Supporting Information for

AIM review tool: artificial intelligence for smarter systematic review screening

Sergio Mena[1,2], Esther Rituerto-González[2,3,4], Fiona Coutts[1], Jana von Trott[1,2], Grace R. Jacobs[1], Linda Bryant[1], Louise Moles[1], Nicoleta Sirbu,[1] Liisi Promet[2], Dominic Oliver[1,5], Muhammad S. Ahmed[6], Paolo Fusar-Poli[1], Marian J. Bakermans-Kranenburg[7,8], Marinus H. van IJzendoorn[8,9], Nikolaos Koutsouleris[1,2,3,4*], Paris Alexandros Lalousis[1,2*]

[1] Department of Psychosis Studies, Institute of Psychiatry, Psychology and Neuroscience, King's College London, United Kingdom.
[2] Department of Psychiatry and Psychotherapy, Ludwig-Maximilians-University, Munich, Germany.
[3] Max Planck Institute of Psychiatry, Munich, Germany.
[4] German Centre for Mental Health (DZPG), partner site Munich-Augsburg.
[5] Department of Psychiatry, University of Oxford, United Kingdom.
[6] Department of Psychological Medicine, Institute of Psychiatry, Psychology and Neuroscience, King's College London, United Kingdom.
[7] William James Centre for Research, ISPA, Lisbon, Portugal.
[8] Facultad de Psicología y Humanidades, Universidad San Sebastián, Sede Valdivia, Chile.
[9] Research Department of Clinical, Education and Health Psychology, Faculty of Brain Sciences, UCL, London, United Kingdom.

* Joint Senior Authorship
Corresponding author: Sergio Mena
Email: sergio.mena_ortega@kcl.ac.uk

This PDF includes:

Figure S15. Area under the ROC curve, balanced accuracy and P4 score of supervised learning models.

1. Supplementary Methods

1.1 Client-side architecture and implementation

AIM Review is implemented as a client-side web application, operating entirely without server-side interaction. Users can access all features of the web app without the need for account creation. Progress, including paper screening data, numerical summaries of entries, and model predictions, can be saved and restored locally by exporting and importing spreadsheet files. As a serverless application, all user activity and systematic review data are stored locally and temporarily within the browser's local storage. No data is transmitted or stored externally, ensuring that all information remains confined to the user's device. This design ensures that users remain anonymous and always retain full control over their data.

Alternatively, AIM Review also allows users to register for an account using Firebase Authentication via email, Google or Github accounts. This functionality allows users to save their screening progress in a secured Firebase Storage unit and under a project ID. In this case, the progress, data structures, and email address signed in become visible to AIM Review administrators; however, none of this information will be shared with third parties or used for purposes other than maintaining the application's functionality and debugging errors. This option is entirely optional, as the application is primarily designed to function as a static application served by local files on the user's computer.

1.2 Technology stack

AIM Review is developed using modern web technologies to handle heavy computations in the browser:

- **HTML, CSS, and Client-Side JavaScript** provide the core framework and user interface. Unless stated otherwise, data storage and handling, as well as some computational algorithms are implemented in vanilla JavaScript.
- **Pyodide** (a WebAssembly implementation of Python)[1] enables access to Python-based libraries, including sklearn for machine learning models and text vectorization methods like • Term Frequency-Inverse Document Frequency (TF-IDF) and Latent Semantic Analysis (LSA).
- **TensorFlow.js**[2] is used to implement and access neural network-based models, such as Universal Sentence Transformers and a native implementation of Doc2Vec. TensorFlow.js supports training lightweight models directly in the browser.
- **Hugging Face Transformers.js** library and API service are used to locally access pre-trained transformer models for sentence embeddings (with the JavaScript library) and integrate literature-specific models like SciNCL and SPECTER2,[3,4] only available via API calls.
- **Web Workers** are employed to handle computationally intensive tasks, such as text vectorization and machine learning model training. This ensures a smooth user experience by offloading heavy processing to background threads without blocking the main application interface.

1.3 Application structure

AIM Review is structured as a modular multi-page application. The application has 3 main applications that allow users to screen titles and abstracts of publications for their systematic review, measure inter-rater agreements, make decisions on publications where there is a disagreement, and finally utilize robust machine learning pipelines to predict relevant papers for your systematic review. We will describe the main features of each modular application below, as well as specific components of the applications in the following sections.

Labelling application: This application enables users to upload publication extractions from databases (e.g., PubMed and Web of Science) and manually screen the titles and abstracts of publications as relevant or irrelevant for their systematic review project. The application includes features such as the removal of duplicate entries, highlighting relevant and irrelevant expressions in the title and abstract, and interactive plots that display the screening progress and statistics. Users can opt to screen a randomized subset of publications to create a labeled dataset for training machine learning models. Additionally, the labelling application incorporates an active learning algorithm that iteratively learns from the user's manual screening to assign relevance scores to publications, with an option to sort them by relevance.

Agreement measures: This application enables the assessment of inter-rater agreement and document-level agreement during the title and abstract screening process. It also allows users to resolve disagreements by making final decisions on disputed documents. Labels from different raters, exported from the labelling application, can be imported into this module to calculate agreement metrics and facilitate consensus building.

ML prediction: this application allows the use of robust, nested cross-validated supervised machine learning pipelines to convert the titles and abstracts of publications into numerical representation, train classifiers with papers screened by the used and use the classifiers to predict the relevance of unscreened papers.

The order of these applications reflects the sequence in which users typically perform their systematic review screening. However, the web application is designed to allow all modular applications to run in parallel. This allows users to run all applications at the same time from the same window. Relevant components of the application will be described below. For a full description of the application and all its components, see the documentation pages.

1.4 Inter-rater agreement measures

The inter-rater agreement measures in AIM Review provide two key metrics to evaluate the consistency of labels assigned by multiple raters during the title and abstract screening process:

Percentage of total agreement (%): this metric represents the proportion of entries where all raters assigned the same label. It is calculated as:

$$\mathrm{P} = 100 \cdot \frac{\mathrm{N_{agree}}}{\mathrm{N_{total}}} \tag{1}$$

Where N_{agree} is the number of entries in which all raters agree, and N_{total} is the total number of entries. While straightforward and easy to interpret, this measure does not account for the level of agreement that may occur by chance.

Fleiss' Kappa statistic: Fleiss' kappa is a statistical measure of inter-rater reliability, accounting for chance agreement.[5–7] It evaluates the level of agreement among N raters assigning one of two labels (relevant or irrelevant) to M entries. Fleiss' Kappa, and its statistical significance are calculated following these steps:

Step 1: Calculate the proportion of ratings for each label across all entries. For each label j, calculate the proportion of ratings p_j across all entries:

$$\mathrm{p_j} = \frac{1}{\mathrm{NM}} \sum_{\mathrm{i=1}}^{\mathrm{M}} \mathrm{n_{ij}} \tag{2}$$

Where n_{ij} is the number of raters who assigned label j to entry i, N is the total number of raters, and M is the total number of entries.
Step 2: Calculate the entry-wise agreement score. For each entry i, calculate the agreement score P_i, which measures the extent to which raters agree on that entry:

$$\mathrm{P_i} = \frac{1}{\mathrm{N(N-1)}} \left(\sum_{\mathrm{j=1}}^{\mathrm{k}} \mathrm{n_{ij}} \left(\mathrm{n_{ij}} - 1 \right) \right) \tag{3}$$

Where k is the number of possible labels (2 for relevant/irrelevant), n_{ij} is the number of raters who assigned label j to entry i.

Step 3: Calculate the mean of the entry-wise agreement scores. The mean agreement score $\bar{P}$ is the average of the entry-wise agreement scores:

$$\mathrm{\overline{P}} = \frac{1}{\mathrm{M}} \sum_{\mathrm{i=1}}^{\mathrm{M}} \mathrm{P_i} \tag{4}$$

Step 4: Calculate the expected agreement by chance. The expected proportion of agreement by chance for each label is:

$$\bar{P}_e = \sum_{j=1}^{k} p_j^2 \tag{5}$$

Where p_j is the proportion of raters who assigned label j across all entries.

Step 5: Calculate Fleiss' kappa. Fleiss' kappa is then calculated as:

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e} \tag{6}$$

Step 6: Calculate the standard error of Fleiss' kappa. The standard error SE_k of Fleiss' kappa is calculated as:

$$SE_\kappa = \sqrt{\frac{2}{N(M-1)}\left(\bar{P}_e(1-\bar{P}_e) + (M-1)\left(\frac{\bar{P}_e}{N} - \frac{2}{M(M-1)}\sum_{j=1}^{k} p_j^3\right)\right)} \tag{7}$$

Step 7: Calculate the 95% confidence interval for Fleiss' kappa. The 95% confidence interval for Fleiss' kappa is calculated as:

$$Z = \frac{\kappa}{SE_\kappa} \tag{8}$$

Where Z is the Z-score corresponding to the desired confidence level (1.96 for a 95% confidence level).

Step 9: Calculate the p-value. The p-value is calculated using the standard normal distribution:

$$p = 2\big(1 - \Phi(|Z|)\big) \tag{9}$$

Where Φ is the cumulative distribution function of the standard normal distribution.

The interpretation of Fleiss' kappa depends on the context being used, sample size and independence during title and abstract screening.[7] A positive Fleiss' Kappa indicates more agreement among raters than would be expected by chance. Higher values suggest stronger agreement. A Fleiss' kappa close to 0 suggests that the agreement is no better than what would be expected by chance alone, indicating that raters are not agreeing more than would be expected randomly. A negative Fleiss' kappa indicates a stronger disagreement that the disagreement

achieved by chance, and it can be statistically significant. This indicates some type of systematic disagreement among the raters.

A suggested interpretation of Fleiss' kappa values:

- $0.21 \leq \kappa \leq 0.40$: Fair agreement.
- $0.41 \leq \kappa \leq 0.60$: Moderate agreement.
- $0.61 \leq \kappa \leq 0.80$: Substantial agreement.
- $0.81 \leq \kappa \leq 1.00$: Almost perfect agreement.

The percentage of total agreement is easy to calculate and interpret but may overestimate true agreement by not accounting for chance. Fleiss' kappa addresses this by adjusting for chance, but its reliability depends on the sample size and assumptions about rater independence. It performs best with many entries and consistent rater behavior.

Finally, correlation and agreement coefficients for pairs of raters can be also calculated. AIM Review provides the following available options are:

- **Spearman's Correlation**: A rank-based correlation coefficient that measures the monotonic relationship between two variables. It ranges from -1 to +1, where values closer to 1 indicate strong positive correlation, and values closer to -1 indicate strong negative correlation.
- **Kendall's Tau**: A non-parametric measure that evaluates the strength and direction of association between two variables by comparing the number of concordant and discordant pairs. Kendall's tau also ranges from -1 to +1.
- **Cohen's Kappa**: A statistic that evaluates the agreement between two raters, accounting for the possibility of agreement occurring by chance. Cohen's kappa is a specific case of Fleiss' kappa when the number of raters is limited to two.

1.5 Document-level agreement measures

In addition to inter-rater agreement scores, AIM Review provides a detailed breakdown of agreement percentages for document. This is the percentage of agreement among raters for each document. These percentages are visually represented in a bar plot displayed at the bottom of the application. The color coding of the bars indicates the level of agreement among raters:

- **Green**: Agreement percentage exceeds 80%, indicating a high level of consensus.
- **Yellow**: Agreement percentage is between 60% and 80%, representing moderate agreement.
- **Red**: Agreement percentage is below 60%, reflecting low agreement.

This visual representation allows users to quickly identify areas with strong consensus as well as entries where disagreement among raters is more pronounced. Documents were there was an initial disagreement can be opened in the application and a finalized label can be applied to the document following consensus among the raters.

The statistical significance of correlation and agreement coefficients between two raters are corrected for multiple comparisons to control for the risk of Type I errors (false positives). The available correction methods are:

- **False Discovery Rate (FDR)**: Correction using the Benjamini–Hochberg procedure, which controls the expected proportion of false discoveries among significant results.

- **Bonferroni Correction**: A more conservative method that adjusts the significance level by dividing it by the number of comparisons.

