant publications. Moreover, a study comparing performance of both strategies in found that the accuracy gain of uncertainty-based screening was not significant [12]. Furthermore, uncertainty-based active learning is far better equipped at dealing with imbalanced data in active learning [30]. Therefore, certainty-based sampling is most equipped to the current scenario in which we are dealing with highly imbalanced datasets and where the goal is to identify all relevant publications as soon as possible.

Models

The seven models consist of the components described in the Technical Details section, adopting four different classification techniques and two different word embeddings approaches:

First, four models combining every classifier with TF-IDF word embeddings were investigated: - SVM + TF-IDF - NB + TF-IDF - RF + TF-IDF - SVM + TF-IDF

Second, the classifiers were combined with Doc2Vec word embeddings, leading to the following three models:[1]

- SVM + Doc2Vec
- RF + Doc2Vec
- SVM + Doc2Vec

Simulation study

For each of the seven models, performance was evaluated by simulating the model on the screening process of six systematic reviews. Performance of the seven models was evaluated by simulating their behaviour in the screening process of six systematic review datasets. Put differently, 42 simulations were carried out. For every model - dataset combination, hyperparameters were optimized[2]. To account for variance, every simulation was repeated for 15 trials. Simulations were run using ASReview's simulation mode [18]. There was no need for a human reviewer as the model could query the labels in the data instead.

Every simulation started with an initial training set of one relevant and one irrelevant publication to represent a 'worst case scenario' where the reviewer has minimal prior knowledge on the publications in the data. To account for bias, the initial training set was randomly sampled from the dataset for every of the 15 runs. Although varying over runs, the initial training sets were kept constant over datasets to allow for a direct comparison of models within datasets. A seed value was set to ensure reproducability. The classifier was

[1]The combination NB + D2V is impossible because NB can only process positive input features, whereas D2V
[2]see the Appendix for more information

retrained every time after a publication had been labeled. The simulation ended after all publications in the dataset had been labeled.

Datasets

The models were simulated on a convenience sample of six systematic review datasets. The data selection process was driven by two factors. Firstly, datasets were selected based on their background, given the need for datasets from diverse research areas. Secondly, datasets were selected by their availability, given the limited timespan of the current project. The datasets were retrieved from a collection of open systematic review datasets to be used for text mining purposes[3].

Three out of six datasets originated from the *medical sciences*: Ace, Wilson, and Virus. The Wilson dataset [31] is on a review on effectiveness and safety of treatments of Wilson Disease, a rare genetic disorder of copper metabolism [32]. The Ace dataset contains publications on the efficacy of Angiotensin-converting enzyme (ACE) inhibitors, a drug treatment for heart disease [33]. The Virus dataset is from a systematic review on studies that performed viral Metagenomic Next-Generation Sequencing (mNGS) in farm animals [34]. From the field of *software engineering*, the Software dataset contains publications from a review on fault prediction in source code [35]. The Nudging dataset [36] belongs to a systematic review on nudging healthcare professionals [37], stemming from the area of *behavioural public administration*, The PTSD dataset contains publications from the field of *psychology*. The corresponding systematic review is on studies applying latent trajectory analyses on posttraumatic stress after exposure to trauma [38]. Of these six datasets, Ace, and Software have been used for model simulations in previous studies on AI-aided title and abstract screening, respectively [33] and [11].

Data were preprocessed from their original source into a test dataset, containing title and abstract of the publications obtained in the initial search. Candidate studies with missing abstracts and duplicate instances were removed from the data. Test datasets were labelled to indicate which candidate studies were included in the systematic review, thereby indicating relevant publications. All test datasets consisted of thousands of candidate studies, of which only only a fraction was deemed relevant to the systematic review. Inclusion rates were centered around 1-2 percent with one outlier of about 5 percent (Table 1).

Table 1: Statistics on datasets from original systematic reviews.

	Original study			Test collection		
Dataset	Candidate studies	Relevant studies	Inclusion rate (%)	Candidate studies	Relevant studies	Inclusion rate (%)
Ace	2544	41	1.61	2235	41	1.83
Nudging	2006	100	4.99	1847	100	5.41
PTSD	6185	38	0.61	5031	38	0.76
Software	8911	104	1.17	8896	104	1.17
Virus	2481	120	4.84	2304	114	4.95
Wilson	3453	26	0.75	2333	23	0.99

Evaluating performance

Model performance was visualized by plotting recall curves and further assessed by three different measures, Work Saved over Sampling (WSS), Relevant References Found (RRF), and Average Time to Discovery (ATD). Results were averaged over 15 trials for every simulation.

Plotting recall as a function of the number of screened publications offers insight in model performance throughout the screening process [11,13]. The curves give informations in two directions. On the one hand they display the number of publications that need to be screened to achieve a certain level of recall (WSS),

[3]https://github.com/asreview/systematic-review-datasets, [18]

but on the other hand they present how many relevant publications are identified after screening a certain proportion of all publications (RRF).

