Manuscript drafts

Gerbrich Ferdinands

22/04/2020

Introduction

Methods

Results

Discussion

Introduction

Systematic reviews are top of the bill in building evidence in research. A systematic review attempts to bring together all studies relevant to answer a specific research question [24]. Systematic reviews inform practice and policy [10] and are key in developing clinical guidelines [5].

Systematic reviews are costly. Conducting a systematic review involves 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 [32]. Conducting a systematic review typically requires over a year of work from a team of researchers [3].

Systematic reviewers face competing demands. The production of a systematic review is a time-consuming process, but is often bound to a limited budget and timeframe. Currently, the demand for systematic reviews exceeds the available time and resources by far [14]. 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 [12]. More specifically, the field of AI-aided title and abstract screening is rapidly evolving. Various learning models have been proposed, aiming to detect 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) [22].

Screening prioritization is a well-established approach in increasing efficiency in title and abstract screening [7, 30]. In screening prioritization, the learning model reorders publications to be screened by their likeliness to be relevant, aiming to present the reviewer with the most relevant publications first. 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 [7].
Moreover, when combined with a stopping criterion, screening prioritization can be used to reduce the number of publications needed to screen. Additionally, several studies report increasing efficiency beyond reducing workload [22].

Recent studies have demonstrated the effectiveness of screening prioritization by means of active learning models [35, 33, 18]. Active learning is when the model can iteratively improve its predictions by allowing the model to choose the data from which it can learn [29]. Active learning has proven to be an efficient strategy in large datasets where labels are scarce, which makes identifying relevant publications an ideal candidate for such models.
When applied in the screening process, an active learning model can iteratively improve its relevancy predictions by interacting with the reviewer. The model decides which publication it wants the reviewer to screen next, who then provides the model with a label ('relevant', 'irrelevant'). In this active learning cycle, the model can incrementally improve its predictions on the relevancy of the remaining unlabeled publications. Based on previous labelling decisions by the reviewer, the model constantly reorders the remaining publications in the dataset.

Although the application of learning models in reducing workload of systematic reviews has been extensively studied, the complex nature of the field is making it difficult to draw overarching conclusions about best practice. Moreover, the lack of replication on data outside the biomedical sciences makes it impossible to draw conclusions about the general effectiveness of such technologies [22, 17]. Up to now only few studies have investigated active learning models for screening prioritization, with promising results. The question remains how different active learning models for screening prioritization perform across different review contexts. Hence, additional evaluations of active learning models are required.

The purpose of this 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 present a pipeline of active learning for prioritization screening, called `ASReview`. `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. Working towards a general consensus in this emerging field, this study implements various active learning models (in `ASReview`). Models are evaluated by performing a simulation on data from six existing systematic reviews from various research areas. All data presented in this study are openly published, hoping 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

Introduction paragraph: - (Research problem) Aim of the tool: detect sparse inclusions in SR dataset. - Aim of the study: evaluate performance several models across contexts. - means: simulation on existing SR datasets from various research areas.

The aim of the experiment was to assess performance of several active learning models on reducing workload in the screening process/study selection in systematic reviews. identifying relevant studies for systematic reviews by screening prioritization. Several active learning models where developed with the goal of identify the sparse relevant publications as early as possible / present for labelling.

To evaluate retrospective analysis. Models were demonstrated by simulating systematic reviews from various research areas. A retrospective study design: simulation study retrospective, various areas for generalizability

Performance of several machine learning models was demonstrated on six systematic review datasets. study design: retrospective

various datasets is necessary cause it is difficult to compare models accross datasets but also model performance is very data dependent!

To perform all these computations the research was carried out using the ASReview software by Utrecht University, which has a simulation mode that you can just input your labelled review file into and perform a simulation study with it. To be found on GitHub. It has many adjustable components/is very versatile.

Then if we still have time left we explore the effect of another feature extraction method, namely doc2vec. This might possibly increase performance as it performs better at grasping structure/hidden relations/hierarchy between words in the texts. So, it could be interesting in more 'fuzzy' research areas. A downside is that it takes more computing time.

Then if there's even more time we might want to compare a different balance strategy.

models The models who were tested: - Naive Bayes + TF-IDF - . . .

Simulation study: The screening process is simulated using ASReview, seeing if the original inclusions replicate.

- simulation mode
- assumptions (prior knowledge etc, oracle = reviewer).

