Utrecht University

Out with the Old and in with the New? - A Comparison of Classical vs. State-of-the-Art Feature Extractors in the Context of Systematic Reviews

Ana Caklovic

Master's Thesis

July 2022

Programme – Applied Data Science
Keywords – Feature Extraction, Systematic Reviews, Transformers, Text Classification

Supervisor: Jelle Teijema
First Examiner: Rens van de Schoot
Second Examiner: Marco van Leeuwen

Abstract

Feature extraction is the process of transforming the raw data into features that the model will be trained on while trying to preserve as much information as possible. Choosing the proper feature extractor can greatly affect the performance of a classifier. Feature extraction has evolved from the older techniques such as tf-idf and Doc2Vec to transformers that have already been pre-trained on large corpora. However, although the newer techniques seem promising, it is not always clear when and why one feature extractor may outperform another. The aim of this study is to examine if state-of-the-art feature extractors (i.e., transformers like RoBERTa, MPNET, and SPECTER) can outperform classical feature extractors (i.e., tf-idf and Doc2Vec) when classifying systematic reviews as relevant or irrelevant. The study involved running multiple simulations with the ASReview software to see how well the different feature extractors (in combination with various classifiers) classified research articles as relevant or irrelevant. The results indicated that a tf-idf feature extractor, in combination with a Naive Bayes classifier, outperformed all other combinations, including the sentence transformer feature extractors.

All of the code for this study can be found on GitHub: https://github.com/acaklovic/Comparison-of-feature-extractors-using-ASReview/tree/main

Table of Contents

1. Introduction

1.1) Motivation and Context

In recent years, there has been a large influx of academic research papers. The purpose of systematic reviews is to summarize and analyze research studies that focus on the same topic of interest. However, in order for systematic reviews to avoid bias, they must attempt to find all possible relevant research papers on the topic (O'Mara-Eves, Thomas, McNaught, Miwa, & Ananiadou, 2015). The task can be made easier with the use of machine learning and, more specifically, text classification - the machine learning task of assigning text to predefined categories.

Systematic reviews are a laborious process, with the average manual screening time lasting 2.4 years and with 23% of systematic reviews needing to be updated after two years due to the constant flow of new research (Kontonatsios, Spencer, Matthew, & Korkontzelos, 2020). Machine learning can ameliorate the systematic process in numerous ways. When a single screener performs a systematic review, bias may be introduced, but asking more researchers to perform the screening also may not always be feasible (O'Mara-Eves, 2015). Active learning, in combination with machine learning text classification, can be used to avoid bias, increase accuracy, and decrease screening time while keeping the researcher in the loop.

ASReview is an AI active learning system that uses the titles and abstracts of research papers to classify a set of papers as relevant or irrelevant for the researcher (van de Schoot et al., 2021). This study will be using the ASReview software to examine the effects of different feature extraction methods on the performance of the machine learning classifier. ASReview allows the user to implement various feature extraction methods in order to reach optimal performance and find as many relevant articles as possible in the least amount of screening time. Thus, the feature extraction methods are a vital part of the ASReview process and determine how successful a model's performance is.

ASReview:

The ASReview pipeline consists of four main parts: 1) data preparation + feature extraction, 2) training the model, 3) tuning the model, and 4) repeating the cycle until all relevant records are found (full details of the steps in the ASReview pipeline can be seen in Figure 1). The pipeline utilizes a process called active learning, which is meant to keep the "human-in-the-loop" by asking them to repeatedly label studies that then become part of the training set, and then using the new training set to retrain the model (van de Schoot et al., 2021).

Active learning involves the machine learning model interacting with the reviewer in a cyclical process where the accuracy of the model improves with every iteration. The initial training set is created by the reviewer labeling a certain amount of articles as relevant or irrelevant (in the case of ASReview, the training set can be chosen randomly or through keyword search). Then, the model ranks the articles from most to least relevant and asks the reviewer to label the article ranked highly, thus expanding the training set during the next run. This repeats until a pre-determined stopping criterion is reached (O'Mara-Eves, 2015). Thus, the advantage of active learning is the decrease in workload (how many labeling tasks the system has to go through). In

addition, the reviewer maintains some control during the process since they are informing the machine which articles they think are relevant or not (van de Schoot et al., 2021).

