

Machine learning functionality in EPPI-Reviewer

Introduction

The capability of automation systems to improve review workflow has been growing over recent years. There are four ways in which 'machine learning' can be used in EPPI-Reviewer to make reviewing more efficient:

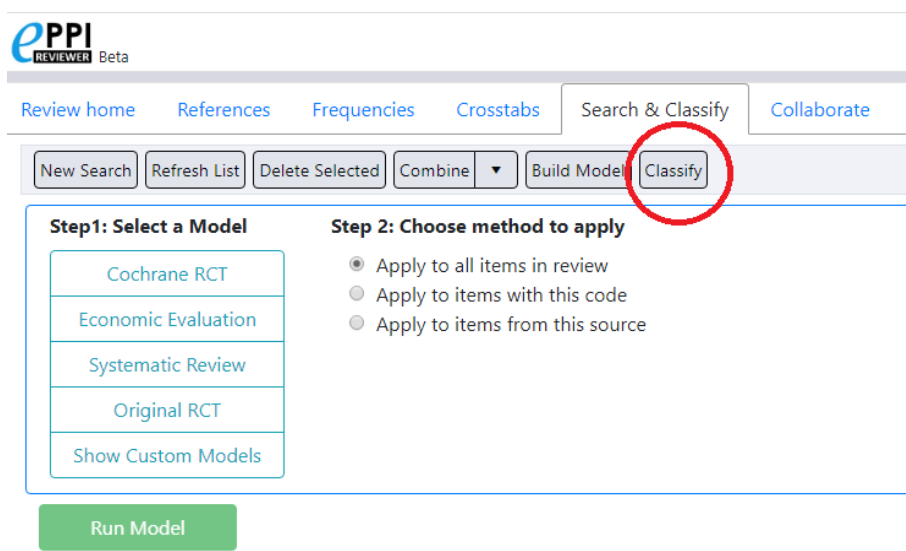
1. Using pre-built classification for particular study types
2. Building your own bespoke classifiers
3. Using 'priority screening' to order records for manual screening
4. Automatically clustering records based on the text they contain

The first three of these tools are described below, with most attention given to 'priority screening' later in the document. Please see the main user manual under 'text mining' for information on automatic clustering.

N.B. This document contains information for using the web interface for using the pre-built and custom classifiers. There is another document available with screenshots depicting the same functions using the Silverlight interface.

1. Pre-built study type classifiers

These classifiers can be accessed from the icon depicted below on the right-hand-side 'codes' panel in EPPI-Reviewer. They are not yet available in version 5 (deployed internally at NICE).



EPPI REVIEWER Beta

Review home References Frequencies Crosstabs Search & Classify Collaborate

New Search Refresh List Delete Selected Combine Build Model Classify

Step 1: Select a Model

- Cochrane RCT
- Economic Evaluation
- Systematic Review
- Original RCT
- Show Custom Models

Step 2: Choose method to apply

- ☒ Apply to all items in review
- ☐ Apply to items with this code
- ☐ Apply to items from this source

Run Model

These pre-built classifiers have 'learned' from thousands of records and so are highly accurate within the domains that they have been designed for. The records on which they have been built – and so the ones which they will work best for – are biomedical records of human studies, such as randomised controlled trial records found on PubMed. It is important to bear in mind that these classifiers were built in the health domain and so will not necessarily work well in, for example, education.

They can be used simply by selecting the classifier on the classifier window (Cochrane RCT, economic evaluation, systematic review, or original RCT), and specifying on the right whether all the records in the review should be classified, or only a subset. After 'apply model' is clicked, the records are uploaded to the machine learning server and scored. This can take two or three minutes.

The RCT Classifiers

There are two RCT classifiers, both of which were built using 280,000 records manually classified by Cochrane Crowd.[1] Those users who have used the system for a number of years will be familiar with the 'original RCT' classifier. The 'Cochrane RCT' classifier was recently added. This is built from the same data, but is an ensemble of two classifiers, which gives it slightly better performance for some tasks. It has been calibrated on the McMaster

‘Hedges’ dataset (a dataset of 49,000 records that has been used to validate many search filters) to achieve a 99% recall. When you use this classifier, two sets of records are returned, rather than the usual single set. One set is the records which are highly unlikely to be RCTs, and the other set is the set of records that *may* be RCTs.