Data selection & preprocessing Six systematic reviews from various research areas were used to simulate the automated systematic review process. Data were preprocessed from the original source into a test dataset, containing all publications obtained in the systematic search. The test datasets are labelled indicating the relevant and irrelevant publications. Using these labels, a computer simulation of an automated systematic review can be performed on the systematic review data.

The test datasets contain title information on all publications obtained in the search strategy. The instances in the data consist of a title and an abstract and were labeled to indicate which publications were included in the systematic review. Instances with missing abstracts and duplicate instances were removed from the data. The data preprocessing scripts can be found on the GitHub[1]

Cohen et al. collected systematic review datasets from the medical sciences [6]. All systematic reviews in this database are on drug efficacy. The *ace* dataset used in the current study comes from a systematic review on the efficacy of Angiotensin-converting enzyme (ACE) inhibitors.

The *software* dataset is retrieved from [34], who collected datasets on literature reviews from the software engineering field. This dataset is on fault prediction in software engineering by [11].

---

[1]https://github.com/GerbrichFerdinands/asreview-thesis

The *nudging* dataset comes from a review from the behavioural public administration area. The SR includes studies on nudging healthcare professionals [20]. The data was stored on the Harvard Dataverse [19].

A literature review from field of psychology, *ptsd*. The SR is on studies applying latent trajectory analyses on posttraumatic stress after exposure to trauma [28]. The corresponding data can be found on the Open Science Framework [. . .].

wilson, a dataset from the medical sciences [1]. TA review on effectiveness treatments of Wilson disease [2].

All datasets started with an initial pool of thousands of papers. A fraction of these papers where deemed relevant for the SR, with inclusion rates around 1-2 percent with one outlier of about 5 percent (Table 1).

Table 1: Statistics on datasets from original systematic reviews.

Dataset	Original study			Test collection		
	Candidate studies	Final inclusions	Inclusion rate (%)	Candidate studies	Final inclusions	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
wilson	3453	26	0.75	2333	23	0.99

How will results be analysed/performance will be evaluated?

how is 'good' performance defined/quantified?

metrics used:

Goal: evaluate performance of different models of the ASReview tool. What would happen if the citation screening would have been performed using asreview? All datasets accompanying the systematic reviews are openly published. We built several machine learning models to perform automated systematic reviews, who are then applied on existing systematic reviews.

automatically (by machine learning techniques) In the current study we focus on the effectiveness of different machine learning models in reducing workload in the citation screening process. Assisting the reviewer. Combining various machine learning techniques. Combining recent insights on, we propose ASReview.

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. Simulations were carried out using ASReview . . . , Statistical analyses were performed using R version . . . All simulations were run using through cartesius EINF-156

Models

x active learning models (M) were built. The models consist of multiple components, of which the classifier and the feature extraction strategy varied over the models.

Classifiers Every model applies a classifier c to predict the relevance of publications in the data. The classifier predicts the class of a publication in the dataset expediting a training dataset $\mathcal{L}$, predicts the class of an instance given input features $\mathbf{X}$

Logistic Regression (LR) - L2-regularized logistic regression. Logistic models the posterior directly, naive bayes has higher bias but lower variance, logistic might perform better when trainign set increases. [21] Naive Bayes (NB) - Naive Bayes assumes all features are independent given the class value. This is obviously not the case but still the algorithm performs impressively [36]. Especially at . . . tasks. Random Forests (RF) is 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 input $\mathbf{X}$ [4]. Support Vecor Machine (SVM) - finds a multidimensional hyperplane to separate classes. [31]

Word representation To predict the class of a publication class (e.g. whether a publication should be included or excluded), the classifier uses information from the publications in the dataset. Examples of such information are titles and abstracts. However, the classifier cannot predict the publication class from titles and abstracts as they are. Their textual content to needs to be mapped to feature vectors. This process of numerically representing textual content is called 'word embeddings'.

A classical example of word embeddings a 'bag of words' 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 [23]. The bag-of-words method is simplistic and will highly value often occuring but otherwise meaningless words such as "and". Term-frequency Inverse Document Frequency (TF-IDF) [25] circumvents this problem by adjusting a term frequency in a text with the inverse docuement frequency, the frequency of a given word in the entire data set.

The model expedits features of previously labeled publications and Based on features of the previously labeled inclusions and exclusions, the model

The features used are title and abstract from every publication. - this is what is prescribed by prisma, (check!)

(TF-IDF) (Doc2Vec)

Fixed components All models M apply the same query strategy of certainty sampling, in which the presented publication is always the one of which the model is most certain for it to be relevant.