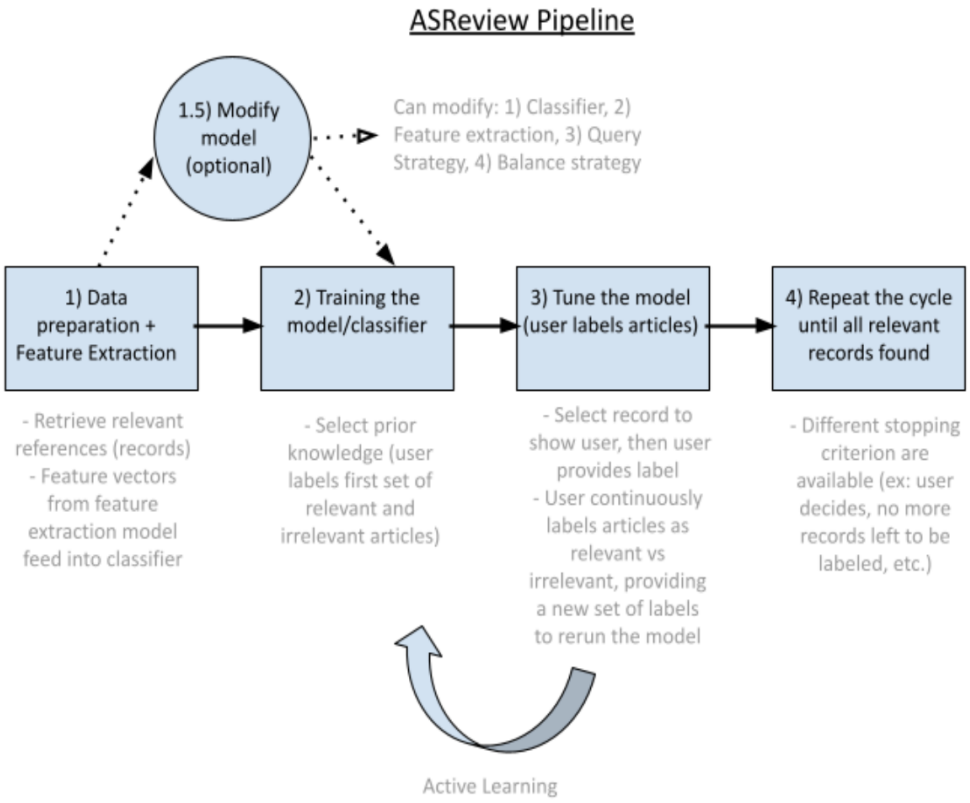

Figure 1: ASReview Pipeline - The software uses active learning in order to build machine learning models that find all relevant articles

An important step in the ASReview pipeline is the extraction of features from the raw text data, which is performed during the data preparation stage (before the active learning cycle and before the model training begins).

1.2) Literature review

Feature Extraction:
Feature extraction transforms the raw text input into a usable numerical vector representation of "features" that are then fed into the machine learning classifier (Kontonatsios et al., 2020). Feature extractor selection can be just as important as classifier selection when it comes to ensuring the optimal performance of the model. Incorporating the correct feature extractor can decrease the search space of the classifier by focusing only on the most important features. Over the years, feature extraction models have moved from tf-idf and Doc2Vec to newer transformer models like SBERT (and variants like RoBERTa, XLNET, and MPNet).

Transformers:
When examining the plethora of language models in the academic field, newer models such as neural networks are competing to overtake the more classical models such as tf-idf. Neural networks, such as recurrent neural networks (RNNs), use a layered architecture to work with dense word vectors and automatically learn features that capture semantic and syntactic information (unlike the high-dimensional and computationally heavy vectors of tf-idf) (Kalyan et al., 2022). However, in recent years an even newer state-of-the-art architecture has emerged. Transformers are the next generation of neural networks with a unique architecture that solely focuses on the attention mechanism. The transformer model uses transfer learning by training a model on one task and then applying it to a completely different task (Acheampong et al., 2021). When a model is pre-trained, it means that it has already been trained using a different task and dataset. Transfer learning employs pre-training so that the weights from training on one task are then used for a new task to make the completion of the new task faster and more accurate (Qiu et al., 2020).

Certain layers of the transformer architecture are similar to RNNs, but the inputs that feed into these neural layers have changed to more dense embeddings that can contain more information. This is done by utilizing three new components in the architecture: 1) Positional encoding, 2) Self-attention, and 3) Multi-head attention (Briggs, n.d.).

Positional encoding means that the order of the words in the sentence (also called sequence) is considered. The network structure includes “encoder” and “decoder” layers in the place of the recurrent/convolutional layers of a neural network so that the syntax of the sequence of text is not lost (Briggs, n.d.; Vaswani et al., 2017). The encoder layer takes the input vector and uses self-attention to make sure the global context of the word (how the meaning of the word changes across all contexts) is coded. The advantage is that complex language nuances can then be encoded in these vectors, making for a more accurate model (Kalyan et al., 2022).