While the two classifiers operate in very similar ways, and produce almost identical results in terms of the relative likelihood that a given record does, or does not, describe an RCT, the scores they produce are not directly comparable. For example, on the original RCT classifier, early evaluation suggested that RCTs were very unlikely to score less than 10, and we used this as a rough cut-off for some purposes. The scores that are obtained from the Cochrane RCT classifier are closer to genuine probabilities though, so 10% of RCTs may well score less than 10. The cut-off score that is used for this classifier is actually 0.24 (which is rounded to 0 in the user interface). Those items in the ‘may be RCTs’ list score above this threshold, and those in the other list score below. Thus, the classifier has been designed to be used according to the two lists of results returned, rather than by interacting with the scores.

Which RCT classifier should you use? It depends on your use scenario. If you want to follow the Cochrane ‘rule’ for determining which records should, and might not, be looked at, you should use the Cochrane RCT classifier, and look at all records that the classifier determines *may* describe RCTs. If you would like to interact with the scores returned – and set a threshold for yourself – you are probably better off using the ‘original RCT’ classifier.

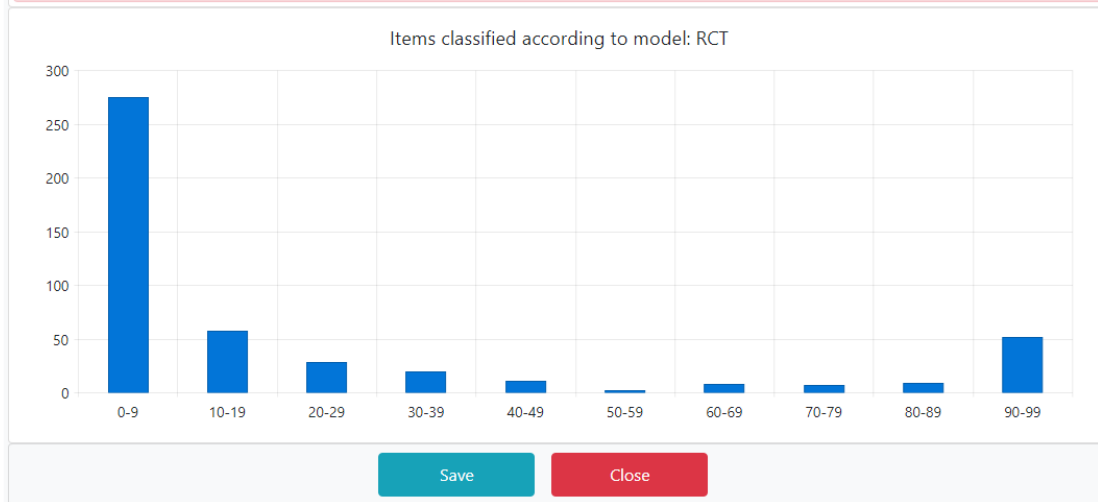
Once the machine learning has been completed, the results are available in the ‘search’ screen. Use the ‘refresh’ button periodically to check to see if results are available.

| | No | Name | Created By | Date | Hits | Classifier |
|--------------------------|----|--|--------------|-----------|------|------------|
| <input type="checkbox"/> | 38 | Items classified according to model: RCT | James Thomas | 4/11/2019 | 467 | true |

Clicking the ‘true’ link will show in graphical form the distribution of scores from the classifier (on the x axis) with the number of records in each decile shown on the y axis. These items can be assigned to ten codes using the ‘Create’ button at the top of the window. The screenshot below shows a fairly typical output of the classifier. It is confident that the records ranked 90-99 are very likely to be RCTs, and that those ranked 0-9 are not. There are a smaller number of records in the middle of which it is less certain.

[New Search](#) [Refresh List](#) [Delete Selected](#) [Combine](#) [Build Model](#) [Classify](#)

To represent the classifier result as codes, select a location in a coding tool to the right.



Clicking the number of hits on the search screen (previous graphic) will list the search results as usual. Click 'view options' to select and display the 'score' column, which shows you what score a given item was given.