WSS indicates the reduction in publications needed to be screened, at a given level of recall [33]. When measured at the typical recall level of 0.95 [33], WSS yields an estimate of the amount of work that can have be saved at the cost of failing to identify 5% of relevant publications. In the current study, WSS is computed at 0.95 and 1.00 recall level. RRF statistics are computed at 10%, representing the proportion of relevant studies that were identified after screening 10% of all publications.

Both RRF and WSS are sensitive to random effects as these statistics are strongly dependent on the position of the cutoff value. Moreover, WSS makes assumptions about acceptable recall levels whereas this might depend on the research question at hand [8]. A statistic that is not dependent on some arbitrary cutoff value is the ATD, which is the average number of publications that are screened to find a relevant publication, divided by the total number of publications in the data. The ATD is proportional to the area above the recall curve.

Analyses were carried out using R [39], version 3.6.1. All datasets accompanying the systematic reviews are openly published. This study was approved by the Ethics Committee of the Faculty of Social and Behavioural Sciences of Utrecht University, filed as an amendement under study 20-104. All simulations were run using through Cartesius (EINF-156).

Optimizing hyperparameters

Every model component contains hyperparameters, leading to a unique set of hyperparameters for each model. To maximize model performance, we need to find optimal values for the hyperparameters. For every model the optimal hyperparameter values are determined by optimizing on the data d. The hyperparameters are optimized by running several hundreds of optimization trials, in which hyperparameter values are sampled from their possible parameter space. A description of all hyperparameters and their sample space can be found in the appendix.

Maximum model performance is defined as the average time it takes to find an inclusion in the data, or more specific: the loss function minimizes the average number of papers needed to screen to find an inclusion (e.g. the area above the curve in the inclusion plot).

The optimization data d consists of (a subset from) the six systematic review datasets D mentioned above. Three different approaches in composing d are explored:

- **one**, where hyperparameters are optimized on only one of the six datasets, $d \in D$. Such hyperparameters are expected to lead to maximum performance in the same dataset d.
- **n**, where hyperparameters are optimized on all six data sets, $d = D$. This optimization approach intends to serve in producing the most optimal hyperparameters overall.
- **n-1**, where hyperparameters are optimized on all six datasets but one, $d \subset D$. Serving as a sensitivity analysis for the former condition, e.g. how sensitive are the hyperparamters. also as a cross-validation later on: hyperparameters obtained by training data, test data is never seen before. where d are all datasets but the one where we want to simulate later on. This results in $6 + 6 + 1 = 13$ sets of hyperparameters for every model.

Results were visually inspected to check if an optimum (minimal loss) has been reached. More trials were run if the loss still seemed to go down at a quick pace.

The hyperparameter values that were found to lead to a minimum loss value were visually inspected.

Background

[11], [11] simulated 32 svm classifiers, on software engineering. A popular classifier is SVM. succes with HUTM (fastread), uncertainty, mix of weighting and agressive undersampling, In terms of Yu et al, we adopt .CT.

SVM - tf-idf on medical data, uncertainty sampling, agressive undersampling. [4]

abstrackr svm certainty

SVM + Weighting + uncertainty (bow) produced good methods [12] Also include social sciences data besides medical data.

[33] perceptron-based classifier (neural network)

SVM on legal documents (no balancing, certainty) [13] in limitations section mentions that LR yields about same results, nb inferior results.

[40] - SVM, naive bayes, boosting and combinations. future work should optimize parameters. "Regarding the base classifiers used in identifying method- ologically rigorous studies, boosting consistently strikes the best balance between precision and recall, whereas naive Bayes in general performs well on recall (demonstrating a tradeoff between recall and precision), as does polynomial SVM on precision. The AUC results are mixed, although boosting has a slight edge overall. These results demonstrate that different classifiers can be used to satisfy different information needs (SVM for specificity, naive Bayes for sensitivity, and boosting for balance between the two, for example)."

Our extensions is that we try different classifiers, on more datasets.

When no balancing is applied, the training data set = labeled data set $\mathcal{L}$ The model can query the labels, who serve as the reviewer, active learning then perform active learning to detect inclusions.

a machine learning-based citation classification tool to reduce workload in systematic reviews of drug class efficacy. Using a perceptron classifier, WSS@95% = 56.61 in [33]. (5x2 crossvalidation). Can we beat this? The data

Openness, reproducible,

Benefits:

Adopting some sort of stopping criterion (outside scope of the current thesis) the reviewer can quit reviewing after having read only a fraction of candidate studies. meaning the screening process can be finished after reading a fraction of all candidate studies. Saving hours of time and resources.

paper organization:

The goal is to gain insight in classifiers other than the widely applied SVM, overall various research areas. So not only medical sciences.

Although considerable research has been devoted to ..., less attention has been paid to the comparison of different classifiers. Few studies have evaluated different classifiers in any systematic way different models to use haven't been investigated in a systematic way (only in software engineering)

1) the lack of replication of methods is making it impossible to draw any overall conclusions about best practice/ best approaches of the problem of reducing workload in screening.
2) screening prioritization is appealing to systematic reviewers because ...
3)
4) implemantation: usabliity and acceptability of such tools amongst researchers conducting a systematic review.

lack of studies investigating the effect of different methods over different research areas. (yu but only software engineering, miwa perhaps?) There is also a significant lack of examples outside of healthcare with the exception of one example in software engingeering

However, there is a need for consensus () and transparancy?