The significance level (α) for statistical tests is also configurable, with a default value set at 0.05. This threshold determines the probability at which results are considered statistically significant. All inter-rater and document-level measures of agreement are developed in native JavaScript in the client-side application.

1.6 Text vectorization methods

Text vectorization is the process of converting textual data like titles and abstracts of publications into numerical representations that algorithms can process. It transforms the titles and abstracts into numeric vectors while preserving the meaning of individual words, semantic and syntactic relationships.

AIM Review provides multiple state-of-the-art vectorization methods with varying degrees of complexity to convert the titles and abstracts uploaded into the application into a numerical representation that the machine learning classifiers can use. Following, a description of each method is provided.

Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF is a fast vectorization method that represents text as a weighted vector, where each word's numerical value reflects its importance within a document relative to the entire collection of documents. The importance increases proportionally to a word's frequency in the document but is offset by its occurrence across all documents, reducing the weight of common terms. AIM Review uses Python's scikit-learn (sklearn) implementation of TF-IDF, which allows for flexible parameter configuration, such as normalization, definition of maximum vector length and cut-off words with a minimum and maximum frequency. Users can adjust these settings in the application window to optimize the vectorization process. For further details, refer to the official sklearn documentation.

Latent Semantic Analysis (LSA) of TF-IDF. LSA of TF-IDF applies dimensionality reduction to the TF-IDF feature matrix to uncover latent relationships between words and documents. By reducing the number of features, LSA captures the most significant semantic structure in the text while mitigating sparsity. AIM Review utilizes Python's scikit-learn implementation of TF-IDF and truncated singular value decomposition (SVD) for LSA. Detailed parameter explanations are available in the sklearn documentation.

Universal Sentence Enconder (tf.js)
Universal Sentence Encoder is a transformer-based neural network model that generates high-quality contextual embeddings for universal text. Provided via TensorFlow.js, Universal BERT converts text of any length into fixed 512-dimensional numerical vectors. AIM-Review allows users to configure whether to:

- Merge titles and abstracts before vectorization (producing a single 512-dimensional vector), or
- Convert titles and abstracts independently and concatenate the resulting vectors (resulting in a 1024-dimensional representation).

Doc2Vec (native implementation). Doc2Vec is an extension of the Word2Vec algorithm used to create fixed-size vector representations of documents.[10] Unlike simpler methods like TF-IDF, Doc2Vec captures word order as well as the contextual meaning by using a neural network architecture that learns the distributed representations of words in documents. In AIM Review, a lightweight implementation of the distributed bag of words (DBOW) algorithm[11,12] is implemented using artificial neural networks with Tensorflow.js. DBOW works by predicting words within a document based solely on the document vector. To do this, titles and abstracts are tokenized, and a vocabulary is built to convert tokens into indexed sequences; these are indexed numeric representations of the words in the titles and abstracts. These sequences are padded or trimmed to uniform lengths and converted into tensors. A neural network initializes random document vectors and trains them to predict words, optimizing the vectors to represent the document's content. After training, the model outputs fixed-size document embeddings that are numeric representation of the words and their semantic meaning in the document. This implementation captures the essence of Doc2Vec and DBOW while ensuring it remains lightweight and efficient for client-side execution in a web application. In future iterations, as web-based deep learning capabilities continue to advance, we aim to incorporate comprehensive Doc2Vec implementations, comparable to those available in Python's Gensim library.

Sentence transformers (JavaScript and API). Sentence transformers is a Python library provided by Hugging Face that offers pre-trained transformer-based models for computing dense vector representations (embeddings) of text. At their core, sentence transformers are based on models like BERT (Bidirectional Encoder Representations from Transformers)[8,9] and other transformer architectures. These models use self-attention mechanisms to understand the contextual relationships between words within a sentence, enabling them to generate high-quality embeddings that represent the meaning of the titles and abstracts. AIM Review accesses pre-trained sentence transformers using two methodologies. First, via transformers.js, a JavaScript library that allows to run transformers directly in a client-side environment. This implementation includes universal text transformers such as all-MiniLM-L6-v2, mxbai-embed-large-v1 and bge-small-en-v1.5. Additionally, AIM Review also allows to connect to hugging Face inference API service. Users need an API key from Hugging Face and API calls are restricted to 1000/day with a Hugging Face free account. The API key is strictly for personal use and always remains in the user's device at all times. Current models available via the inference API include SciNCL and SPECTER2[3,4], which are transformers trained with a corpus of scientific titles and abstracts.

Multilingual vectorizations of text

While the sentence transformers and Universal Sentence Encoder mentioned above are English-only, TF-IDF, LSA of TF-IDF, and Doc2Vec are language-agnostic and can be applied to any language where whitespace tokenization is feasible. To extend their applicability, AIM Review has multilingual functionality by expanding the available stop word lists beyond English to include many other languages. Users can now configure AIM Review to select either an English-only or a multilingual stop word setting, which includes stop words for more than 50 languages (link), ensuring that TF-IDF, LSA, and Doc2Vec can effectively process texts in a wide range of languages. In addition, AIM Review also integrates the paraphrase-multilingual-MiniLM-L12-v2 model. This multilingual transformer supports more than 50 languages and maps text into a shared vector space, enabling cross-lingual semantic representations. This extension ensures broader applicability of AIM Review to systematic reviews in multiple languages beyond English.

Classification models

AIM Review implements a range of classification algorithms to analyze the vectorized representations of titles and abstracts and generate predictions of publication relevance. Below, we describe the currently available classification models and their configurations.

Logistic regression (sklearn). Logistic regression is a linear model commonly used for binary classification tasks. It predicts the probability of a binary outcome (e.g., 0 or 1) using the logistic function, making it well-suited for scenarios where the relationship between input features and the target variable is approximately linear. AIM Review utilizes the scikit-learn implementation of logistic regression, allowing users to adjust parameters such as the penalty norm (L1 or L2), solver type, class weights, and the regularization parameter (C). For further details, refer to the official sklearn documentation.

Linear SVM (sklearn). The linear support vector machine (SVM) algorithm is a linear model designed for binary or multiclass classification tasks. It identifies the optimal hyperplane that separates data points of different classes with the maximum margin to ensure generalization. Linear SVMs are computationally efficient and particularly effective for high-dimensional datasets. AIM Review utilizes the sklearn implementation of linear SVM, based on the liblinear library.[13] Users can configure parameters such as the regularization parameter C, class weights, penalty, loss and formulation. For further details, refer to the official sklearn documentation.

SVM (sklearn). This option employs the support vector machine algorithm. Unlike linear SVM, this implementation supports non-linear kernels, such as polynomial, radial basis function (RBF), and sigmoid kernels, allowing it to capture more complex patterns in the data. AIM Review utilizes the sklearn implementation of the SVM algorithm, which wraps around libsvm.[14] Configurable parameters include the kernel type, degree of the polynomial kernel, kernel coefficient (gamma), class weights, and the regularization parameter C. For further details, refer to the official sklearn documentation.

Decision tree (sklearn). This option employs the decision tree algorithm. Unlike linear models, decision trees recursively partition the data into subsets based on feature values, forming a tree structure. The sklearn implementation of decision trees used in AIM Review provides flexibility for configuring parameters, including the criterion for splitting, the maximum depth of the tree, the minimum number of samples required to split a node, the minimum number of samples required at a leaf node, and the maximum number of features considered for splitting. For further details, refer to the official sklearn documentation.

MLP (sklearn). This option employs a multilayer perceptron (MLP) classifier algorithm, a feedforward neural network model of fully connected neurons with nonlinear activation functions. AIM Review utilizes the sklearn implementation of MLP classifiers. Users can configure parameters such as the number of hidden layers, activation function, solver type, learning rate, and regularization terms. For further details, refer to the official sklearn documentation.

SeqNN (tensorflow.js). This option employs a sequential neural network composed of fully connected layers defined and computed using the Tensorflow JavaScript API. It supports multiple hidden layers with non-linear activation functions. Users can configure parameters such as the activation function, optimization solver, batch size, learning rate, maximum iterations, early

stopping criteria, validation fraction, class weights, and the number of neurons in each hidden layer. For further details, refer to the official TensorFlow.js API documentation.

While AIM Review offers a range of shallow learners and deep learners, we prioritized the implementation of shallow learners to ensure computational efficiency and maintain low latency in the application.

1.7 Active learning strategies

AIM Review labelling application integrates an active learning algorithm to iteratively provide relevance scores of unscreened publications using the user-provided screening. This process utilizes the user-provided screening data that is available at specific time points, and classifiers are trained recursively every time new manual screening is available to produce new relevance scores. Publications can be ordered by the algorithm predicted relevance score, so that publications with a higher chance of full-text review are screened first. The key steps in the process are the following:

Step 1. Initialization. The application starts with all unlabeled titles and abstracts from publications. The user starts manually screening the titles and abstracts. When the active learning button is pressed for the first time, AIM Review vectorizes all titles and abstracts in the background.

Step 2. Model training. Once the initial vectorization is finished, the application automatically trains a classifier (selected in the configuration) in the background to predict the user's screening using the numerical representations of the titles and abstracts.

Step 3. Relevance scores. The classifier is then used to predict the unlabelled publications and produce a relevance score (0-100%) based on the probability score computed by the classifier. The score is displayed in the application using a color-coded measure of relevance. The titles and abstracts can be sorted by those with a highest relevance first, lowest relevance first, or those with the highest uncertainty (probability closer to the threshold of 0.5). This is particularly useful when only a subset of publications with the highest relevance are going to be reviewed, especially in cases where the total number of publications is large. In such scenarios, users may not want to go through all publications but rather prioritize screening the most relevant ones based on the classifier's predictions. Additionally, publications can be sorted using an uncertainty-based method

Step 4. Continuous learning. Steps 2 and 3 can be repeated by manually pressing the "active learning" button. A new classifier will be retrained with all the available labels, and new relevance scores will be produced. Optionally, the active learning process can be set to recursively retrain the model and produce relevance scores if either new screened publications are available or using a time interval.

A visual representation of the active learning cycle is given in **Figure 1A** of the manuscript.

1.8 Supervised learning strategies

AIM Review uses nested cross-validation (NCV) to obtain a reliable assessment of the model performance in unseen publications while simultaneously tuning hyperparameters of models. NCV involves two loops, an outer cross-validation loop and an inner cross-validation loop. At the outer cross-validation level (CV2), the dataset is split into N folds. One fold of the data is iteratively held back as a validation sample, while the remaining data (N-1 outer folds) were passed into the inner cross-validation (CV1) cycle. Within the inner cross-validation cycle (CV1), the sample (N-1 folds) are further split iteratively into M inner folds. One fold of the data is iteratively held back as a test sample, while the remaining M-1 folds are used as the training set. The training set in this inner framework is used to train models, and the test set is used to evaluate and identify the hyperparameter combinations that performed best on unseen publications. Finally, the outer fold, which is left out as the validation sample, is used to validate the model's performance on data not exposed to the hyperparameter optimization process. This approach ensures that the model selection and performance evaluation are unbiased and generalizable to unseen publications. Inner and outer loop permutations can also be configured in the application. These indicate how many times the samples are randomly shuffled, with the cross-validation process (both inner and outer) repeated for each permutation to ensure robust and reliable performance estimates. Based on the mean performance across the validation sets (for all outer permutations and folds), the optimal model is selected and retrained with all the CV2 training data to be consecutively applied to the CV2 test data. To provide a comprehensive estimate of the model's performance on unseen publications, an ensemble of predictions is generated for all samples when they are in the outer CV2 validation set. This ensemble aggregates the predictions across all iterations of the outer cross-validation, thereby producing a robust estimate of the model's performance on samples that were not used during training.