The attention mechanism is a key aspect of the architecture because it ensures the model is focusing only on the most relevant parts of the data (Acheampong et al., 2021). More specifically, the attention mechanism is self-attention, which codes the different positions of the sequence so that contextual information is retained (Vaswani et al., 2017). Self-attention provides three key advantages in comparison to the recurrent/convolutional layers of neural networks: 1) better computational complexity, 2) improved long-range sequence dependencies (interactions between far apart words), and 3) parallel processing of input sequences (Acheampong et al., 2021; Vaswani et al., 2017).

Input sequences being processed in parallel leads to the concept of multi-head attention, a vital component of the transformer architecture formed from self-attention. Multi-head attention is the stacking of self-attention layers. Self-attention means the meaning (semantics) of a word is found by looking at that word in only one context. However, multi-head attention performs self-attention in parallel so that the representation of a word changes depending on the position/context it is in (Terechshenko et al., 2020; Kalyan et al., 2022).

The question of how all these concepts are pieced together is further explained in the original paper introducing the transformer architecture, *Attention is All You Need*, by Vaswani et al., 2017. The representation of the transformer architecture can be seen in the diagram below:

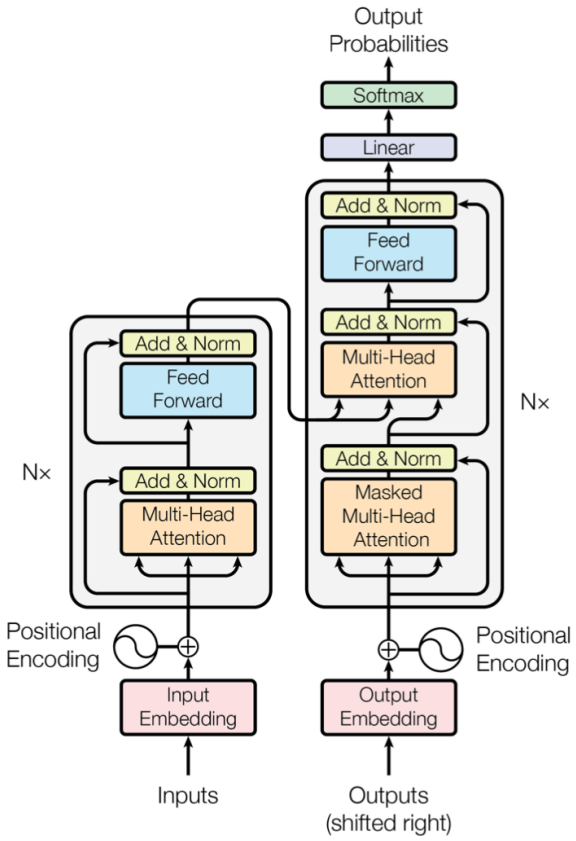

Figure 2: Transformer model architecture from *Attention is all you Need* by Vaswani et al., 2017

The diagram above shows that transformers consist of the same feed-forward layers that are found in neural networks but indicates the key differences discussed above. The general picture is that the presence of the positional encoding mechanism allows for the order of a sequence (syntax) to be maintained (Vaswani et al., 2017). In addition, the masked multi-head attention mechanism focuses on only the most relevant information in the input in order to cut down on computation costs while still maintaining accuracy. This means that the inputs feeding into the neural layers are more information-rich than in a classic neural network like an RNN (Vaswani et al., 2017).

Additional Feature Extraction Methods:
ASReview offers a couple of classical feature extractor techniques. A basic overview of the feature extractors is given below, along with an explanation of specific transformer models that will be evaluated in the study:

There are four basic, but very important, differences that can be discerned between the feature extraction models: 1) Syntax (the arrangement of the words/sentence), 2) Semantics (the meaning of the words/sentence), 3) Context (the meaning may change depending on the

surrounding words/sentences), and 4) Out of Vocabulary (OOV) (words that are not in the training set but are in the test set) (Naseem et al., 2021). Each of these four differences will be examined for each feature extractor below, along with other advantages and disadvantages:

1) Classical: Tf-idf

Tf-idf is one of the first feature extractors to be implemented. Tf-idf is a categorical word representation, which is one of the easiest ways to represent text because it simply uses symbology. Unfortunately, categorical representations create sparse vectors of features and thus suffers from the curse of dimensionality. In this model, term frequency is calculated as how often a word appears in a document. The Inverse Document Frequency assigns bigger weights to terms with a higher (or lower) frequency so that common words don't have as large of an impact (Naseem et al., 2021). The full equation is:

$$tf\text{-}idf\ (t, d) = tf\ (t, d) * idf(t),$$

where t is the term in document d for the total number of documents n, so that $idf\ (t) = log\ (n / df\ (t)) + 1$ and $df\ (t)$ is the document frequency of t. In addition, due to being a categorical word representation, tf-idf does not consider syntax, semantics, context, or OOV sentences. The benefits of tf-idf include the fact that it is easy to compute, and common terms do not impact results (Naseem et al., 2021).