So far, however, there has been little discussion about the different classifiers to be used. Support Vector Machine has been the default in almost all studies.

Several algorithms to assist the reviewer in the abstract screening process have been proposed. (they are available in many shapes/sorts).

For truly evaluate the effectiveness of such methods,

Such a solution can save time and resources. Time saving is not the only benefit: and help to minimize bias..

Proposed solution This study is about how machine learning models can increase efficiency in systematic reviews. Their main objective is to identify ..

The strategy to reduce workload proposed in the current study is by prioritizing publications that are deemed most relevant to the systematic review. As the relevant publications are screened first, the reviewing process can be quit earlier.

The stage of abstract screening where abstracts are systematically screened is where a lot is to be gained. This stage is the target of possible learning algorithms that can assist the reviewer in selecting the relevant papers. Together with the reviewer /human machine interaction. The algorithm aims to compute which papers in the pool need to be excluded and which need to be included, based on the reviewers decisions. It learns from the reviewers decisions and asks the reviewer to provide more labels, incrementally improving its class predictions.

The goal of the algorithm defined in the current study is to reduce to number of abstracts needed to screen (maybe not right term, bit biomedical). To be more specific, the algorithm aims to present the reader with the primary studies as soon as possible. This means that at some point you probably have seen all relevant abstracts and are only viewing excluded papers, which means you can stop reviewing much earlier (theoretically spoken). Also reviewing is now much more fun. As compared to when you have to review all abstracts and you perhaps see only one relevant abstract every other week/day.

Background

... starts with a search for potentially relevant publications. This initial set of candidate studies need to be manually screened to identify the publications relevant for answering the research question. Because the initial set often consists of thousands of papers and the

To gather the findings relevant to answering the research question, a systematic search is performed. A systematic search starts with collecting all publications that meet pre-specified eligibility criteria. From this collection of candidate studies the researcher has to identify the publications relevant for answering the research question. Of all candidate studies only a fraction is relevant [...]. As more and more papers are published and reproducibility crisis has emerged, A systematic search often results in thousands of candidate studies. Relevant publications are then identified by screening title and abstract of all candidate studies. This screening process is a manual task often executed by multiple reviewers to ensure reliability. This is a time consuming process that weighs heavily on resources.

Most often the SVM classifier is used, popular and very good results. Also lots of other configurations. However, other classifiers have not been tested a lot (polygon thing by cohe, naïve bayes and random forest by ...), but mostly SVM still. Also, most research in the medical sciences (well there are some exceptions of course [conversation between cohen and matwill]

A solution is ...

It is important reflect on research by giving an overview of research areas which is typically done by a systematic review [...].

To review a specific research area, one starts out with an initial search of thousands of academic papers. All these papers abstracts need to be screened to find an initial batch of possibly relevant papers. With now hopefully only a couple of hundred papers left, the researcher needs to read these papers full-text to arrive at a final selection of papers that are relevant for the final systematic review [this is prisma process?]. This whole processes costs this and this much time [shelmilt].

---

active learning for systematic reviews

corpus = all the text:

Active learning = increasing classification performance with every query. The query strategy determines the way unlabeled papers are queried to the researcher.

[41]