A receiver operating characteristic (ROC) curve is then constructed based on the predictions in the validation sets. The ROC curve serves as a visual tool to assess the trade-off between sensitivity and specificity at various decision thresholds and aids the user in selecting a probability threshold with an estimated level of sensitivity and specificity to be applied to the unlabeled publications and generate the relevance predictions. For example, if the user opts for a conservative approach by selecting a high probability threshold (e.g., 0.75), only those publications with a very high predicted probability of relevance (above 95%) will be labeled as relevant. This minimizes the risk of false positives, ensuring that only the most confidently predicted relevant papers are identified, but may also result in fewer publications being classified as relevant, thereby reducing recall. Conversely, a liberal approach with a lower probability threshold (e.g., 0.25) will label more publications as relevant, increasing recall but potentially introducing more false positives. In this case, a broader range of publications will be flagged as relevant, even those with lower confidence, which may increase the number of irrelevant papers identified as relevant. The choice of threshold thus directly impacts the balance between sensitivity (true positive rate) and specificity (true negative rate), and AIM Review users can adjust it based on their desired trade-off between these metrics. A visual representation of the supervised training strategies is given in **Figure 1B** of the manuscript.

Once a probability threshold is selected (0.5 as default), AIM Review also provides the following metrics to estimate the predictive performance of the model:

Accuracy (ACC): proportion of correctly predicted cases (both positive and negative) out of the total number of cases and is calculated as

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{10}$$

Balanced accuracy (BAC): average of the true positive rate (TPR) and the true negative rate (TNR), calculated as

$$\mathrm{BAC} = \frac{1}{2}\left(\frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} + \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}\right) \tag{11}$$

Area under the receiver operating characteristic (AUROC): measure of the area under the ROC curve. It's a measure of the classifier's ability to distinguish between classes considering all probability thresholds, calculated using the Simpson's rule as

$$\mathrm{AUROC} \approx \frac{\Delta\mathrm{FPR}}{3}\left[\mathrm{TPR}_0 + 4 \cdot \sum_{\mathrm{i}=1,\,\mathrm{odd}}^{\mathrm{n}-1} \mathrm{TPR}_\mathrm{i} + 2 \cdot \sum_{\mathrm{i}=2,\,\mathrm{even}}^{\mathrm{n}-2} \mathrm{TPR}_\mathrm{i} + \mathrm{TPR}_\mathrm{n}\right] \tag{12}$$

$$\Delta\mathrm{FPR} = \frac{\mathrm{FPR}_\mathrm{n} - \mathrm{FPR}_0}{\mathrm{n}} \tag{13}$$

True positive rate (TPR), or sensitivity/recall: measures the proportion of actual positive cases that are correctly predicted, calculated as

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{14}$$

False positive rate (FPR), or fall-out: the proportion of actual negative cases that are incorrectly predicted as positive, calculated as

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{15}$$

True negative rate (TNR), or specificity: the proportion of actual negative cases that are correctly predicted, calculated as:

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} \tag{16}$$

False negative rate (FNR), or miss rate: the proportion of actual positive cases that are incorrectly predicted as negative

$$\mathrm{FPR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}} \tag{17}$$

Positive predictive value (PPV), or precision: the proportion of predicted positive cases that are actually positive, measured as

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{18}$$

Negative predictive value (NPV): the proportion of predicted negative cases that are actually negative, calculated as

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} \tag{19}$$

F_1 score: harmonic mean of precision (PPV) and recall (TPR), calculated as:

$$\mathrm{F}_1 = \frac{2 \cdot \mathrm{PPV} \cdot \mathrm{TPR}}{\mathrm{PPV} + \mathrm{TPR}} \tag{20}$$

P_4 score: composite performance measure that integrates Precision (PPV), Recall (TPR), Specificity (TNR), and Negative Predictive Value (NPV) into a single score. P_4 is designed in similar way to F_1 metric, however addressing the criticisms leveled against F1 regarding the lack of consideration of true negatives. It is defined as the geometric mean of these four metrics, calculated as:

$$\mathrm{P}_4 = \frac{4}{\frac{1}{\mathrm{PPV}} \cdot \frac{1}{\mathrm{TPR}} \cdot \frac{1}{\mathrm{TNR}} \cdot \frac{1}{\mathrm{NPV}}} \tag{21}$$

Where:

True Positive (TP): Cases where the model correctly predicts relevant publications as relevant.
True Negative (TN): Cases where the model correctly predicts irrelevant publications as irrelevant.
False Positive (FP): Cases where the model predicts the publication as relevant, but it is irrelevant.
False Negative (FN): Cases where the model predicts the publication as irrelevant, but it is relevant.

1.9 Ensemble strategies

In addition to individual classifiers, AIM Review supports ensemble methods to enhance prediction performance. Two ensemble strategies can be configured in the application:

Model Stacking. Model stacking trains an independent model for each vectorization modality. The predictions from these base models are then combined to train a meta-model, which produces the final output prediction. This approach leverages the strengths of individual models, capturing diverse aspects of the data. While model stacking can improve overall performance, it may require significant computational resources due to the training of multiple models.

Fusion of Modalities. This method integrates all vectorization modalities into a single dataset. A single model is trained directly on the fused feature set to generate the predictions, simplifying the pipeline and reducing computational overhead compared to model stacking. This strategy is particularly advantageous when computational resources are limited or when the modalities provide complementary information.

1.10 Case studies

To assess the performance of the text mining and machine learning methods implemented in AIM Review, we tested the performance of the pipelines on three case studies of systematic reviews [26–28] identified via professional networks. The case studies with different number of hits, proportion of accepted titles and abstracts and screening methodologies used. An additional three case studies[29–31] were later accessed from Synergy[32], a free and open dataset on study selection in systematic reviews, to evaluate the generalizability of AIM Review to other disciplinary fields such as computer science, environmental health and endocrinology. Following, we describe the methodologies employed on the title and abstract screening for case studies 1-3. An extensive description of case studies 4-6 can be found elsewhere (https://github.com/asreview/synergy-dataset).

Case Study 1: Systematic review of Adult Attachment Interview Studies. This case study utilizes the title and abstract screening from a systematic review of studies utilizing the Adult Attachment Interview (AAI),[15] which has been a pivotal tool in attachment research since its development in 1985. The systematic review process was preregistered on OSF (https://osf.io/zf82c/) and comprised two independent searches.

- **First Search**:
  - Search Terms: TS=("Adult Attachment Interview") NOT TS=(animal*) NOT TS=(chem*).
  - Databases: Web of Science Clarivate, Preprint Citation Index, ProQuest Dissertations and Theses Citation Index.
  - Inclusion: Original research publications that use the Adult Attachment Interview.
  - Exclusion: Meta-analyses, reviews, and papers published before 2005 (already included in a 2009 study).
  - Initial hits: 688 studies after removal of duplicates.
  - Screening: Titles and abstracts were reviewed by two independent coders.
  - Intercoder reliability: Cohen's k = 0.77 ($p < .001$).
  - Final selection: 362 studies.
- **Second Search**:

- Updates: Expanded sources to include biomedical sciences and extended the search period to December 2023.
- Additional hits: 89 (non-overlapping with the first search).
- Screening: Titles and abstracts by one coder.
- Final selection: 38 studies after removing duplicates and initial full-text screening.

The entries from the two searches were merged, resulting in a total of 777 records. To ensure compatibility with the AIM Review pipelines, entries without an available abstract were discarded. This filtering step reduced the dataset to 729 publications. Of these, 389 records (53.36%) were selected for full-text screening and assessment of eligibility. This case study represents a specific systematic review search with a moderate number of hits and with a balanced number of publications being accepted for full-text screening.

Case Study 2: Systematic Review on Data-driven Biological Subtypes in Schizophrenia Spectrum Disorders and Bipolar Disorder. This case study utilizes the titles, abstracts and screening from a systematic review of data-driven biological subtypes in schizophrenia and bipolar disorder. The review methodology was preregistered on PROSPERO (CRD42024572364).

- Search terms: (schizophrenia OR schizoaffective OR schizophreniform OR psychosis OR bipolar OR psychotic OR psychoses OR "bipolar disorder") AND (biotype OR subtype OR subgroup OR cluster OR subtyping OR clustering) AND ("brain structur*" OR neuroimaging OR "structural neuroimaging" OR neuroanatom* OR MRI OR "gray matter volume" OR "white matter volume" OR "fractional anisotropy" OR "functional connectivity" OR "biomarker" OR protein OR lipid OR mRNA OR SNP OR "single-nucleotide polymorphism").
- Keywords: brain structur*, neuroimaging, structural neuroimaging, neuroanatom*, magnetic resonance imag*, MRI, gray matter volume, grey matter volume, GMV, cortical thickness, cortical folding, gyrification, surface area, white matter volume, WMV, fractional anisotropy, FA, white matter tractography, diffusion tensor, diffusion-weighted, DTI, DWI, functional connectivity, seed-based connectivity, connectiv*, connectom*, neurotransmitter, PET, positron emission tomograph*, spectroscop*, blood, serum, plasma, saliva, cerebrospinal fluid, biomarker, proteom*, protein, lipid*, transcript*, mRNA, immun*, inflamm*, metabol*, electrophys*, electroencephalograph*, EEG, eye movement, oculomotor*, ocular, saccad*, electroretinogram, electroretinograph*, retina*, optical coherence tomography, genet*, genom*, polygen*, SNP and single-nucleotide polymorphism.
- Databases: PubMed, PsycINFO, Embase, Web of Science, and Scopus, with searches from database creation up to the search date on August 5, 2024.
- Screening: Titles and abstracts were reviewed by two independent investigators (blinded to each other's ratings). Discrepancies were resolved by agreement among the independent investigators.
- Inclusions: Original, peer-reviewed studies focusing on schizophrenia spectrum disorders or bipolar disorder, including both cross-sectional and longitudinal designs; published in

English; studies that employ data-driven clustering methods (e.g., k-means, hierarchical clustering, machine learning-based clustering) to identify biological subtypes; studies involving neuroimaging, electrophysiological, biochemical, or genetic data; studies that include both patient populations and healthy controls.

- Exclusions: Reviews, meta-analyses, and conference proceedings; studies that do not focus on schizophrenia spectrum disorders or bipolar disorder; studies that do not use data-driven clustering methods to identify subtypes; studies without relevant neuroimaging, electrophysiological, biochemical, or genetic data; studies that do not provide sufficient data for meta-analysis.
- Initial hits: 5,536 studies after removal of duplicates and entries with missing titles and abstracts.
- Intercoder reliability: Cohen's k = 0.67 ($p < .001$).
- Final selection: 190 studies after removing duplicates and initial full-text screening.

From the 5,536 studies, 389 (7.03%) were selected for full-text screening and assessment of eligibility. This case study represents a specific systematic review search with a moderate to high number of hits and with an unbalanced number of publications being accepted for full-text screening.

Case Study 3: Systematic Review and Meta-Analysis of Universal Prevention for Affective and Psychotic Disorders. This case study utilizes the titles, abstracts and screening from a systematic review and meta-analysis of universal prevention strategies for affective and psychotic disorders, conducted in compliance with PRISMA 2020 and MOOSE guidelines.[16] The review methodology was preregistered on PROSPERO (CRD42023428746) and followed the EQUATOR Reporting Guidelines.

- Search Terms: (prevent* OR population-level OR universal OR public health) AND (psychosis OR psychot* OR psychological OR depress* OR bipolar OR schizo* OR affective OR mood OR mental) AND (randomised controlled trial OR randomized controlled trial OR randomi?ed* OR RCT OR trial).
- Databases: Ovid (MEDLINE, EMBASE, PsycINFO) from inception to May 23, 2023.
- Screening: Titles and abstracts were reviewed sequentially by two independent investigators (blinded to each other's ratings). Discrepancies were resolved by including studies in the next round of screening.
- Inclusions: Parallel-group or cluster RCTs of universal preventive interventions targeting depression, bipolar disorder, or psychosis; published in peer-reviewed journals in English or Italian; baseline-free samples; follow-up incidence/prevalence of affective or psychotic disorders; and meta-analysable data.
- Exclusions: Reviews, conference proceedings, pilot data, selective/indicated interventions, non-English/Italian publications, overlapping samples, absence of control groups, non-target outcomes, or missing meta-analytic data. Overlapping samples were identified and resolved by including the largest, most representative study.