To decrease the class imbalance in the training data, the model rebalances the training set by Dynamic Supersampling (DS). DS decreases the number of irrelevant papers in the training data, whereas the number of relevant papers are increased (by copy) such that the total number of samples remains the same. The ratio between relevant and irrelevant papers is not fixed, but dynamically updated and depends on the number of training samples, the total number of publications and the ratio between relevant and irrelevant publications.

Datasets

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

[34], [35] 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. [32]

abstrackr

SVM + Weighting + uncertainty (bow) produced good methods [18] Also include social sciences data besides medical data.

[6] perceptron-based classifier (neural network)

SVM on legal documents (no balancing, certainty) [8] in limitations section mentions that LR yields about same results, nb inferior results.

[13] - 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 [6]. (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].

A SR can be divided into phases. Everything starts with a **systematic search**, leading to then citation screening is performed, then full-text screening [24]

What must be the objective of our tool?: It is the tedious task citation screening part where loads of time can be saved.

models are designed in a 'realistic' way (you have some inclusions)

Selecting papers is a two-step process: abstract & fulltext screening

Binary classification problem. predict whether a paper in the pool is an inclusion or exclusion, based on labeled instances from $\mathcal{L}$ and use training data to understand how input variables are related to class.

We're building active learning model who takes X as an input an predicts class using labels from training set. Model improves by deciding asking more information from the reviewer (oracle), accounting for imbalance.

The researcher has some prior knowledge about the pool, some papers ought to be included in the SR.

Assumptions

1) decisions of the original SR are **ground truth** (benchmark) (oracle)

The inclusion rate is ... data is imbalanced. what is the philosophy False negatives must be avoided ... The cost of a false negative outweighs the cost of a false positive. Note that we assume the oracle/original user to hold the truth. This is of course not always the case.

There are two classes in the data: exlusions and inclusions. The inclusions are clearly the minority class.

Datasets from the medical and social sciences, software engineering and public administration. Medical sciences SRs are viewed as more 'strict'/'structured' and social sciences more messy.

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.

[9]

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).

Hyperparameters Random Forest + TF−IDF
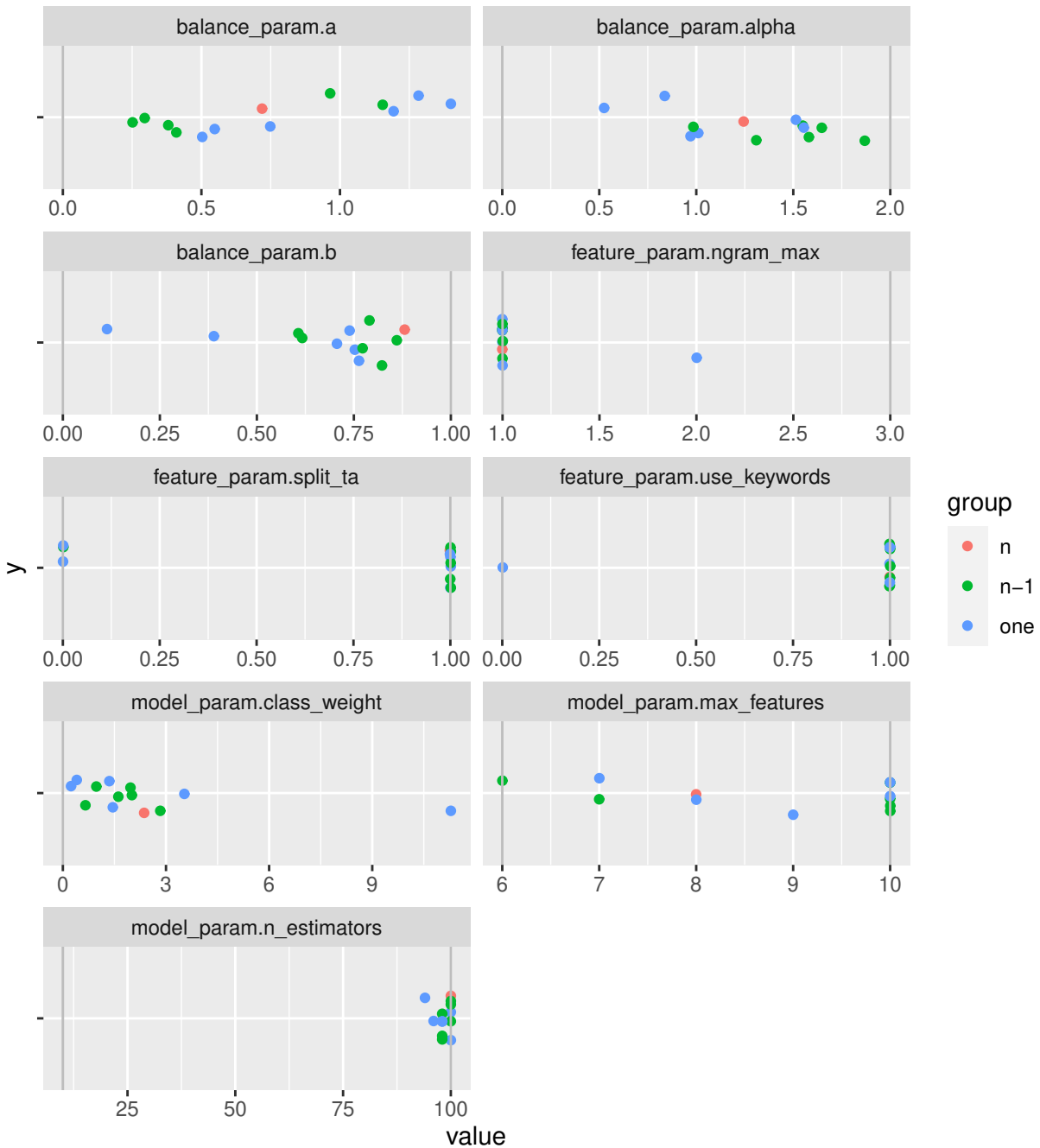
Appendix x - Hyperparameters and their sample space