Optimization results

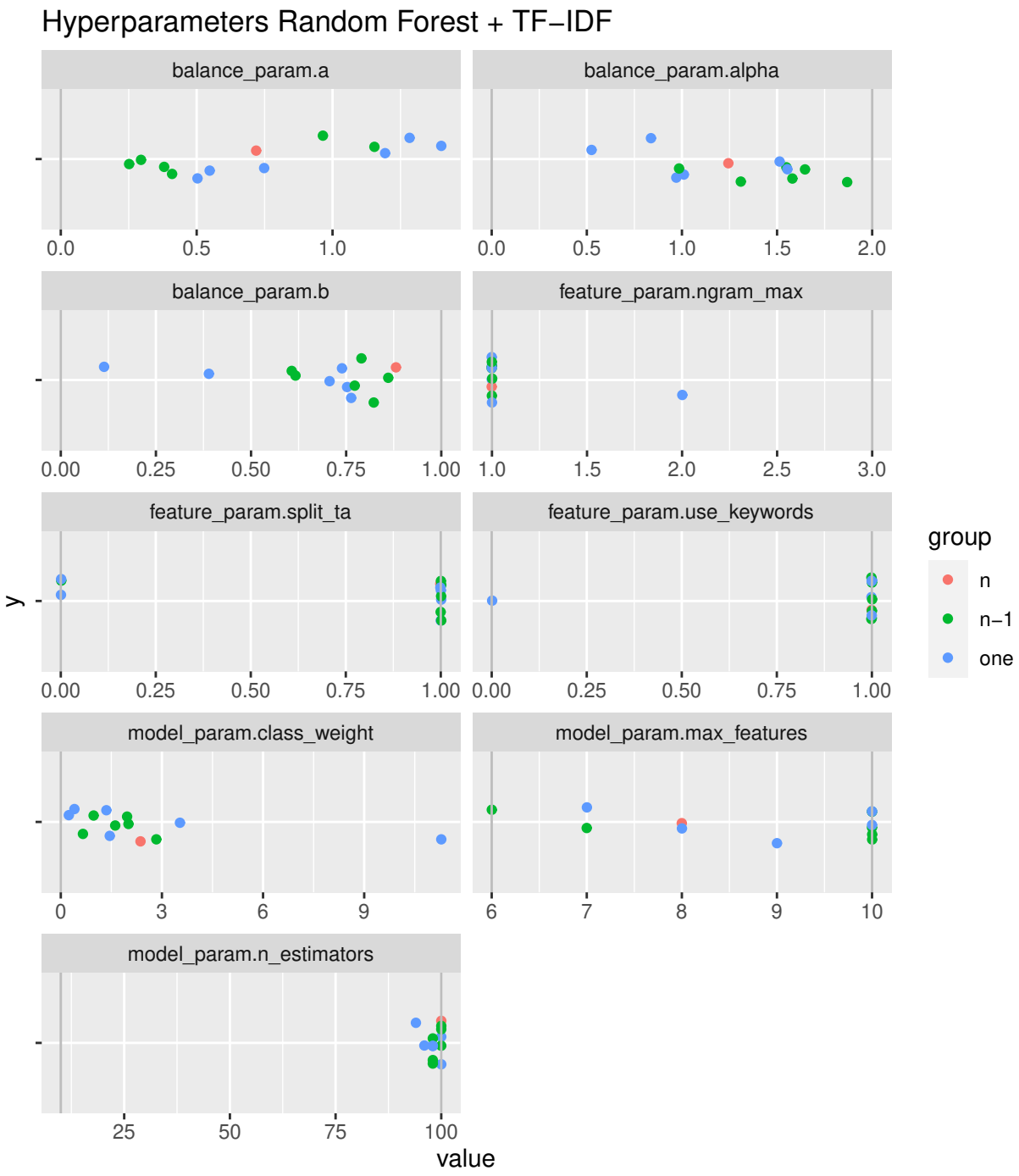
For every model, 13 sets of hyperparameters were optimized.

to include: plotting the loss reduction over trials

Optimal values of the hyperparameters were visually inspected.

As an example, the hyperparameters of the RF_TF-IDF model are presented in Figure 1. A panel displays the optimal values for a certain hyperparameter, where the blue colored dots represent the **one** condition, the green dots the **n-1** condition and the orange dot represents the optimal hyperparameter value when optmizing over all datasets (**n**). The x-axis represents the possible parameter space where the vertical grey lines mark the boundaries of the hyperparameter space (if possible). Note that the feature parameters ngram_max, split_ta, use_keywords and model parameters max_features and n_estimators are categorical.

Overall, the optimal values are distributed over the parameter space. Outlying values all belong to the **one** condition where optimization was dependent on one dataset only. Fore example, the class_weight parameter has an outlying value of 11.3 that belongs to the nudging dataset. (Why this is the case I can only speculate, but it is worth mentioning that this dataset has a relatively high inclusion rate of 5.41%, compared to the other datasets).



Appendix x - Hyperparameters and their sample space

Classifier hyperparameters

```
class_weight: normal(0,1) constrained to be > 0
class weight of the inclusions.
```

Logistic Regression

```
c: float. normal(0,1), constrained to be > 0
```

Support Vector Machine

```
gamma: ["auto", "scale"],
Gamma parameter of the SVM model.

C:
C parameter of the SVM model.

kernel:
SVM kernel type.["linear", "rbf", "poly", "sigmoid"]
```

Naive Bayes

```
"model_param.alpha", # exp(normal(0,1))
```

Random Forest

```
max_features: int (between 6 and 10)
    Number of features in the model.

n_estimators: int between 10 and 100
    Number of estimators.

model_param.n_estimators" #quniform(10,100,1)
```

Balance strategy hyperparameters: Dynamic supersampling:

```
a: float
    Governs the weight of the 1's. Higher values mean linearly more 1's
    in your training sample.
alpha: float
    Governs the scaling the weight of the 1's, as a function of the
    ratio of ones to zeros. A positive value means that the lower the
    ratio of zeros to ones, the higher the weight of the ones.
b: float
    Governs how strongly we want to sample depending on the total
    number of samples. A value of 1 means no dependence on the total
    number of samples, while lower values mean increasingly stronger
    dependence on the number of samples.
```

Feature extraction strategy hyperparameters

```
split_ta: 0 or 1
   whether titles and abstracts are split

use_keywords: 0 or 1
    whether keywords should be used
```

TF-IDF

```
ngram_max: 1, 2 or 3
    Can use up to ngrams up to ngram_max. For example in the case of
    ngram_max=2, monograms and bigrams could be used.
```