- Initial hits: 18,176 after removing duplicates.
- Screening: Titles and abstracts were reviewed by two independent coders.
- Intercoder reliability: Not measured.
- Final selection: 306 studies after removing duplicates and initial full-text screening.

To ensure compatibility with the AIM Review pipelines, entries without an available abstract or not in English were discarded. This filtering step reduced the dataset to 16,661 publications. Of these, 203 records (1.21%) were selected for full-text screening and assessment of eligibility. This case study represents a systematic review search with many hits and with a highly unbalanced number of publications being accepted for full-text screening.

1.11 Computational environment of benchmarking

AIM Review has been tested across major browsers—including Google Chrome, Mozilla Firefox, and Microsoft Edge—and on multiple devices, such as Windows PCs, Android phones, and tablets. Benchmarking was conducted on a single desktop computer equipped with an Intel Core i7-4770 CPU, 32 GB of RAM, and an NVIDIA GeForce GT 620 GPU. For these tests, AIM Review was run in Google Chrome (version 136.0) on Windows 10.

1.12 Methodology configuration for validation strategies

To assess and compare the performance of the vectorization and classification methods implemented in AIM Review, we set as constant a range of configuration parameters.

Configuration of vectorization methods: The configuration of the vectorization methods was identical in the validation of the active learning and supervised strategies. TF-IDF was configured to generate features at the word level, with no restrictions on minimum or maximum term frequency (set to 0 and 1, respectively). The output vectors were normalized using L2 normalization, ensuring unit length, and the maximum vector length was limited to 300 features. All other parameters were set to their default values as specified in the sklearn documentation. LSA of TF-IDF was configured to generate 50 components from a 500-feature extraction from TF-IDF. Other configuration parameters from TF-IDF were identical to the original TF-IDF implementation, while other LSA parameters were set to their default values as specified in the sklearn documentation. SPECTER2 was configured to merge titles and abstracts prior to vectorization. The pretrained sentence transformer was used to generate uniform 768-dimensional embeddings, with no additional configuration parameters selected. All other settings were retained as specified in the default pretrained model. All-MiniLM-L6-v2 was configured to merge titles and abstracts prior to vectorization. The pretrained sentence transformer was applied to generate uniform 384-dimensional embeddings, with no configuration parameters selected. All other settings were maintained as per the default pretrained model. Universal BERT was configured to merge titles and abstracts before vectorization. The pretrained universal sentence encoder was used to produce uniform 512-dimensional embeddings, with no additional configuration parameters selected. All other settings followed the default pretrained model configuration. Doc2Vec was configured using the PV-DBOW (Distributed Bag of Words) model, where titles and abstracts were concatenated prior to vectorization. The Adam optimizer was employed with Mean Squared Error (MSE) as the loss function and Tanh as the activation function. The model was trained over 100 epochs with a batch size of 16, balancing computational efficiency and convergence. The vector size was set to 100, defining the dimensionality of the document embeddings, while the context window was set to 5 to capture surrounding word

dependencies. The vocabulary was capped at 100 most frequent terms, and max padding was applied to standardize input sequence lengths to the longest title and abstract.

The configuration of the classification methods was distinct in the active learning algorithm and the supervised learning algorithm since due to a lack of robust cross-validation strategies, the active learning strategy does not include parameter hyperoptimization and all model parameters are preselected before training.

Configuration of cross-validation and classification methods in supervised learning: For the supervised learning validation, a nested cross-validation framework of 5 outer CV2 permutations, 5 outer CV2 folds, 1 inner CV2 permutation, and 5 inner CV2 folds was chosen. Logistic regression was configured with an L2 penalty and a primal formulation. The class weights were set to 'balanced' to address class imbalances. A range of values for the regularization parameter C (2^{-4} to 2^{4} with incremental exponent steps of 0.5) was used as hyperoptimization parameter to optimise model performance. The solver used for optimization was LBFGS. Linear SVMs were configured with an L2 penalty and primal formulation, similar to the logistic regression setup. The class weights were also set to 'balanced', and a grid of **C** values (2^{-4} to 2^{4} with incremental exponent steps of 0.5) was evaluated to identify the optimal regularization strength. The loss function used was squared hinge loss. Min-max normalization of the decision scores was applied to obtain an estimation of probability of relevance from SVM output decision scores. Sequential neural networks were configured with ReLU activation functions. The optimizer was Adam, known for its adaptive learning rate capabilities, with an initial learning rate of 0.001. The model was trained with a batch size of 32, and early stopping was enabled to prevent overfitting, with a validation fraction of 0.1 and a tolerance of 10 iterations without improvement before stopping. The class weights were set to 'balanced' to address any class imbalances. The maximum number of iterations was set to 100 to ensure sufficient training. The neural network structure was optimized between three possible structures: a single hidden layer of 500 units, two densely connected hidden layers of 50 units, or two densely connected hidden layers of 100 units. The input shape was adapted to the vector size, and the output layer was configured as two softmax units predicting the probability of relevance and irrelevance of publications. In all cases, BAC was used as the performance metric used to find the optimal parameters. The random state was set to 45 for the initial permutation and randomly changed for consecutive permutations. When multiple vectorization methods are applied simultaneously, we used stacked generalization or fusion of modalities (*vide supra*). With stacked generalization, a meta-model is then trained using the predictions from each base model to generate the final outcome prediction. Logistic regression, with the parameters defined above and a regularization parameter of 1 was selected as the meta-model.

Configuration of classification methods in active learning: For the validation of active learning methods, we utilized identical configuration parameters as for the supervised learning case but without a nested cross-validation framework and without hyperoptimization of parameters. Instead, classifiers are trained with all the available manual labels provided by the user to generate relevance scores for the unseen publications. Logistic regression was configured with an L2 penalty and a primal formulation. To address class imbalances, the class weights were set to "balanced." The regularization parameter C was set to 1 (no optimization). The LBFGS solver was used for optimization. Similarly, linear SVMs were configured with an L2 penalty and primal formulation, using the same balanced class weights and C parameter. The squared hinge loss

function was applied, and decision scores were normalized using min-max scaling to estimate probabilities of relevance from the SVM output. Sequential neural networks were designed with ReLU activation functions and trained using the Adam optimizer with an initial learning rate of 0.001. Training was conducted with a batch size of 32, and early stopping was implemented to mitigate overfitting, with a validation fraction of 0.1 and a tolerance of 10 epochs without improvement. The maximum number of iterations was capped at 100 to ensure adequate training. The network architecture was set to a single hidden layer of 500 units. When multiple vectorization methods are applied simultaneously, we used stacked generalization or fusion of modalities (*vide supra*). With stacked generalization, a meta-model is then trained using the predictions from each base model to generate the final outcome prediction. Logistic regression, with the parameters defined above and a regularization parameter of 1 was selected as the meta-model.

1.13 Performance metrics

To measure the performance of the active learning strategies across the case studies, we used the work saved over sampling measure (WSS). WSS quantifies the reduction in the number of publications that need to be screened when using active learning compared to conventional random sampling. It represents the percentage decrease in screening effort while achieving a specified recall level of relevant records. For example, $WSS_{95\%}$ indicates the workload reduction achieved while identifying 95% of the relevant records, meaning 5% of the relevant records are missed. In cases where it is crucial to retrieve all relevant records, $WSS_{100\%}$ measures the reduction in effort required while ensuring a complete recall of all relevant records:

$$\mathrm{WSS_X} = \frac{\mathrm{N_T} - \mathrm{N_{X\ recall}}}{\mathrm{N_T}} \times 100 \tag{22}$$

Where $\mathrm{N_T}$ is the total number of publications in the dataset and $\mathrm{N_{X\ recall}}$ is the number of publications that need to be reviewed to achieve X recall (e.g., $\mathrm{N_{95\%\ recall}}$).

To reliably measure WSS metric, we employed a randomized approach by permuting the publications in the dataset and initializing the active learning cycle (N = 5 permutations). Real labels are provided into the algorithm as the titles and abstracts of the publications were prompted by AIM Review. As the active learning process progresses, every time five new labels are added, the model is retrained, generating updated relevance scores for the publications. These scores are then used to sort the publications based on their relevance, prioritizing those most likely to be relevant. The cycle continues iteratively until all relevant publications have been screened. After that, we measure $\mathrm{WSS_{95\%}}$ and $\mathrm{WSS_{100\%}}$ to evaluate the percentage of work saved using the active learning strategy.

To measure the performance of the supervised learning strategies, we measured and compared the performance metrics measured by AIM Review (*vide supra*) over the validation sets (OOT), with focus on BAC and AUROC. Additionally, we measured the performance of the classifier in the held-out set (OOCV).

References

1. Jefferson, T., Gregg, C. & Piech, C. PyodideU: Unlocking Python Entirely in a Browser for CS1. in *Proceedings of the 55th ACM Technical Symposium on Computer Science Education V. 1* 583–589 (2024).
2. Smilkov, D. *et al.* Tensorflow. js: Machine learning for the web and beyond. *Proceedings of Machine Learning and Systems* **1**, 309–321 (2019).
3. Ostendorff, M., Rethmeier, N., Augenstein, I., Gipp, B. & Rehm, G. Neighborhood contrastive learning for scientific document representations with citation embeddings. *arXiv preprint arXiv:2202.06671* (2022).
4. Cohan, A., Feldman, S., Beltagy, I., Downey, D. & Weld, D. S. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180* (2020).
5. Conger, A. J. Integration and generalization of kappas for multiple raters. *Psychol Bull* **88**, 322–328 (1980).
6. Cohen, J. A coefficient of agreement for nominal scales. *Educ Psychol Meas* **20**, 37–46 (1960).
7. Falotico, R. & Quatto, P. Fleiss' kappa statistic without paradoxes. *Qual Quant* **49**, 463–470 (2015).
8. Cer, D. Universal sentence encoder. *arXiv preprint arXiv:1803.11175* (2018).
9. Devlin, J. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
10. Le, Q. & Mikolov, T. Distributed representations of sentences and documents. in *International conference on machine learning* 1188–1196 (PMLR, 2014).
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. Distributed representations of words and phrases and their compositionality. *Adv Neural Inf Process Syst* **26**, (2013).
12. Mikolov, T. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* **3781**, (2013).
13. Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. & Lin, C.-J. LIBLINEAR: A library for large linear classification. *the Journal of machine Learning research* **9**, 1871–1874 (2008).
14. Chang, C.-C. & Lin, C.-J. LIBSVM: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.* **2**, (2011).
15. Bakermans-Kranenburg, M. J., Dagan, O., Cárcamo, R. A. & van IJzendoorn, M. H. Celebrating more than 26,000 adult attachment interviews: mapping the main adult attachment classifications on personal, social, and clinical status. *Attach Hum Dev* 1–38 doi:10.1080/14616734.2024.2422045.
16. Brodeur, S. *et al.* Why we need to pursue both universal and targeted prevention to reduce the incidence of affective and psychotic disorders: Systematic review and meta-analysis. *Neurosci Biobehav Rev* **161**, 105669 (2024).

Supplementary Tables

Table S1. Extended performance metrics of active learning and supervised learning pipelines using model stacking and fusion of features in case studies 4, 5 and 6.