[Import Items](#) [With this Code](#) [Assign Code](#) [Cluster](#) [Coding Report](#) [Export to RIS](#)

First Previous Page: 1 of 5 Next Last Showing 100 items of 467 [Close Options](#)

List Options:

- ☒ Document ID
- ☒ Short title
- ☒ Title
- ☐ Journal
- ☐ Info

Page size: 100

- ☐ Your document ID
- ☐ Authors
- ☒ Year
- ☐ Document type
- ☒ Score

Showing Items classified according to model: RCT

| <input type="checkbox"/> | ID | Short title† | Title | Year | Score |
|--------------------------|----------------------------|--------------|--|------|-------|
| GO | <input type="checkbox"/> I | 40772015 | [Treatment of... (2004)] | 2004 | 46 |
| GO | <input type="checkbox"/> I | 40772172 | A study to compare oral sumatriptan with oral aspirin plus oral metoclopramide in the acute treatment of migraine. The Oral Sumatriptan and Aspirin plus Metoclopramide Comparative Study Group. | 1992 | 98 |
| GO | <input type="checkbox"/> I | 40772132 | Adelman (1995) | 1995 | 45 |

The systematic reviews and economic evaluation classifiers

The economic evaluation and systematic reviews classifiers are available with thanks to the University of York. They have been built from two databases which are no longer updated: NHS EED (economic evaluation) and DARE (systematic reviews of effectiveness). [Information about these databases can be found here](#). The systematic reviews

classifier was built with a set of records supplied by CRD, University of York which were used when creating the DARE database. It consists of 37,109 systematic reviews included in DARE, and 273,968 records that were considered for inclusion, but manually excluded. The economic evaluations classifier was built from 17,615 economic evaluations included in NHS EED and 163,451 records that were considered for inclusion, but manually excluded.


It is important to note that, while they have both been built using large quantities of high-quality data, they have not received the same degree of evaluation as the RCT Classifiers, and we do not have other datasets on which to validate their performance. We think they can be useful to order a batch of records according to the probability that they describe systematic reviews or economic evaluations. Precise cut-off scores will vary according to domain and user needs though, so we recommend that users either: a) use the classifiers for providing a ranked list of records that can then be viewed manually; or b) carry out an evaluation using their own data in order to determine an appropriate cut-off score.

The operation of the two classifiers follows the same stages as outlined above for the RCT classifiers.

2. Building bespoke classifiers

The 'search & classify' screen as used above to score items using pre-built classifiers can be used to build new classifier models based on user-entered data. Click 'build model' to open the model building screen and then follow this process.

First, to build your own model you need to have first classified some 'training' records into two classes, with one code for each class. You can then use the 'build model' screen to build your model. In the example below we have data which have codes denoting those that were included on title & abstract, and those that were excluded on 'target group'; we are about to build a model to distinguish between them.



Build Model

[Feedback](#) [Help](#) James Thomas [Logout](#)

Learn to apply this code:

INCLUDE on title & abstract ▼

Distinguish from this code:

EXCLUDE on target group ▼

Name for your model:

My test model

Build Model

Refresh Models

☐

Title

Att On

Att Not On

Accuracy

AUC

Precision

Recall

☐

Aspirin vs paracet

Aspirin

Paracetamol


0.938

0.974

0.932

0.932

Once 'build model' has been clicked, the data are uploaded to the machine learning server and the model is built. This can take a number of minutes as it is a resource- and memory-intensive process. Once the model has been built, it appears in the list of models, and is available for use in the same way that the pre-built classifiers are available as described in the section above.



request was submitted

[Feedback](#)
[Help](#)
James Thomas
[Logout](#)

Learn to apply this code:

INCLUDE on title & abstract

Distinguish from this code:

EXCLUDE on target group

Name for your model:

My test model

Build Model

Refresh Models

Close/back

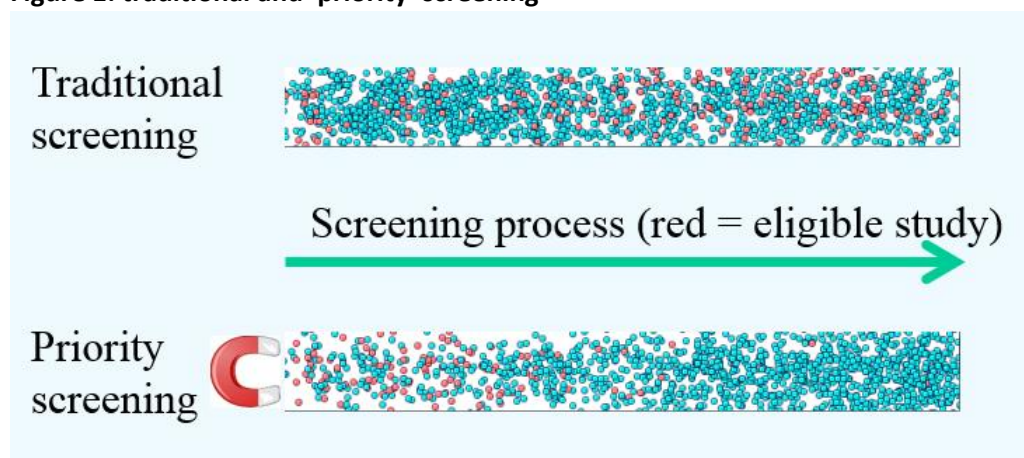
| <input type="checkbox"/> | Title | Att On | Att Not On | Accuracy | AUC | Precision | Recall |
|--------------------------|--------------------|-----------------------------|-------------------------|----------|-------|-----------|----------|
| <input type="checkbox"/> | My test model | INCLUDE on title & abstract | EXCLUDE on target group | 0.667 | 0.5 | 0.5 | 0.999999 |
| <input type="checkbox"/> | Aspirin vs paracet | Aspirin | Paracetamol | 0.933 | 0.974 | 0.932 | 0.932 |

3. Priority screening?¹

What is priority screening?

Systematic reviews are suffering from increasing 'data deluge': reviewers often need to manually assess many thousands of titles and abstracts to determine their relevance. Text (or 'data') mining / machine learning, is one way of reducing this workload.[2], [3] The aim of priority screening is for the machine to 'learn' the characteristics of included and excluded studies, and to be able to predict whether a given record is more likely to be relevant or irrelevant. As Figure 1 shows, using blue dots to indicate 'irrelevant' records and the red dots to indicate 'relevant' records, the records reviewers are interested in are usually distributed at random (essentially) throughout the list of irrelevant records. Priority screening 'pulls' the relevant records towards the beginning of the screening process and 'pushes' the irrelevant ones towards the end.