Doc2Vec

```
vector_size: int (between 32 and 127)
    Output size of the vector.

epochs: int (between 20 and 50)
    Number of epochs to train the doc2vec model.


min_count: int (between 1 and 3)
    Minimum number of occurences for a word in the corpus for it to be
    included in the model.

window: int (between 5 and 9)
    Maximum distance over which word vectors influence each other.

dm_concat: int 0 or 1
    Whether to concatenate word vectors or not.

dm: int
    Model to use.
    0: Use distribute bag of words (DBOW).
    1: Use distributed memory (DM).
    2: Use both of the above with half the vector size and concatenate them.

dbow_words: int 0 or 1
    Whether to train the word vectors using the skipgram method.
```

The software

ASReview takes the following parameters/arguments: We now have 75 combinations. for every for every model (5), for every dataset (5) and for every set of optimized hyperparameters (3), a simulation study consisting trials is performed. From these $5 * 5 * 3 = 75$ simulation studies, performance of the different models is evaluated.

	Configurations
Models	2-Layer Neural Network, Naive Bayes, Random Forest, Support Vector Machine, Logistic Regression
Query Strategies	Cluster Sampling, Maximum Sampling, Cluster * Maximum Sampling, Maximum * Uncertainty Sampling, Maximum * Random Sampling, Cluster * Uncertainty Sampling, Cluster * Random Sampling
Feature extraction strategies	Doc2Vec, TF-IDF, sbert, embeddingIdf

Use these inputs to predict relevance of papers.

Stage 1: hyperparameter optimization

Or, more specific:

Models	Feature extraction strategies
dense_nn	doc2vec
nb	tfidf
rf	tfidf
svm	doc2vec
lr	tfidf

Results

	ace	nudging	ptsd	software	virus	wilson
BCTD	0	0	0	0	0	0
RCTD	0	0	0	0	0	0
SCTD	0	0	0	0	0	0
LCTD	0	0	0	0	0	0
NCTD	0	0	0	0	0	0
BCDD	0	0	0	0	0	0
RCDD	0	0	0	0	0	0
SCDD	0	0	0	0	0	0
LCDD	0	0	0	0	0	0
NCDD	0	0	0	0	0	0

Discussion

- we look for final inclusions but we screen only the abstracts (do they satisfy the imformation need (blake (page 19 omara et evs)))

future research: - stopping rule is not discussed - computation/retraining time

strengths:

- open data
- different research areas
- different models on same dataset
- different datasets on same model

limitations

future research - all models save time, difficult to distinguish performance over datasets, especially when applied on a dataset of which no prior information is known (e.g. inclusions isn't known in practice). Perhaps go for other criteria like the fastest model, replicate study with computation time?

Simulating the title and abstract screening process, models are evaluted on their capability/speed of detecting the final inclusions. However, in a manual SR these final inclusions are selected after reading the fulltext. Information the text mining tool does not have. To truly assess the added value of such a tool, models should be evaluated on their capability of detecting the abstract inclusions. Call for systematic reviewers to openly publish need for open data containing abstract inclusions, not only final inclusions!

Appendix A - list of definitions

Feature Extraction Strategies

split_ta = overall hyperparameter

TF-IDF

hyperparameters

```
ngram_max: int
        Can use up to ngrams up to ngram_max. For example in the case of
        ngram_max=2, monograms and bigrams could be used.
```

Doc2Vec Predicts words from context. Aims at capturing the relations between word (man-woman, king-queen). [29]. Using a neural network.

using Continuous Bag-of-Words (CBOW), Skip-Gram model, ... Word vector W and extra: document vector D, trained to predict words in the text.

From gensim [42].

```
    Arguments
    ---------
    vector_size: int
        Output size of the vector.
    epochs: int
        Number of epochs to train the doc2vec model.
    min_count: int
        Minimum number of occurences for a word in the corpus for it to
        be included in the model.
    workers: int
        Number of threads to train the model with.
    window: int
        Maximum distance over which word vectors influence each other.
    dm_concat: int
        Whether to concatenate word vectors or not.
        See paper for more detail.
    dm: int
        Model to use.
        0: Use distribute bag of words (DBOW).
        1: Use distributed memory (DM).
        2: Use both of the above with half the vector size and concatenate
        them.
    dbow_words: int
        Whether to train the word vectors using the skipgram metho
```

SBERT BERT-base model with mean-tokens pooling [43]

embeddingIdf This model averages the weighted word vectors of all the words in the text, in order to get a single feature vector for each text. The weights are provided by the inverse document frequencies

Models

Naive Bayes Naive Bayes assumes all features are independent given the class value. [26]

ASReview uses the `MultinomialNB` from the scikit-learn package [25], that implements the naive Bayes algorithm for multinomially distributed data. `nb`

Hyperparameters

- alpha - accounts for features not present in learning samples and prevents zero probabilities in further computations.

Random Forests A number of decision trees are fit on bootstrapped samples of the original data, [27] RandomForestClassifier from sklearn

Arguments ——— n_estimators: int Number of estimators. max_features: int Number of features in the model. class_weight: float Class weight of the inclusions. random_state: int, RandomState Set the random state of the RNG. """

Support Vector Machine Arguments ——— gamma: str Gamma parameter of the SVM model. class_weight: class_weight of the inclusions. C: C parameter of the SVM model. kernel: SVM kernel type. random_state: State of the RNG.