			Active learning		Supervised learning							
Case	**Model**	**Ensemble**	**$WSS_{95\%}$ (%)**	**$WSS_{100\%}$ (%)**	**BAC OOT (%)**	**Sens. OOT (%)**	**Spec. OOT (%)**	**AUROC OOT**	**BAC OOCV (%)**	**Sens. OOCV (%)**	**Spec. OOCV (%)**	**AUROC OOCV**
Case 4	LR	Model stacking	93.53 [0.27]	91.29 [0.19]	94.75 [1.50]	90.48 [3.01]	99.01 [0.08]	1.00 [0.00]	98.21	98.80	97.62	0.99
		Fusion of features	92.35 [0.48]	76.66 [1.63]	83.56 [2.33]	67.62 [4.67]	99.50 [0.03]	0.99 [0.00]	98.26	98.80	97.72	0.99
	L-SVM	Model stacking	95.33 [0.13]	93.69 [0.09]	92.71 [1.79]	86.67 [3.56]	98.75 [0.13]	1.00 [0.00]	97.82	98.80	96.84	0.99
		Fusion of features	92.21 [0.50]	77.80 [1.42]	88.03 [2.30]	77.14 [4.67]	98.92 [0.11]	0.99 [0.00]	97.21	98.80	95.63	0.99
	SeqNN	Model stacking	-	-	92.85 [0.88]	86.67 [1.90]	99.03 [0.18]	1.00 [0.00]	96.85	98.80	94.90	0.99
		Fusion of features	-	-	70.76 [10.98]	41.90 [22.21]	99.62 [0.29]	0.77 [0.20]	86.39	91.57	81.22	0.91
Case 5	LR	Model stacking	39.73 [0.31]	26.30 [0.28]	77.54 [1.87]	63.16 [3.33]	91.92 [3.31]	0.87 [0.01]	73.32	68.85	77.78	0.79
		Fusion of features	43.01 [0.45]	31.78 [0.11]	70.90 [5.57]	65.26 [7.14]	76.54 [4.77]	0.77 [0.03]	75.42	80.33	70.51	0.82
	L-SVM	Model stacking	41.37 [0.30]	32.33 [0.08]	77.30 [2.92]	64.21 [5.16]	90.38 [2.72]	0.87 [0.01]	71.71	73.77	69.66	0.78
		Fusion of features	42.74 [0.12]	32.05 [0.48]	69.07 [3.53]	67.37 [6.98]	70.77 [3.73]	0.73 [0.04]	74.89	85.25	64.53	0.81
	SeqNN	Model stacking	-	-	78.66 [2.25]	75.79 [5.37]	81.54 [2.61]	0.86 [0.02]	67.58	77.05	58.12	0.73
		Fusion of features	-	-	71.98 [2.87]	74.74 [10.73]	69.23 [10.39]	0.76 [0.03]	68.12	91.80	44.44	0.73
Case 6	LR	Model stacking	80.46 [0.04]	59.81 [0.01]	81.98 [0.36]	69.48 [0.65]	94.48 [0.07]	0.94 [0.00]	89.32	91.60	87.03	0.95
		Fusion of features	79.04 [0.04]	79.04 [0.10]	76.80 [0.51]	57.47 [0.97]	96.14 [0.04]	0.92 [0.01]	77.48	62.30	92.65	0.93
	L-SVM	Model stacking	79.86 [0.11]	60.08 [0.12]	80.11 [1.15]	64.61 [2.27]	95.61 [0.03]	0.94 [0.00]	88.53	87.64	89.41	0.94
		Fusion of features	63.08 [0.07]	58.80 [0.02]	79.85 [2.41]	70.45 [4.87]	89.24 [0.04]	0.89 [0.01]	84.97	86.66	83.28	0.91
	SeqNN	Model stacking	-	-	75.25 [1.42]	55.52 [2.92]	94.99 [0.07]	0.86 [0.00]	86.35	93.41	79.30	0.93
		Fusion of features	-	-	69.15 [0.32]	40.26 [0.65]	98.05 [0.02]	0.92 [0.00]	85.78	86.82	84.74	93.31

Abbreviations: LR, logistic regression; L-SVM, linear support vector machine; SeqNN, sequential neural networks; WSS, work saved over sampling; BAC, balanced accuracy; Sens., sensitivity; Spec., specificity; OOT, out-of-training; OOCV, out-of-cross-validation; AUROC, area under the receiver operating characteristic curve

Table S2. Performance metrics of active learning and supervised learning pipelines using individual text vectorization methods.

			Active learning		Supervised learning			
Case	**Model**	**Vectorization**	**$WSS_{95\%}$ (%)**	**$WSS_{100\%}$ (%)**	**BAC OOT (%)**	**BAC OOCV (%)**	**AUROC OOT**	**AUROC OOCV**
Case 1	LR	TF-IDF	81.18 [2.54]	60.84 [4.03]	77.93 [1.60]	82.83	0.87 [0.01]	0.89
		LSA of TF-IDF	76.05 [1.66]	56.17 [2.72]	78.83 [1.22]	82.32	0.86 [0.01]	0.89
		SPECTER2	78.05 [2.40]	67.04 [4.09]	85.19 [1.28]	84.01	0.92 [0.01]	0.91
		UnivBERT	75.95 [2.10]	43.95 [2.98]	82.41 [1.04]	81.89	0.89 [0.01]	0.87
		Doc2Vec	77.21 [1.72]	55.96 [2.67]	79.82 [1.44]	81.49	0.88 [0.01]	0.89
		all-MiniLM-L6-v2	83.43 [1.57]	65.17 [4.20]	87.27 [0.49]	85.44	0.94 [0.003]	0.92
	L-SVM	TF-IDF	84.50 [2.61]	62.80 [4.54]	76.93 [1.14]	80.80	0.85 [0.01]	0.89
		LSA of TF-IDF	76.64 [2.19]	40.84 [0.51]	78.26 [0.59]	82.44	0.86 [0.01]	0.89
		SPECTER2	79.68 [2.74]	61.79 [6.20]	83.61 [0.99]	82.54	0.91 [0.01]	0.90
		UnivBERT	73.32 [3.35]	51.67 [6.39]	82.32 [0.91]	81.17	0.89 [0.01]	0.88
		Doc2Vec	80.20 [1.60]	61.79 [2.70]	77.96 [1.17]	80.32	0.86 [0.01]	0.87
		all-MiniLM-L6-v2	80.80 [2.41]	65.73 [6.45]	89.71 [0.91]	84.15	0.94 [0.003]	0.92
	SeqNN	TF-IDF	-	-	75.86 [2.93]	80.58	0.83 [0.01]	0.88
		LSA of TF-IDF	-	-	73.43 [2.33]	82.44	0.82 [0.01]	0.89
		SPECTER2	-	-	81.47 [1.19]	81.89	0.90 [0.01]	0.90
		UnivBERT	-	-	73.46 [1.94]	78.90	0.81 [0.03]	0.86
		Doc2Vec	-	-	75.92 [1.87]	78.50	0.85 [0.02]	0.87
		all-MiniLM-L6-v2	-	-	83.66 [2.00]	80.61	0.91 [0.01]	0.88
Case 2	LR	TF-IDF	76.12 [2.48]	50.32 [5.63]	76.23 [1.11]	77.29	0.84 [0.01]	0.85
		LSA of TF-IDF	54.81 [0.17]	33.22 [0.02]	79.03 [3.24]	77.71	0.87 [0.01]	0.85
		SPECTER2	91.61 [1.92]	71.73 [5.49]	82.89 [1.73]	84.91	0.91 [0.01]	0.92
		UnivBERT	46.81 [3.11]	18.81 [1.78]	65.75 [1.97]	77.22	0.71 [0.02]	0.81
		Doc2Vec	85.20 [1.61]	57.13 [3.92]	78.96 [1.76]	83.32	0.86 [0.02]	0.90
		all-MiniLM-L6-v2	79.75 [3.10]	45.42 [4.52]	81.27 [0.83]	77.64	0.89 [0.01]	0.86
	L-SVM	TF-IDF	53.80 [0.28]	22.20 [0.01]	77.71 [2.02]	78.24	0.85 [0.01]	0.87
		LSA of TF-IDF	52.90 [0.24]	24.93 [0.02]	79.06 [2.65]	77.82	0.87 [0.01]	0.86
		SPECTER2	90.52 [2.50]	65.00 [6.73]	82.05 [2.94]	84.89	0.90 [0.02]	0.91
		UnivBERT	36.67 [0.50]	15.50 [0.09]	65.67 [2.49]	73.68	0.71 [0.01]	0.79
		Doc2Vec	66.82 [0.18]	49.52 [0.05]	74.59 [1.98]	80.07	0.83 [0.02]	0.88
		all-MiniLM-L6-v2	51.47 [0.22]	18.03 [0.01]	81.25 [1.81]	79.2	0.89 [0.003]	0.86
	SeqNN	TF-IDF	-	-	73.96 [2.45]	72.72	0.79 [0.02]	0.79
		LSA of TF-IDF	-	-	76.47 [1.81]	75.72	0.86 [0.02]	0.84
		SPECTER2	-	-	82.87 [2.10]	83.99	0.91 [0.01]	0.92
		UnivBERT	-	-	62.30 [1.81]	74.75	0.67 [0.01]	0.80
		Doc2Vec	-	-	79.33 [1.38]	83.75	0.86 [0.02]	0.90
		all-MiniLM-L6-v2	-	-	76.76 [2.47]	75.82	0.86 [0.02]	0.82

Case 3	LR	TF-IDF	17.17 [0.41]	8.79 [0.01]	66.93 [2.36]	68.66	0.71 [0.02]	0.73
		LSA of TF-IDF	19.09 [0.50]	9.20 [0.01]	64.64 [1.83]	68.57	0.69 [0.02]	0.74
		SPECTER2	21.70 [0.98]	12.36 [0.01]	71.77 [1.06]	73.58	0.77 [0.01]	0.78
		UnivBERT	15.66 [0.38]	6.32 [0.01]	63.39 [1.89]	64.84	0.66 [0.02]	0.69
		Doc2Vec	18.96 [0.25]	10.30 [0.01]	66.10 [2.23]	70.22	0.72 [0.02]	0.76
		all-MiniLM-L6-v2	17.86 [0.41]	8.38 [0.02]	69.91 [1.99]	69.69	0.76 [0.01]	0.73
	L-SVM	TF-IDF	13.87 [0.23]	6.18 [0.01]	66.76 [2.67]	68.06	0.72 [0.01]	0.73
		LSA of TF-IDF	17.86 [1.13]	5.36 [0.01]	68.02 [1.96]	68.61	0.73 [0.01]	0.72
		SPECTER2	18.27 [4.40]	7.83 [0.48]	68.42 [1.91]	72.69	0.74 [0.01]	0.77
		UnivBERT	14.01 [1.11]	2.88 [0.02]	62.54 [1.14]	66.83	0.66 [0.02]	0.702
		Doc2Vec	18.00 [0.62]	7.14 [0.10]	64.09 [3.62]	66.85	0.69 [0.02]	0.71
		all-MiniLM-L6-v2	16.35 [0.73]	7.83 [0.11]	71.46 [1.16]	71.15	0.75 [0.01]	0.74
	SeqNN	TF-IDF	-	-	62.22 [1.57]	70.86	0.69 [0.01]	0.74
		LSA of TF-IDF	-	-	63.33 [2.38]	67.86	0.6883 [0.01]	0.73
		SPECTER2	-	-	69.21 [1.07]	72.37	0.75 [0.02]	0.77
		UnivBERT	-	-	60.60 [1.83]	68.02	0.64 [0.03]	0.71
		Doc2Vec	-	-	66.03 [1.43]	71.13	0.70 [0.02]	0.78
		all-MiniLM-L6-v2	-	-	68.07 [0.92]	71.89	0.74 [0.01]	0.75

Table S3. Extended performance metrics of supervised learning pipelines using individual text vectorization methods.