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.
```

Simulations

To demonstrate the models, simulations are performed of the models on six original systematic reviews. With the labeled datasets, the systematic review can be reproduced but with now using a machine learning model to optimize the reviewing process.

The initial training set of the active learning model in the simulation is one prior included publication and one prior excluded publication, randomly sampled from the data. This demonstrates a 'worst case scenario' where the reviewer has minimal prior knowledge on publications in the data. For this simulations there is no need for a human reviewer since the labels in the data can be queried. The oracle is not the reviewer but the labels in the data. The model retrains every time after a label is provided by the oracle. The simulation ends when all publications in the dataset have been queried.

To account for variance, every simulation is repeated for 15 trials t.

Statistical analysis

Results are aggregated over 15 runs of every simulation.

measures

comparison over conditions

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

Performance metrics For each model, Several metrics are used to compare performance of different models over datasets,

Dataset	Naive Bayes	Random Forests	Support Vector Machine	Logistic Regression	Dense Neural Network
ptsd	?				
ace	?				
hall	?				
nagtegaal	?				
....	?				

The goal is twofold: we want to identify all relevant papers, as fast as we can. Tradeoff: identifying all relevant papers and reducing workload. A good metric to evaluate this is..

What is more important: recall or precision?

Recall more highly valued than precision.

What about class imbalance?

RRF Amount of relevant references found after having screened a certain percentage of the total number of abstracts.

Work saved over sampling (WSS) Indicates how much time can be saved, at a given level of recall. WSS is in terms of the percentage of abstracts that don't have to be screened by the researcher. Typically, WSS is measured at a recall of 0.95. Reasonable because..

$$\texttt{WSS} = \frac{TN + FN}{N} - (1 - recall)$$

Raoul

Utility?

F-measure

ROC/AUC Is performance related to some characteristic (n, inclusion rate, . . .)

? How to compare outcomes of 3 different optimization strategies?

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). [15]. 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 [26].

```
    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 [27]

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. [36]

ASReview uses the `MultinomialNB` from the scikit-learn package [23], 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, [4] 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) [16]
- 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

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] Christian Appenzeller-Herzog. *Data from Comparative Effectiveness of Common Therapies for Wilson Disease: A Systematic Review and Meta-analysis of Controlled Studies.* Zenodo, Jan. 2020.

[2] Christian Appenzeller-Herzog et al. "Comparative Effectiveness of Common Therapies for Wilson Disease: A Systematic Review and Meta-Analysis of Controlled Studies". en. In: *Liver International* 39.11 (2019), pp. 2136–2152. ISSN: 1478-3231. DOI: 10.1111/liv.14179.

[3] Rohit Borah et al. "Analysis of the Time and Workers Needed to Conduct Systematic Reviews of Medical Interventions Using Data from the PROSPERO Registry". en. In: *BMJ Open* 7.2 (Feb. 2017), e012545. ISSN: 2044-6055, 2044-6055. DOI: 10/f9tf57.