Logistic Regression

Dense Neural Network

Query Strategies

- Max - Choose the most likely samples to be included according to the model
- Uncertainty - choose the most uncertain samples according to the model (i.e. closest to 0.5 probability) [44]
- Random - randomly selects abstracts with no regard to model assigned probabilities.
- Cluster - Use clustering after feature extraction on the dataset. Then the highest probabilities within random clusters are sampled

The following combinations are simulated:

- cluster
- max
- cluster * random
- cluster * uncertainty
- max * cluster
- max * random
- max * uncertainty

Balance Strategies

amount of training data

- n_instances = number of papers queried each query
- n_queries = number of queries
- n_prior_included: 5
- n_prior_excluded:

Combinations

This leads to 119 combinations of configurations.

- Naive bayes only goes with tfidf feature extraction.
- For the feature extraction strategies we will focus on doc2vec and tfidf. (but will compute all 4)
- This leads to 3 * 7 * 4 * 3 + 1 * 7 * 1 * 3 = 273 combinations.

See appendix A for a table containing all 273 combinations.

Cross-validation

Should give an accurate estimate of maximum performance / future systematic reviews to be performed.

Appendix B - combinations

Model	Query Strategy	Feature extraction strategy
dense_nn	cluster	doc2vec
dense_nn	max	doc2vec
dense_nn	max * cluster	doc2vec
dense_nn	max * uncertainty	doc2vec
dense_nn	max * random	doc2vec
dense_nn	cluster * uncertainty	doc2vec
dense_nn	cluster * random	doc2vec
dense_nn	cluster	tfidf
dense_nn	max	tfidf
dense_nn	max * cluster	tfidf
dense_nn	max * uncertainty	tfidf
dense_nn	max * random	tfidf
dense_nn	cluster * uncertainty	tfidf
dense_nn	cluster * random	tfidf
dense_nn	cluster	sbert
dense_nn	max	sbert
dense_nn	max * cluster	sbert
dense_nn	max * uncertainty	sbert
dense_nn	max * random	sbert
dense_nn	cluster * uncertainty	sbert

(continued)

Model	Query Strategy	Feature extraction strategy
dense_nn	cluster * random	sbert
dense_nn	cluster	embeddingIdf
dense_nn	max	embeddingIdf
dense_nn	max * cluster	embeddingIdf
dense_nn	max * uncertainty	embeddingIdf
dense_nn	max * random	embeddingIdf
dense_nn	cluster * uncertainty	embeddingIdf
dense_nn	cluster * random	embeddingIdf
nb	cluster	tfidf
nb	max	tfidf
nb	max * cluster	tfidf
nb	max * uncertainty	tfidf
nb	max * random	tfidf
nb	cluster * uncertainty	tfidf
nb	cluster * random	tfidf
rf	cluster	doc2vec
rf	max	doc2vec
rf	max * cluster	doc2vec
rf	max * uncertainty	doc2vec
rf	max * random	doc2vec
rf	cluster * uncertainty	doc2vec
rf	cluster * random	doc2vec
rf	cluster	tfidf
rf	max	tfidf
rf	max * cluster	tfidf
rf	max * uncertainty	tfidf
rf	max * random	tfidf
rf	cluster * uncertainty	tfidf
rf	cluster * random	tfidf
rf	cluster	sbert
rf	max	sbert
rf	max * cluster	sbert
rf	max * uncertainty	sbert
rf	max * random	sbert
rf	cluster * uncertainty	sbert
rf	cluster * random	sbert
rf	cluster	embeddingIdf
rf	max	embeddingIdf
rf	max * cluster	embeddingIdf
rf	max * uncertainty	embeddingIdf
rf	max * random	embeddingIdf
rf	cluster * uncertainty	embeddingIdf
rf	cluster * random	embeddingIdf
svm	cluster	doc2vec
svm	max	doc2vec
svm	max * cluster	doc2vec

(continued)

Model	Query Strategy	Feature extraction strategy
svm	max * uncertainty	doc2vec
svm	max * random	doc2vec
svm	cluster * uncertainty	doc2vec
svm	cluster * random	doc2vec
svm	cluster	tfidf
svm	max	tfidf
svm	max * cluster	tfidf
svm	max * uncertainty	tfidf
svm	max * random	tfidf
svm	cluster * uncertainty	tfidf
svm	cluster * random	tfidf
svm	cluster	sbert
svm	max	sbert
svm	max * cluster	sbert
svm	max * uncertainty	sbert
svm	max * random	sbert
svm	cluster * uncertainty	sbert
svm	cluster * random	sbert
svm	cluster	embeddingIdf
svm	max	embeddingIdf
svm	max * cluster	embeddingIdf
svm	max * uncertainty	embeddingIdf
svm	max * random	embeddingIdf
svm	cluster * uncertainty	embeddingIdf
svm	cluster * random	embeddingIdf
lr	cluster	doc2vec
lr	max	doc2vec
lr	max * cluster	doc2vec
lr	max * uncertainty	doc2vec
lr	max * random	doc2vec
lr	cluster * uncertainty	doc2vec
lr	cluster * random	doc2vec
lr	cluster	tfidf
lr	max	tfidf
lr	max * cluster	tfidf
lr	max * uncertainty	tfidf
lr	max * random	tfidf
lr	cluster * uncertainty	tfidf
lr	cluster * random	tfidf
lr	cluster	sbert
lr	max	sbert
lr	max * cluster	sbert
lr	max * uncertainty	sbert
lr	max * random	sbert
lr	cluster * uncertainty	sbert
lr	cluster * random	sbert
lr	cluster	embeddingIdf