			Supervised learning					
Case	**Model**	**Vectorization**	**BAC OOT (%)**	**Sens. OOT (%)**	**Spec. OOT (%)**	**BAC OOCV (%)**	**Sens. OOCV (%)**	**Spec. OOCV (%)**
Case 1	LR	TF-IDF	77.93 [1.60]	76.92 [3.24]	76.92 [3.24]	82.83	81.71	83.96
		LSA of TF-IDF	78.83 [1.22]	73.33 [2.05]	84.32 [0.45]	82.32	89.63	75.01
		SPECTER2	85.19 [1.28]	93.33 [2.61]	77.04 [0.42]	84.01	85.98	82.05
		UnivBERT	82.41 [1.04]	85.13 [1.92]	79.69 [0.31]	81.89	86.59	77.19
		Doc2Vec	79.82 [1.44]	85.64 [3.08]	73.99 [0.40]	81.49	76.22	86.76
		all-MiniLM-L6-v2	87.27 [0.49]	99.49 [1.03]	75.05 [0.07]	85.44	83.54	87.34
	L-SVM	TF-IDF	76.93 [1.14]	75.38 [2.05]	78.48 [0.36]	80.80	78.05	83.55
		LSA of TF-IDF	78.26 [0.59]	72.31 [1.03]	84.22 [0.30]	82.44	90.85	74.04
		SPECTER2	83.61 [0.99]	89.23 [2.51]	77.99 [0.93]	82.54	87.80	77.28
		UnivBERT	82.32 [0.91]	90.26 [1.92]	74.38 [0.28]	81.17	85.37	76.98
		Doc2Vec	77.96 [1.17]	76.41 [2.99]	79.50 [0.75]	80.32	79.27	81.37
		all-MiniLM-L6-v2	89.71 [0.91]	94.87 [1.62]	84.55 [0.27]	84.15	88.41	79.90
	SeqNN	TF-IDF	75.86 [2.93]	82.56 [4.97]	69.16 [5.68]	80.58	91.46	69.69
		LSA of TF-IDF	73.43 [2.33]	60.00 [8.67]	86.86 [4.92]	82.44	90.85	74.04
		SPECTER2	81.47 [1.19]	80.00 [8.94]	82.93 [7.71]	81.89	81.71	82.07
		UnivBERT	73.46 [1.94]	67.69 [9.12]	79.22 [6.77]	78.90	82.32	75.49
		Doc2Vec	75.92 [1.87]	82.56 [2.99]	69.28 [2.71]	78.50	79.27	77.73

		all-MiniLM-L6-v2	83.66 [2.00]	90.26 [3.40]	77.06 [5.99]	80.61	74.39	86.83
Case 2	LR	TF-IDF	76.23 [1.11]	91.85 [2.77]	60.60 [0.59]	77.29	88.96	65.62
		LSA of TF-IDF	79.03 [3.24]	75.56 [6.87]	82.51 [0.47]	77.71	84.05	71.37
		SPECTER2	82.89 [1.73]	83.70 [3.78]	82.07 [0.46]	84.91	84.05	71.62
		UnivBERT	65.75 [1.97]	88.15 [4.32]	43.36 [0.51]	77.22	82.82	71.62
		Doc2Vec	78.96 [1.76]	77.78 [3.31]	80.15 [0.42]	83.32	77.30	89.33
		all-MiniLM-L6-v2	81.27 [0.83]	82.96 [1.81]	79.59 [0.37]	77.64	69.94	85.34
	L-SVM	TF-IDF	77.71 [2.02]	88.89 [4.06]	66.54 [0.89]	78.24	85.89	70.60
		LSA of TF-IDF	79.06 [2.65]	75.56 [5.54]	82.56 [0.32]	77.82	81.60	74.05
		SPECTER2	82.05 [2.94]	82.22 [6.37]	81.89 [0.93]	84.89	83.44	86.35
		UnivBERT	65.67 [2.49]	54.81 [4.32]	76.53 [0.94]	73.68	80.3	67.00
		Doc2Vec	74.59 [1.98]	82.96 [5.54]	66.22 [1.75]	80.07	74.23	85.90
		all-MiniLM-L6-v2	81.25 [1.81]	90.37 [2.96]	72.14 [1.05]	79.2	76.07	82.33
	SeqNN	TF-IDF	73.96 [2.45]	77.78 [4.06]	70.14 [1.19]	72.72	71.78	73.65
		LSA of TF-IDF	76.47 [1.81]	80.00 [4.44]	72.93 [4.87]	75.72	66.87	84.57
		SPECTER2	82.87 [2.10]	84.44 [4.91]	81.30 [0.90]	83.99	85.28	82.71
		UnivBERT	62.30 [1.81]	60.74 [11.14]	63.86 [7.95]	74.75	79.75	69.75
		Doc2Vec	79.33 [1.38]	75.56 [1.81]	83.11 [2.28]	83.75	84.66	82.85
		all-MiniLM-L6-v2	76.76 [2.47]	85.19 [3.31]	68.33 [2.07]	75.82	67.48	84.15
Case 3	LR	TF-IDF	66.93 [2.36]	60.24 [2.82]	73.62 [2.32]	68.66	59.29	78.02
		LSA of TF-IDF	64.64 [1.83]	60.00 [2.10]	69.28 [2.49]	68.57	70.83	66.30
		SPECTER2	71.77 [1.06]	76.00 [2.18]	67.54 [1.74]	73.58	69.87	77.29
		UnivBERT	63.39 [1.89]	75.76 [0.94]	51.01 [3.09]	64.84	69.23	60.44
		Doc2Vec	66.10 [2.23]	82.35 [2.78]	49.86 [3.38]	70.22	83.65	56.78
		all-MiniLM-L6-v2	69.91 [1.99]	72.00 [1.73]	67.83 [2.81]	69.69	75.64	63.74
	L-SVM	TF-IDF	66.76 [2.67]	60.47 [3.21]	73.04 [2.69]	68.06	67.63	68.50
		LSA of TF-IDF	68.02 [1.96]	58.35 [3.76]	77.68 [2.69]	68.61	67.63	69.60
		SPECTER2	68.42 [1.91]	73.65 [3.38]	63.19 [3.25]	72.69	72.12	73.26
		UnivBERT	62.54 [1.14]	83.06 [2.18]	42.03 [1.59]	66.83	66.99	66.67
		Doc2Vec	64.09 [3.62]	60.94 [4.43]	67.25 [2.84]	66.85	64.10	69.60
		all-MiniLM-L6-v2	71.46 [1.16]	73.65 [1.20]	69.28 [1.69]	71.15	80.77	61.54
	SeqNN	TF-IDF	62.22 [1.57]	70.82 [3.19]	53.62 [4.85]	70.86	66.99	74.73
		LSA of TF-IDF	63.33 [2.38]	71.29 [3.38]	55.36 [3.60]	67.86	73.08	62.64
		SPECTER2	69.21 [1.07]	66.82 [3.44]	71.59 [3.50]	72.37	79.17	65.57
		UnivBERT	60.60 [1.83]	78.59 [7.68]	42.61 [6.65]	68.02	65.71	70.33
		Doc2Vec	66.03 [1.43]	76.71 [2.92]	55.36 [4.80]	71.13	63.14	79.12
		all-MiniLM-L6-v2	68.07 [0.92]	77.88 [2.28]	58.26 [2.81]	71.89	80.77	63.00

Supplementary Figures

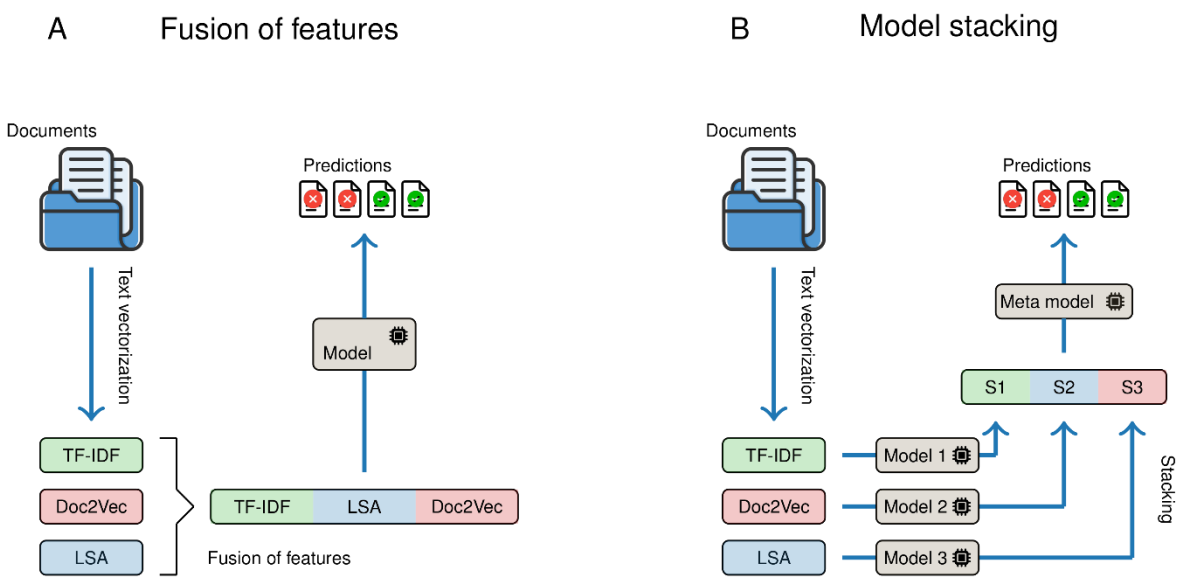

Figure S1. Ensemble strategies in AIM Review. (A) Example pipeline of feature fusion as ensemble method. Feature fusion combines different text representations (TF-IDF, Doc2Vec, LSA) into a single feature set for a single model that generates the document predictions of relevance. (B) Example pipeline of model stacking as ensemble method. Model stacking uses multiple base models trained with different representations to generate decision scores (S1, S2 and S3). These decision scores are then used by a meta model to combine their predictions and generate a final prediction of document relevance.

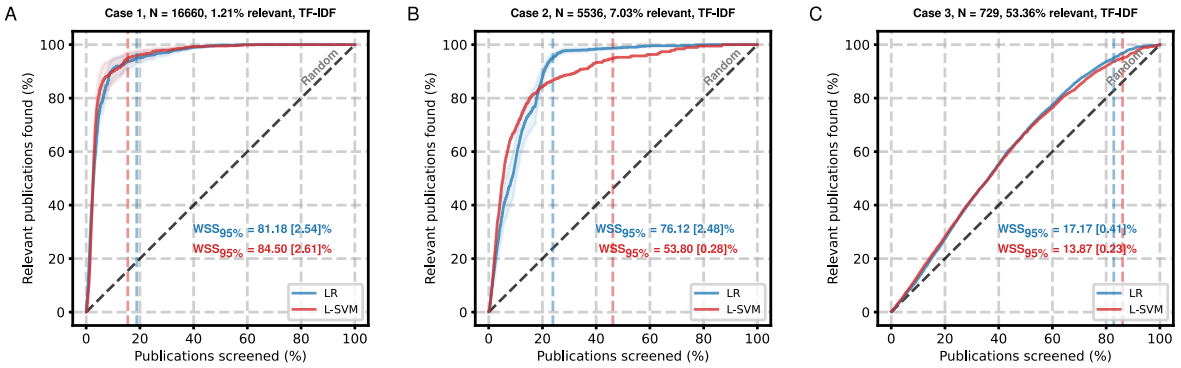

Figure S2. Active learning simulation results with TF-IDF features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained with TF-IDF features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used.

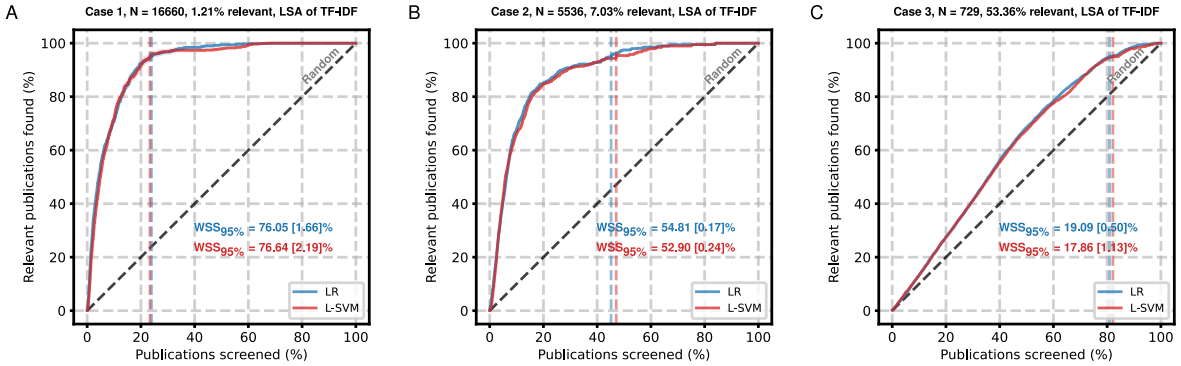

Figure S3. Active learning simulation results with LSA of TF-IDF features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained with LSA of TF-IDF features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used.