[4] Leo Breiman. "Random Forests". en. In: *Machine Learning* 45.1 (Oct. 2001), pp. 5–32. ISSN: 1573-0565. DOI: 10.1023/A:1010933404324.

[5] Iain Chalmers. "The Lethal Consequences of Failing to Make Full Use of All Relevant Evidence about the Effects of Medical Treatments: The Importance of Systematic Reviews". In: *Treating Individuals—from Randomised Trials to Personalised Medicine.* Lancet, 2007, pp. 37–58.

[6] A.M. Cohen et al. "Reducing Workload in Systematic Review Preparation Using Automated Citation Classification". In: *Journal of the American Medical Informatics Association : JAMIA* 13.2 (2006), pp. 206–219. ISSN: 1067-5027. DOI: 10.1197/jamia.M1929.

[7] Aaron M. Cohen, Kyle Ambert, and Marian McDonagh. "Cross-Topic Learning for Work Prioritization in Systematic Review Creation and Update". en. In: *Journal of the American Medical Informatics Association* 16.5 (Sept. 2009), pp. 690–704. ISSN: 1067-5027. DOI: 10/c3shq2.

[8] Gordon V. Cormack and Maura R. Grossman. "Evaluation of Machine-Learning Protocols for Technology-Assisted Review in Electronic Discovery". In: *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval.* SIGIR '14. Gold Coast, Queensland, Australia: Association for Computing Machinery, July 2014, pp. 153–162. ISBN: 978-1-4503-2257-7. DOI: 10.1145/2600428.2609601.

[9]   Tivadar Danka and Peter Horvath. "modAL: A Modular Active Learning Framework for Python". In: ().

[10]   David Gough and Diana Elbourne. "Systematic Research Synthesis to Inform Policy, Practice and Democratic Debate". en. In: *Social Policy and Society* 1.3 (July 2002), pp. 225–236. ISSN: 1475-3073, 1474-7464. DOI: 10/bdmp7h.

[11]   Tracy Hall et al. "A Systematic Literature Review on Fault Prediction Performance in Software Engineering". In: *IEEE Transactions on Software Engineering* 38.6 (Nov. 2012), pp. 1276–1304. ISSN: 2326-3881. DOI: 10.1109/TSE.2011.103.

[12]   Hannah Harrison et al. "Software Tools to Support Title and Abstract Screening for Systematic Reviews in Healthcare: An Evaluation". In: *BMC Medical Research Methodology* 20.1 (Jan. 2020), p. 7. ISSN: 1471-2288. DOI: 10.1186/s12874-020-0897-3.

[13]   H. Kilicoglu et al. "Towards Automatic Recognition of Scientifically Rigorous Clinical Research Evidence". en. In: *Journal of the American Medical Informatics Association* 16.1 (Jan. 2009), pp. 25–31. ISSN: 1067-5027, 1527-974X. DOI: 10/bjkhh9.

[14]   Joseph Lau. "Editorial: Systematic Review Automation Thematic Series". In: *Systematic Reviews* 8.1 (Mar. 2019), p. 70. ISSN: 2046-4053. DOI: 10/ggsmwf.

[15]   Quoc V. Le and Tomas Mikolov. "Distributed Representations of Sentences and Documents". en. In: *arXiv:1405.4053 [cs]* (May 2014). arXiv: 1405.4053 `[cs]`.

[16]   David D. Lewis and Jason Catlett. "Heterogeneous Uncertainty Sampling for Supervised Learning". en. In: *Machine Learning Proceedings 1994*. Ed. by William W. Cohen and Haym Hirsh. San Francisco (CA): Morgan Kaufmann, Jan. 1994, pp. 148–156. ISBN: 978-1-55860-335-6. DOI: 10.1016/B978-1-55860-335-6.50026-X.

[17]   Iain J. Marshall et al. "Semi-Automated Evidence Synthesis in Health Psychology: Current Methods and Future Prospects". In: *Health Psychology Review* 14.1 (2020), pp. 145–158. DOI: 10/ggjv98. eprint: https://doi.org/10.1080/17437199.2020.1716198.

[18]   Makoto Miwa et al. "Reducing Systematic Review Workload through Certainty-Based Screening". en. In: *Journal of Biomedical Informatics* 51 (Oct. 2014), pp. 242–253. ISSN: 1532-0464. DOI: 10.1016/j.jbi.2014.06.005.