(continued)

Model	Query Strategy	Feature extraction strategy
lr	max	embeddingIdf
lr	max * cluster	embeddingIdf
lr	max * uncertainty	embeddingIdf
lr	max * random	embeddingIdf
lr	cluster * uncertainty	embeddingIdf
lr	cluster * random	embeddingIdf

Appendix C - supercomputer Cartesius

500,000 SBU

Running on Cartesius is charged in System Billing Units (SBUs), and charging is based on the wall clock time of a job. On fat and thin nodes, an SBU is equal to using 1 core for 1 hour (a core hour), or 1 core for 20 minutes on a GPU node. Since compute nodes are allocated exclusively to a single job at a time, you will be charged for all cores on that node - even if you are using less.

In the current study, the classifier and the feature extraction strategy are varied, whereas the query and balance strategy remain fixed. In the current study only a fraction of all possible configurations are tested for the sake of brevity. There are many more options available and open to exploration.

References

1. PRISMA-P Group, Moher D, Shamseer L, Clarke M, Ghersi D, Liberati A, et al. Preferred reporting items for systematic review and meta-analysis protocols (PRISMA-P) 2015 statement. Systematic Reviews. 2015;4:1.

2. Gough D, Elbourne D. Systematic Research Synthesis to Inform Policy, Practice and Democratic Debate. Social Policy and Society. Cambridge University Press; 2002;1:225–36.

3. Chalmers I. The lethal consequences of failing to make full use of all relevant evidence about the effects of medical treatments: The importance of systematic reviews. Treating individualsfrom randomised trials to personalised medicine. Lancet; 2007. pp. 37–58.

4. Wallace BC, Trikalinos TA, Lau J, Brodley C, Schmid CH. Semi-automated screening of biomedical citations for systematic reviews. BMC Bioinformatics. 2010;11:55.

5. Borah R, Brown AW, Capers PL, Kaiser KA. Analysis of the time and workers needed to conduct systematic reviews of medical interventions using data from the PROSPERO registry. BMJ Open. British Medical Journal Publishing Group; 2017;7:e012545.

6. Lau J. Editorial: Systematic review automation thematic series. Systematic Reviews. 2019;8:70.

7. Harrison H, Griffin SJ, Kuhn I, Usher-Smith JA. Software tools to support title and abstract screening for systematic reviews in healthcare: An evaluation. BMC Medical Research Methodology. 2020;20:7.

8. O'Mara-Eves A, Thomas J, McNaught J, Miwa M, Ananiadou S. Using text mining for study identification in systematic reviews: A systematic review of current approaches. Systematic Reviews. 2015;4:5.

9. Cohen AM, Ambert K, McDonagh M. Cross-Topic Learning for Work Prioritization in Systematic Review Creation and Update. Journal of the American Medical Informatics Association. Oxford Academic; 2009;16:690–704.

10. Shemilt I, Simon A, Hollands GJ, Marteau TM, Ogilvie D, O'Mara-Eves A, et al. Pinpointing needles in giant haystacks: Use of text mining to reduce impractical screening workload in extremely large scoping reviews. Research Synthesis Methods. 2014;5:31–49.

11. Yu Z, Kraft NA, Menzies T. Finding better active learners for faster literature reviews. Empirical Software Engineering. Springer Science and Business Media LLC; 2018;23:3161–86.

12. Miwa M, Thomas J, O'Mara-Eves A, Ananiadou S. Reducing systematic review workload through certainty-based screening. Journal of Biomedical Informatics. 2014;51:242–53.

13. Cormack GV, Grossman MR. Evaluation of machine-learning protocols for technology-assisted review in electronic discovery. Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval. Gold Coast, Queensland, Australia: Association for Computing Machinery; 2014. pp. 153–62.

14. Cormack GV, Grossman MR. Autonomy and Reliability of Continuous Active Learning for Technology-Assisted Review. arXiv:150406868 [cs] [Internet]. 2015; Available from: http://arxiv.org/abs/1504.06868

15. Settles B. Active Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning. 2012;6:1–114.

16. Yu Z, Menzies T. FAST2: An intelligent assistant for finding relevant papers. Expert Systems with Applications. 2019;120:57–71.

17. Marshall IJ, Johnson BT, Wang Z, Rajasekaran S, Wallace BC. Semi-Automated evidence synthesis in health psychology: Current methods and future prospects. Health Psychology Review. Routledge; 2020;14:145–58.