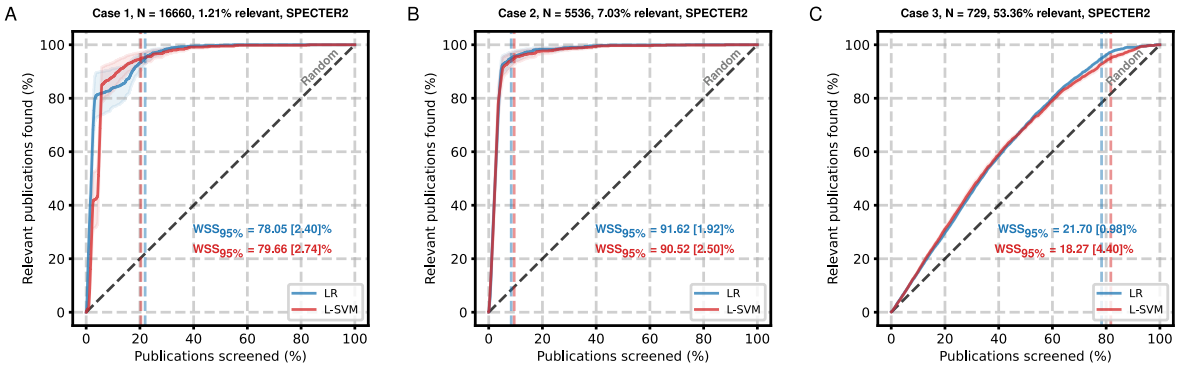

Figure S4. Active learning simulation results with SPECTER2 features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained SPECTER2 features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used.

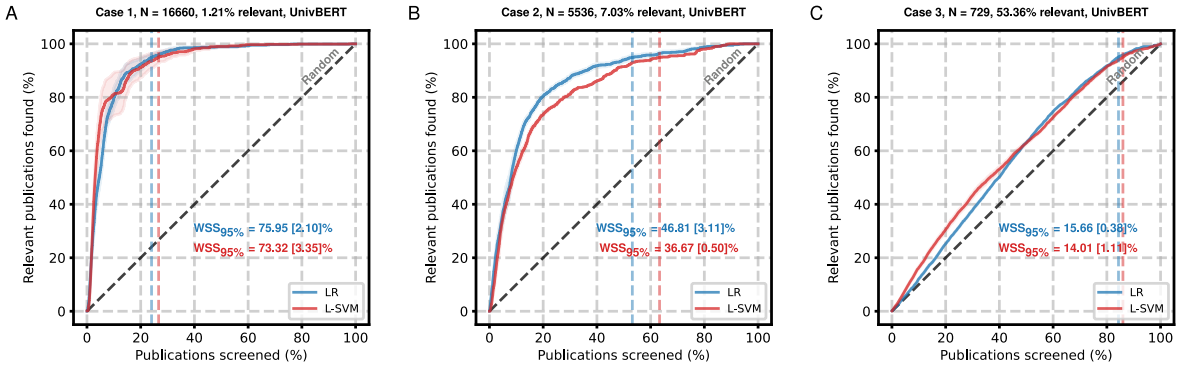

Figure S5. Active learning simulation results with univBERT features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained univBERT features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. . The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used. der of publications was randomized and no active learning was used.

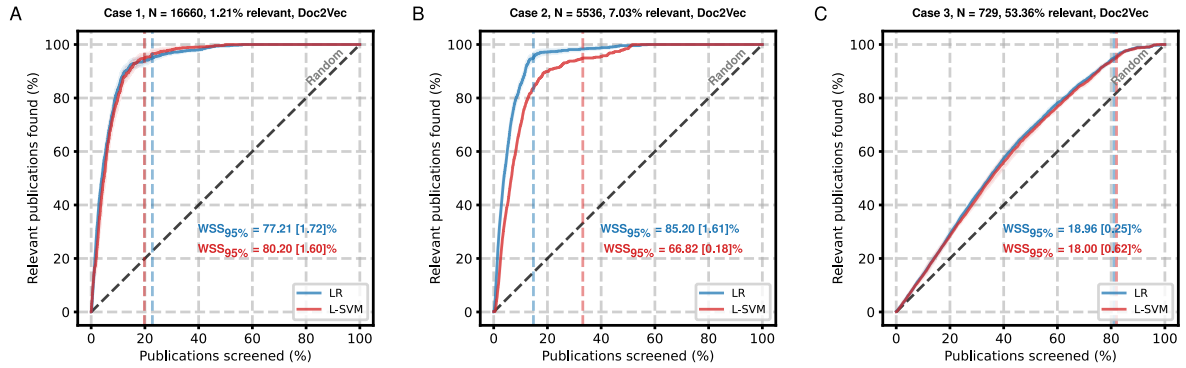

Figure S6. Active learning simulation results with Doc2Vec features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained Doc2Vec features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used.

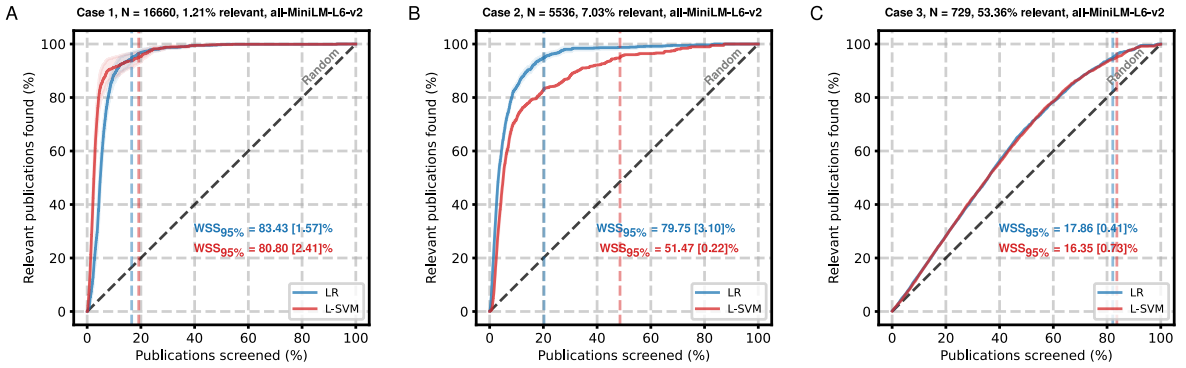

Figure S7. Active learning simulation results with all-MiniLM-L6-v2 features. Mean and standard deviation of relevant publications identified during the screening simulation versus the total publications screened in (A) case study 1, (B) case study 2, and (C) case study 3. The selected classifiers (LR or L-SVM) are trained all-MiniLM-L6-v2 features from titles and abstracts. Insets in the figures display the mean and standard deviation of $WSS_{95\%}$ for each classifier type. The grey dashed line shows the expected mean number of relevant publications found if the order of publications was randomized and no active learning was used.

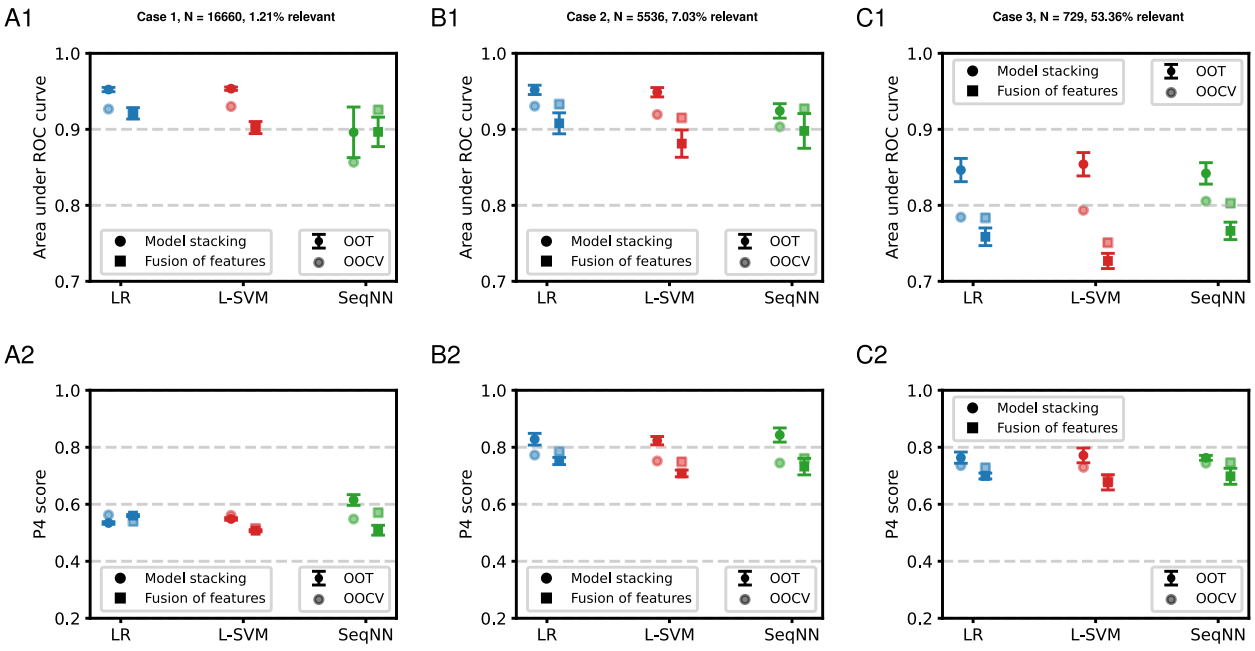

Figure S8. Area under the ROC curve and P4 score of supervised learning results. The area under the ROC curve (top) and P4 score (bottom) for (A) case study 1, (B) case study 2 and (C) case study 3 were calculated at the OOT (mean and standard deviation, n = 5 outer permutations) and OOCV using stacking generalization or concatenation of features. In the stacked generalization approach, the chosen classifiers (LR, L-SVM or SeqNN) are independently trained using features derived from each text vectorization method (TF-IDF, LSA, Doc2Vec, univBERT, all-MiniLM-L6-v2, and SPECTER2). A meta-model (LR with regularization parameter C = 1) is then trained on the predictions of the base models to produce the final predictions. In the feature fusion approach, the features extracted from all text vectorization methods are concatenated and used to train a single model (LR, L-SVM, or SeqNN) for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

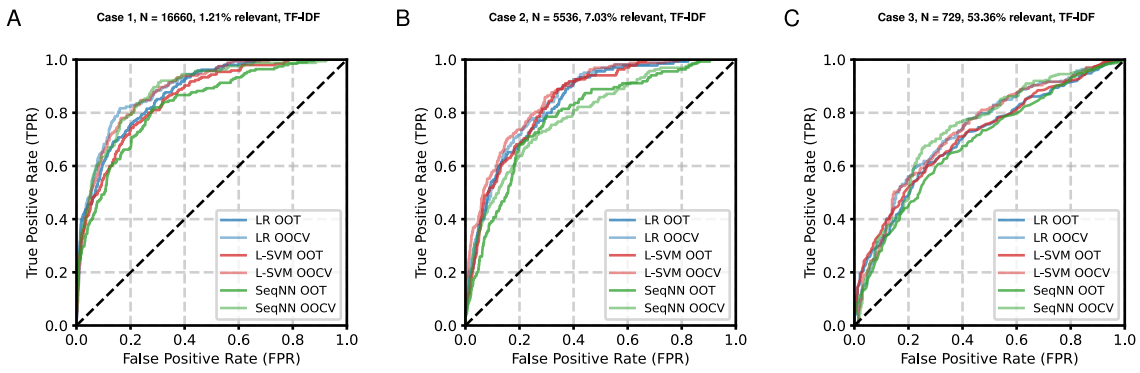

Figure S9. Supervised learning ROC results with TF-IDF features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using TF-IDF features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