Figure 1: traditional and 'priority' screening

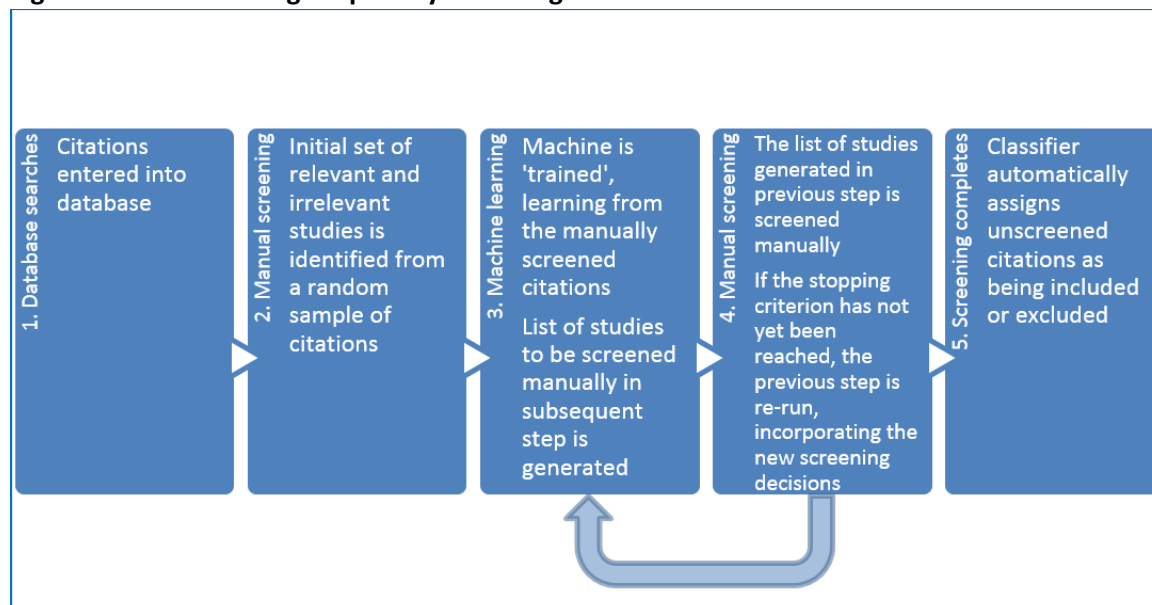


The way in which priority screening works is through a process known as 'active learning'[4] which is illustrated in Figure 2. Briefly put, 'active learning' is an iterative process whereby the accuracy of the predictions made by the machine are improved through interaction with users (reviewers). When used in a review, active learning involves the reviewer screening a small number of records manually; the machine then 'learns' from these decisions and generates a list of records for the reviewer to look at next. This cycle continues, with the number of reviewer decisions growing, until a given stopping criterion is reached and the process ends (e.g. the reviewer has identified all the relevant records they had expected; they have run out of time; they have screened all the records manually; etc). The mechanism for generating the list of records to be examined manually is under active consideration.[5]

In EPPI-Reviewer, the length of iteration grows as the number of documents screened increases. When relatively few records have been screened, the algorithm re-learns and re-scores remaining records every 25 items screened. Once hundreds – and thousands – of records have been screened, the algorithm runs less frequently (every 100, 500 and 1000 records) in order to conserve server resource. Simulation studies have shown that this does not affect performance adversely, since once a good number of records have been screened, the re-ranking of records changes their order less and less.

¹ This section is based on text and figures from: Thomas J (2013) Diffusion of innovation in systematic review methodology: why is study selection not yet assisted by automation?1. Thomas J. Diffusion of innovation in systematic review methodology: why is study selection not yet assisted by automation? . OA Evidence-Based Medicine. 2013;1(2):12.

Figure 2: active learning for priority screening



Automation does not replace manual work, but aims to reduce it (hence, it should properly be referred to as ‘semi-automation’), by focusing manual effort on the most relevant records, aiming to identify 100% of eligible records as quickly as possible.

A range of possible results is shown in Figure 3: with the process of screening each records manually being presented along the x-axis (0-100%) and the cumulative number of relevant records (‘includes’) identified along the y-axis (0-100%). In a traditional review, where records are screened essentially at random, we expect to see relevant records identified in proportion to the number screened: depicted by the grey diagonal line. The green line shows priority screening operating fairly effectively, with relevant records identified at a much quicker rate and 100% of them found by the time about half of the records have been examined. In theory, the reviewer can then discard the remaining 50% of records, safe in the knowledge that they are not relevant to their review. Such results have been reported by several teams. Wallace et al report that their technique might have reduced screening effort by between 40 and 50% in three reviews [4]; and by between 67 and 92% in four examples of review updates [6]. Aaron Cohen and colleagues present similarly promising results (6,7), and at least three groups are building systems which use machine learning to facilitate the retrieval of studies in reviews [7]–[9]. However, the advisability of truncating screening in specific situations is unknown at present, and is the subject of current empirical work.

How does the machine learning work?

As described above, the machine ‘learns’ from records that have been labelled by users as being relevant and irrelevant. This learning takes two phases: 1) ‘feature selection’, in which the records are transformed into numerical form; and 2) ‘model building’. Once the ‘learning’ is complete, then the machine can apply its ‘knowledge’ to unseen records.