18. van de Schoot R, de Bruin J, Schram R, Zahedi P, Kramer B, Ferdinands G, et al. ASReview: Active learning for systematic reviews. Zenodo; 2020;

19. Tong S, Koller D. Support vector machine active learning with applications to text classification. Journal of machine learning research. 2001;2:45–66.

20. Kremer J, Steenstrup Pedersen K, Igel C. Active learning with support vector machines. WIREs Data Mining and Knowledge Discovery. 2014;4:313–26.

21. Wallace BC, Small K, Brodley CE, Lau J, Trikalinos TA. Deploying an interactive machine learning system in an evidence-based practice center: Abstrackr. Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium. Miami, Florida, USA: Association for Computing Machinery; 2012. pp. 819–24.

22. Cheng SH, Augustin C, Bethel A, Gill D, Anzaroot S, Brun J, et al. Using machine learning to advance synthesis and use of conservation and environmental evidence. Conservation Biology. 2018;32:762–4.

23. Ouzzani M, Hammady H, Fedorowicz Z, Elmagarmid A. Rayyana web and mobile app for systematic reviews. Systematic Reviews. 2016;5:210.

24. Przybyła P, Brockmeier AJ, Kontonatsios G, Pogam M-AL, McNaught J, Erik von Elm, et al. Prioritising references for systematic reviews with RobotAnalyst: A user study. Research Synthesis Methods. 2018;9:470–88.

25. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. Journal of Machine Learning Research. 2011;12:2825–30.

26. Zhang H. The Optimality of Naive Bayes. Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004. 2004.

27. Breiman L. Random Forests. Machine Learning. 2001;45:5–32.

28. Ramos J, others. Using tf-idf to determine word relevance in document queries. Proceedings of the first instructional conference on machine learning. Piscataway, NJ; 2003. pp. 133–42.

29. Le QV, Mikolov T. Distributed Representations of Sentences and Documents. arXiv:14054053 [cs] [Internet]. 2014; Available from: http://arxiv.org/abs/1405.4053

30. Fu JH, Lee SL. Certainty-Enhanced Active Learning for Improving Imbalanced Data Classification. 2011 IEEE 11th International Conference on Data Mining Workshops. Vancouver, BC, Canada: IEEE; 2011. pp. 405–12.

31. Appenzeller-Herzog C. Data from Comparative effectiveness of common therapies for Wilson disease: A systematic review and meta-analysis of controlled studies. Zenodo; 2020.

32. Appenzeller-Herzog C, Mathes T, Heeres MLS, Weiss KH, Houwen RHJ, Ewald H. Comparative effectiveness of common therapies for Wilson disease: A systematic review and meta-analysis of controlled studies. Liver International. 2019;39:2136–52.

33. Cohen AM, Hersh WR, Peterson K, Yen P-Y. Reducing Workload in Systematic Review Preparation Using Automated Citation Classification. Journal of the American Medical Informatics Association : JAMIA. 2006;13:206–19.

34. Kwok KTT, Nieuwenhuijse DF, Phan MVT, Koopmans MPG. Virus Metagenomics in Farm Animals: A Systematic Review. Viruses. Multidisciplinary Digital Publishing Institute; 2020;12:107.

35. Hall T, Beecham S, Bowes D, Gray D, Counsell S. A Systematic Literature Review on Fault Prediction Performance in Software Engineering. IEEE Transactions on Software Engineering. 2012;38:1276–304.

36. Nagtegaal R, Tummers L, Noordegraaf M, Bekkers V. Nudging healthcare professionals towards evidence-based medicine: A systematic scoping review. Harvard Dataverse; 2019.

37. Nagtegaal R, Tummers L, Noordegraaf M, Bekkers V. Nudging healthcare professionals towards evidence-based medicine: A systematic scoping review. Journal of Behavioral Public Administration. 2019;2.

38. van de Schoot R, Sijbrandij M, Winter SD, Depaoli S, Vermunt JK. The GRoLTS-Checklist: Guidelines for reporting on latent trajectory studies. Structural Equation Modeling: A Multidisciplinary Journal. Routledge; 2017;24:451–67.

39. R Core Team. R: A language and environment for statistical computing. Vienna, Austria: R Foundation for Statistical Computing; 2019.

40. Kilicoglu H, Demner-Fushman D, Rindflesch TC, Wilczynski NL, Haynes RB. Towards Automatic Recognition of Scientifically Rigorous Clinical Research Evidence. Journal of the American Medical Informatics Association. 2009;16:25–31.

41. Danka T, Horvath P. modAL: A modular active learning framework for Python.

42. Řehůřek R, Sojka P. Software framework for topic modelling with large corpora. Proceedings of the LREC 2010 workshop on new challenges for NLP frameworks. Valletta, Malta: ELRA; 2010. pp. 45–50.

43. Reimers N, Gurevych I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:190810084 [cs] [Internet]. 2019; Available from: http://arxiv.org/abs/1908.10084

44. Lewis DD, Catlett J. Heterogeneous Uncertainty Sampling for Supervised Learning. In: Cohen WW, Hirsh H, editors. Machine Learning Proceedings 1994. San Francisco (CA): Morgan Kaufmann; 1994. pp. 148–56.