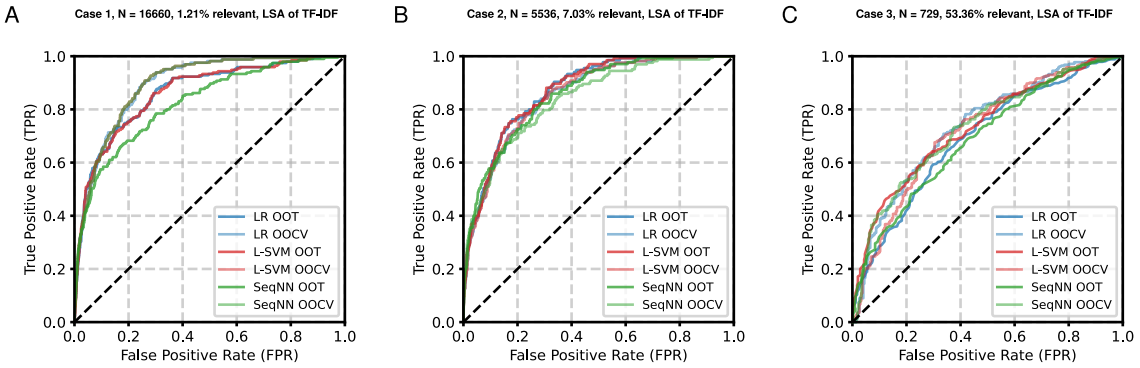

Figure S10. Supervised learning ROC results with LSA of TF-IDF features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using LSA of TF-IDF features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

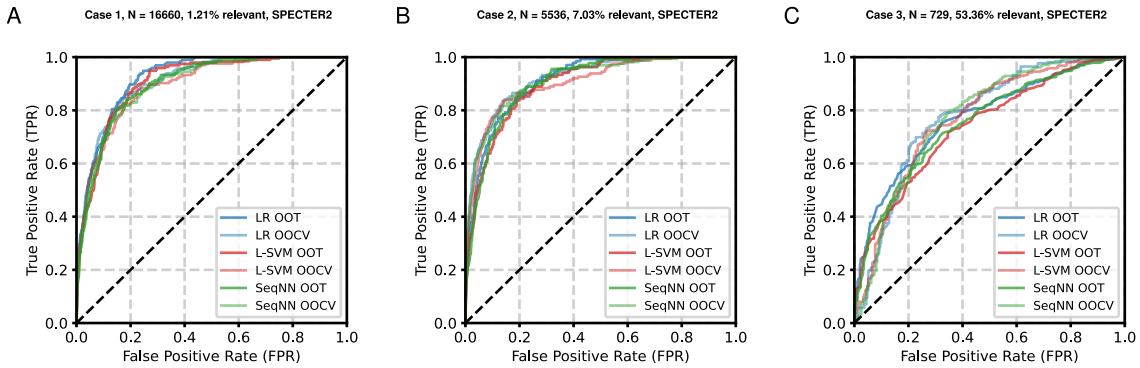

Figure S11. Supervised learning ROC results with SPECTER2 features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using SPECTER features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

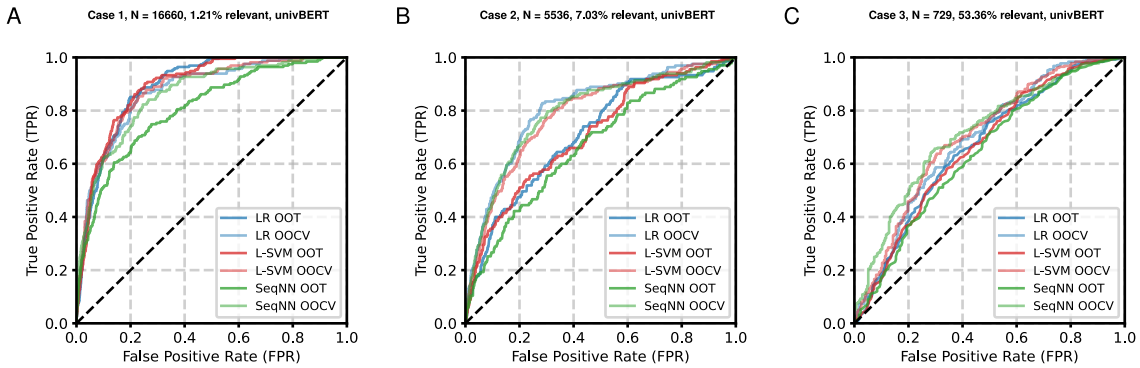

Figure S12. Supervised learning ROC results with UnivBERT features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using UnivBERT features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

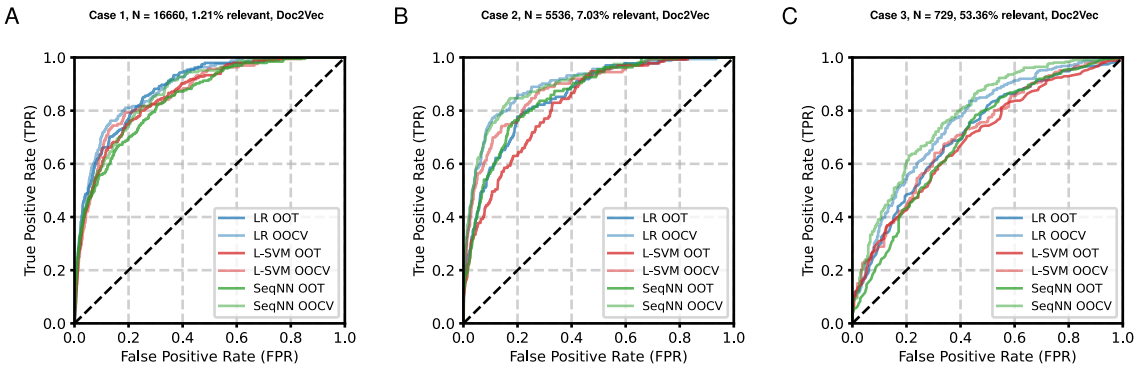

Figure S13. Supervised learning ROC results with Doc2Vec features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using Doc2Vec features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

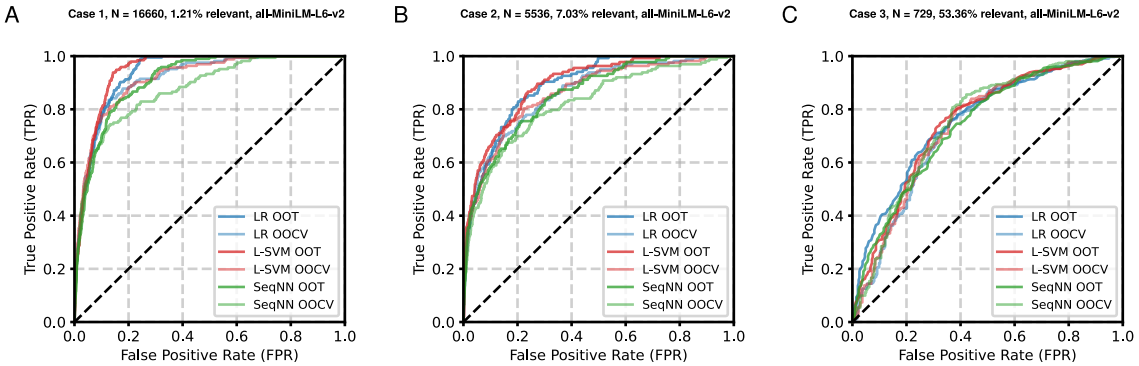

Figure S14. Supervised learning ROC results with all-MiniLM-L6-v2 features. ROC curves at OOT (20% of publications) and OOCV level (80% of publications) are shown for (A) case study 1, (B) case study 2 and (C) case study 3 using TF-IDF features. Classifiers (LR, L-SVM or SeqNN) are trained using all-MiniLM-L6-v2 features for predicting publication relevance. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample.

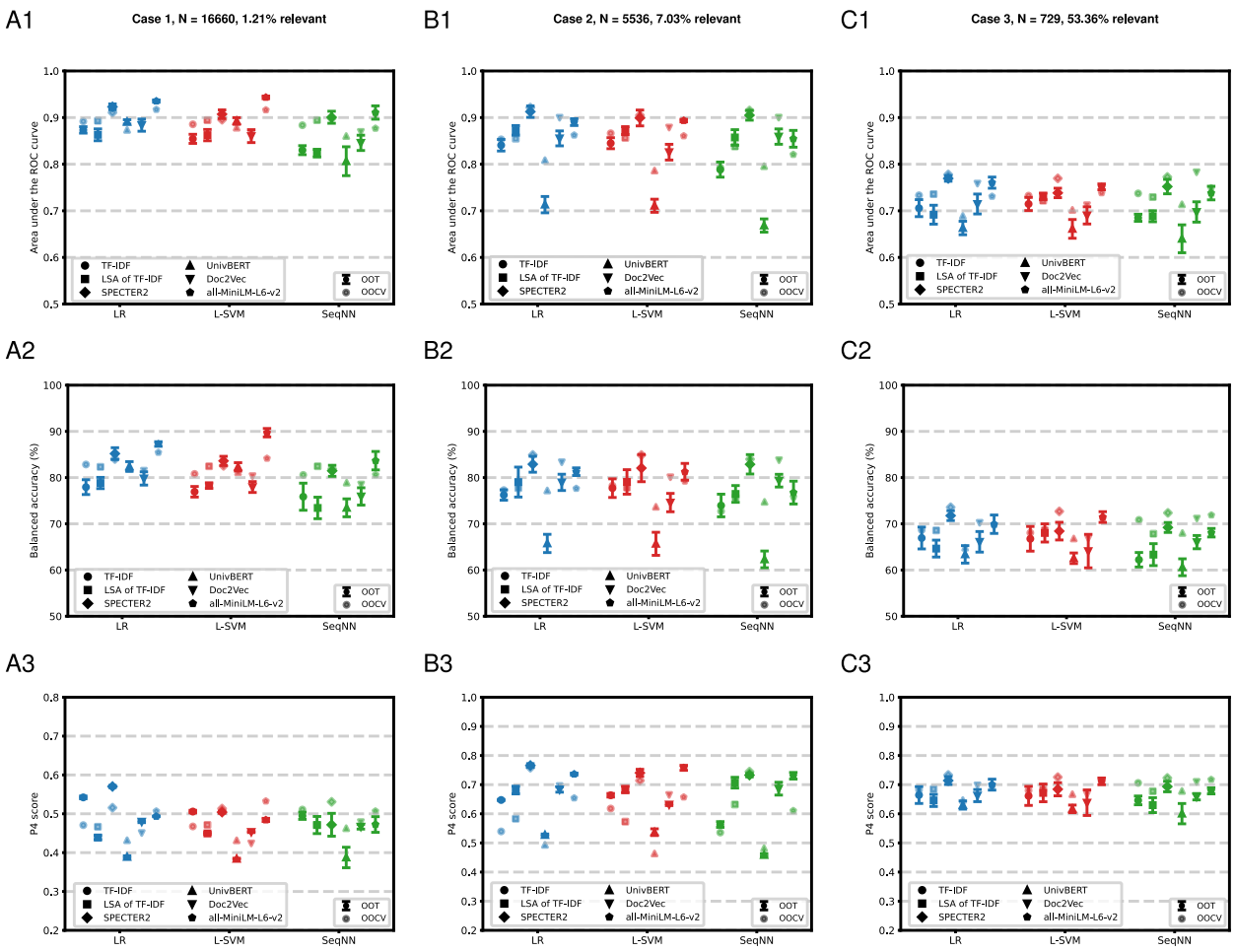

Figure S15. Area under the ROC curve, balanced accuracy and P4 score of supervised learning models. The area under the ROC curve (top), balanced accuracy (middle) and P4 score (bottom) for (A) case study 1, (B) case study 2 and (C) case study 3 were calculated at the OOT (mean and standard deviation, n = 5 outer permutations) and OOCV levels using three classifiers: LR, L-SVM and SeqNN and the features produced by one text vectorization method (TF-IDF, LSA, Doc2Vec, univBERT, all-MiniLM-L6-v2, or SPECTER2), displayed with different shapes. Models are trained and validated at the OOT level using a NCV framework of 5 outer permutations, 5 outer folds and 5 inner folds. The final optimum model is trained with all training publications and tested in the OOCV sample. Balanced accuracies and P4 scores are calculated by applying a threshold probability of relevance of 0.5.