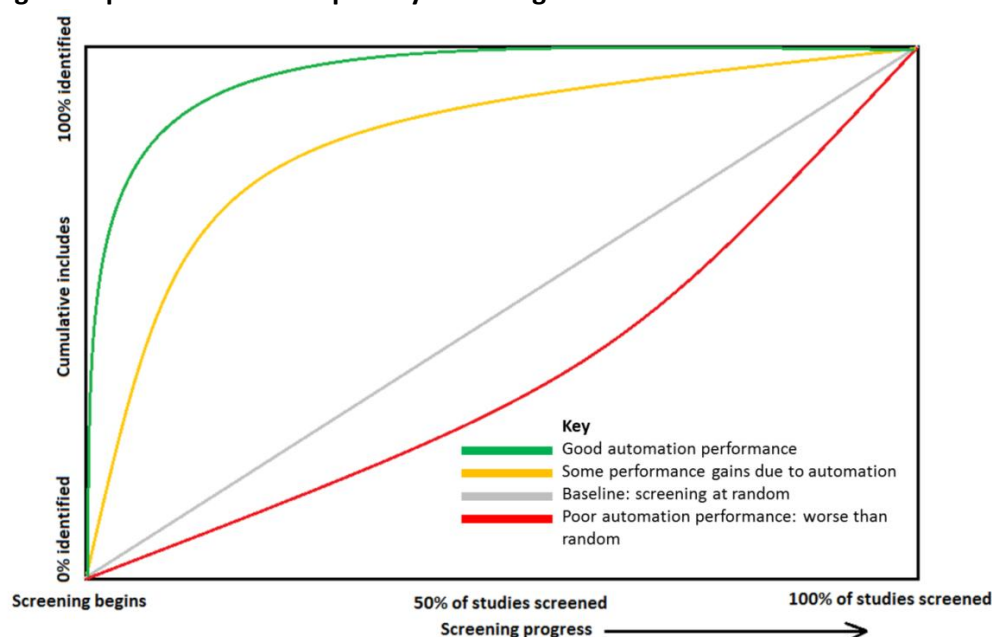
Feature selection involves the transformation of text into numbers, which can be used in a machine learning algorithm. We use a ‘tri-gram’ ‘bag of words’ approach, which involves building a list of every word, every pair of words, and every triplet of words in every records. We then index all the records (titles and abstracts) against this list of words (after removing the words which appear in the PubMed stopwords list). At its most basic, this involves simply counting the number of times a word occurs. In our implementation we use ‘term frequency – inverse document frequency’ which captures more information about how well the word (or word-phrase) distinguishes one records from another. Unlike some implementation of ‘bag of words’ features, we do not stem the words, but leave them intact, as valuable information can be lost when stemming words.

Once we have transformed the records into numeric form, a statistical model is built, which aims to distinguish one class of records (the ones that are relevant to the review) from the other. The algorithm we use is a ‘support vector

machine’ as implemented in the Scikit-Learn Python machine learning library.[10] This algorithm was selected because it is efficient (and quick) and scales to large numbers of documents. This is important in a live environment.

Once the model has been built, all unscreened records are then ‘scored’ according to the model. These records are then presented for screening in order, with the most relevant (i.e. with the highest scores) at the top of the list.

Figure 3: possible results of priority screening



How to use priority screening in EPPI-Reviewer 4

Methods for using active learning in systematic reviews are still being developed. Detailed here are our current ideas about how best to utilise priority screening, based on previous work in collaboration with the Behaviour and Health Research Unit, University of Cambridge and National Centre for Text Mining, University of Manchester, and are likely to change in the future.[11]

Stage 1: text preparation

All records should be uploaded and de-duplicated before priority screening commences.

Stage 2: baseline inclusion rate / inter-reviewer reliability

As documented elsewhere [11] it is a good idea to establish the ‘**baseline inclusion rate**’ (BIR) before priority screening begins so that: a) an unbiased sample of ‘training’ data (i.e. manual screening decisions) is available; and b) the likely number of relevant citations can be estimated. This is accomplished by manually screening a random sample of records (the required size of the sample can be determined by a standard power calculation [11]). If multiple reviewers will be screening records, then the stage of obtaining the BIR can be combined with testing for, and ensuring, good inter-reviewer reliability.

Stage 3: Codeset preparation

Once the BIR has been established and a reasonable quantity of ‘training’ data been coded (at least six relevant records and six irrelevant records should be identified), the priority screening can begin.

There are three operations to complete before reviewers click ‘begin screening’ and begin to plough through records.

1. First, the dropdown box entitled ‘Screening Code set’ should be set to the code set that you are going to use for screening.
2. *Second, the include / exclude codes should be set. These are automatically entered when a screening codeset is selected, but can be manually adjusted. The checkbox next to each code should be ticked for an ‘include’ code, and unticked for an ‘exclude’ code.