[19]   Rosanna Nagtegaal et al. *Nudging Healthcare Professionals towards Evidence-Based Medicine: A Systematic Scoping Review.* Version V1. tyep: dataset. 2019.

[20]   Rosanna Nagtegaal et al. "Nudging Healthcare Professionals towards Evidence-Based Medicine: A Systematic Scoping Review". In: *Journal of Behavioral Public Administration* 2.2 (2019). DOI: doi.org/10.30636/jbpa.22.71.

[21]   Andrew Y. Ng and Michael I. Jordan. "On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes". In: *Advances in Neural Information Processing Systems 14*. Ed. by T. G. Dietterich, S. Becker, and Z. Ghahramani. MIT Press, 2002, pp. 841–848.

[22]   Alison O'Mara-Eves et al. "Using Text Mining for Study Identification in Systematic Reviews: A Systematic Review of Current Approaches". In: *Systematic Reviews* 4.1 (Jan. 2015), p. 5. ISSN: 2046-4053. DOI: 10.1186/2046-4053-4-5.

[23]   F. Pedregosa et al. "Scikit-Learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[24]   PRISMA-P Group et al. "Preferred Reporting Items for Systematic Review and Meta-Analysis Protocols (PRISMA-P) 2015 Statement". en. In: *Systematic Reviews* 4.1 (Dec. 2015), p. 1. ISSN: 2046-4053. DOI: 10.1186/2046-4053-4-1.

[25]   Juan Ramos et al. "Using Tf-Idf to Determine Word Relevance in Document Queries". In: *Proceedings of the First Instructional Conference on Machine Learning*. Vol. 242. Piscataway, NJ. 2003, pp. 133–142.

[26] Radim Řehůřek and Petr Sojka. "Software Framework for Topic Modelling with Large Corpora". English. In: *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta: ELRA, May 2010, pp. 45–50.

[27] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks". In: *arXiv:1908.10084 [cs]* (Aug. 2019). arXiv: 1908.10084 [cs].

[28] Rens van de Schoot et al. "The GRoLTS-Checklist: Guidelines for Reporting on Latent Trajectory Studies". In: *Structural Equation Modeling: A Multidisciplinary Journal* 24.3 (2017), pp. 451–467. DOI: 10/gdpcw9. eprint: https://doi.org/10.1080/10705511.2016.1247646.

[29] Burr Settles. "Active Learning". en. In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 6.1 (June 2012), pp. 1–114. ISSN: 1939-4608, 1939-4616. DOI: 10 . 2200 / S00429ED1V01Y201207AIM018.

[30] Ian Shemilt et al. "Pinpointing Needles in Giant Haystacks: Use of Text Mining to Reduce Impractical Screening Workload in Extremely Large Scoping Reviews". en. In: *Research Synthesis Methods* 5.1 (2014), pp. 31–49. ISSN: 1759-2887. DOI: 10.1002/jrsm.1093.

[31] Simon Tong and Daphne Koller. "Support Vector Machine Active Learning with Applications to Text Classification". In: *Journal of machine learning research* 2.Nov (2001), pp. 45–66.

[32] Byron C. Wallace et al. "Semi-Automated Screening of Biomedical Citations for Systematic Reviews". In: *BMC Bioinformatics* 11.1 (Jan. 2010), p. 55. ISSN: 1471-2105. DOI: 10.1186/1471-2105-11-55.

[33] Zhe Yu, Nicholas Kraft, and Tim Menzies. "Finding Better Active Learners for Faster Literature Reviews". In: *Empirical Software Engineering* (Mar. 2018). DOI: 10.1007/s10664-017-9587-0.

[34] Zhe Yu, Nicholas A. Kraft, and Tim Menzies. "Finding Better Active Learners for Faster Literature Reviews". In: *Empirical Software Engineering* 23.6 (Mar. 2018), pp. 3161–3186. ISSN: 1573-7616. DOI: 10.1007/s10664-017-9587-0.

[35] Zhe Yu and Tim Menzies. "FAST2: An Intelligent Assistant for Finding Relevant Papers". en. In: *Expert Systems with Applications* 120 (Apr. 2019), pp. 57–71. ISSN: 0957-4174. DOI: 10.1016/j.eswa. 2018.11.021.

[36] Harry Zhang. "The Optimality of Naive Bayes". In: *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2004*. Vol. 2. Jan. 2004.