3. Third, the 'create list of items to screen' button should be clicked. This will generate the initial list of citations to be screened.

* Often, this will simply be the same code set as the codes you entered in 1 came from. However, sometimes it is useful to train on a sub-set of codes: for example, if you have a date filter, and the studies before your cutoff date 'look' the same as those afterwards to the text mining, it might be better not to include this in your list of codes to train from, as it might affect the classifier's performance negatively.

Stage 4: Priority screening can begin

Once the above steps have been taken, reviewers simply log in, go to the 'screening' tab and click 'begin screening'; they will be taken to the familiar coding page and presented with a records for screening. Coding then proceeds as usual, except for the fact that the index number of the record reflects its position in the current list, and can go up as well as down (as the priority screening code reorders items behind the scenes).

The screening tab gives a graphical display showing screening progress, and the term list (listing the terms that can be highlighted in titles and abstracts) can also be edited from this tab.

The priority screening will prioritise the most relevant records for manual screening; however, we currently recommend that all items are screened manually, as further empirical work is needed to determine the impact in different situations of halting screening early.

References

- [1] I. Marshall, A. N. Storr, J. Kuiper, J. Thomas, and B. C. Wallace, 'Machine Learning for Identifying Randomized Controlled Trials: an evaluation and practitioner's guide', *Res. Synth. Methods*, no. December 2017, pp. 1–13, 2018.
- [2] J. Thomas, 'Diffusion of innovation in systematic review methodology: Why is study selection not yet assisted by automation?', *OA Evidence-Based Med.*, vol. 1, no. 2, pp. 1–6, 2013.
- [3] J. Thomas, J. McNaught, and S. Ananiadou, 'Applications of text mining within systematic reviews', *Res. Synth. Methods*, vol. 2, no. 1, pp. 1–14, 2011.
- [4] B. C. Wallace, T. A. Trikalinos, J. Lau, C. E. Brodley, and C. H. Schmid, 'Semi-automated screening of biomedical citations for systematic reviews', *BMC Bioinformatics*, vol. 11, no. 1, p. 55, 2010.
- [5] M. Miwa, J. Thomas, A. O'Mara-Eves, and S. Ananiadou, 'Reducing systematic review workload through certainty-based screening', *J. Biomed. Inform.*, vol. 51, pp. 242–253, 2014.
- [6] B. C. Wallace, K. Small, C. E. Brodley, J. Lau, C. H. Schmid, L. Bertram, C. M. Lill, J. T. Cohen, and T. A. Trikalinos, 'Toward modernizing the systematic review pipeline in genetics: Efficient updating via data mining', *Genetics in Medicine*, vol. 14, no. 7, pp. 663–669, 2012.
- [7] S. Ip, C. D'Ambrosio, K. Patel, N. Obadan, G. D. Kitsios, M. Chung, and E. M. Balk, 'Auto-titrating versus fixed continuous positive airway pressure for the treatment of obstructive sleep apnea: A systematic review with meta-analyses', *Syst. Rev.*, vol. 1, no. 1, 2012.
- [8] J. Thomas, J. Brunton, and S. Graziosi, 'EPPI-Reviewer: software for research synthesis'. Social Science Research Unit, Institute of Education, London, 2010.
- [9] J. J. Yang, A. M. Cohen, and M. S. McDonagh, 'SYRIAC: The systematic review information automated collection system a data warehouse for facilitating automated biomedical text classification.', *AMIA Annu. Symp. Proc.*, pp. 825–829, 2008.
- [10] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, G. Louppe, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and Edouard Duchesnay Pedregosa, al David Cournapeau, M. Perrot matthieuperrot, and cea Edouard Duchesnay, 'Scikit-learn: Machine Learning in Python', *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [11] I. Shemilt, A. Simon, G. J. Hollands, T. M. Marteau, D. Ogilvie, A. O'Mara-Eves, M. P. Kelly, and J. Thomas, 'Pinpointing needles in giant haystacks: Use of text mining to reduce impractical screening workload in extremely large scoping reviews', *Res. Synth. Methods*, vol. 5, no. 1, pp. 31–49, 2014.