2) Classical: Word2Vec (and Doc2Vec)

Word2Vec employs word embeddings and is in fact a continuous word representation, which solves the curse of dimensionality and lack of syntax and semantics that are associated with tf-idf. Word embeddings are vectors representing text that are based on the rule that if certain words are used in a similar context, then they should have the same meaning. Note that Doc2Vec is simply a variation of Word2Vec that processes sentences instead of words (Tabinda Kokab et al., 2022; Naseem et al., 2021).

Word2Vec consists of two shallow, hidden neural network layers that turn each word into a vector representation (or in the case of Doc2Vec, it turns the sentence into a vector). When creating the vector representation, words that are variations of each other (ex: "run" and "running") are near each other in the vector space. The issue with Word2Vec and Doc2Vec is that these models need a large corpus in order to perform well (Naseem et al., 2021). Word2Vec and Doc2Vec fail to recognize that the meaning of a word/sentence can change due to context (Terechshenko et al., 2020; Qiu et al., 2020).

State-of-the-art: SBERT and its variants

SBERT is a transformer, more specifically a sentence transformer, that incorporates the foundation of neural networks with an attention mechanism in order to create a self-supervised model. SBERT is based off of BERT (Bidirectional Encoder Representations from Transformers) and it creates sentence embeddings by adding a pooling operation to the output of the BERT model (Reimers & Gurevych, 2019). BERT was trained on a huge corpus consisting of Book-Corpus and the English Wikipedia (Liu et al., 2019). BERT is a self-supervised model, meaning that it was trained on raw texts instead of a labelled dataset, and instead uses certain training tasks to create inputs and labels (Hugging Face, n.d.).

The two tasks that are used to train BERT: 1) Masked language model (MLM) and 2) Next sentence prediction (NSP). An MLM is when 15% are randomly replaced with the "mask" token and the model then predicts the masked tokens. In the NSP task, the model has to recognize when the second sentence follows the first, in a pair of given sentences (Naseem et al., 2021).

One advantage of BERT is that it uses bidirectional self-attention, which means that the model reads text from both directions (Cohan et al., 2020). Unlike tf-idf and Word2Vec, BERT considers the context (neighboring words) in a sentence and OOV words (Rajapaksha et al., 2021). One disadvantage is that the MLM doesn't account for dependencies between predicted tokens (Song et al., 2020). BERT also has limited input length and less training than some of its variants (Adoma et al., 2020; Liu et al., 2019).

3) State-of-the-art: RoBERTa

RoBERTa is a variation of SBERT that was trained on a larger corpus of text (Book-Corpus, CC-News, Open-Web-Text, Stories, Wiki) in order to increase performance (Liu et al., 2020). In comparison to SBERT, RoBERTa has fewer parameters, but better performance (Liu et al., 2019). RoBERTa's advantages include better training (longer training period with a larger set of data), no longer having the next sentence prediction (NSP) task, larger batch sizes, training on longer sequences, and the use of dynamic masking (meaning that the masked tokens in MLM change during training instead of staying static, like with BERT) (Naseem et al., 2021; Glazkova, 2021). One of the main issues with RoBERTa in comparison to BERT is that it is more computationally expensive and takes longer to run (Acheampong et al., 2021). In addition, the corpus does not contain scientific research articles, which is a disadvantage in this study.

4) State-of-the-art: MPNet

The default sentence transformer in the ASReview software is all-mpnet-v2. MPNet stands for masked and permuted language modeling, and is another model based on BERT that was created to improve performance (Song et al., 2020). The key difference between BERT and MPNet is that MPNet no longer employs the original version of the Masked Language Model (MLM) task. The issue with MLM is that it causes the model to ignore the positional dependencies between the masked tokens (meaning that the predicted tokens are assumed to be independent of each other - this does not coincide with human language) (Song et al., 2020). Instead, MPNet combines MLM from the BERT architecture, along with the Permutation Language Model (PLM) from XLNet (another model based off of BERT), to create a new model. The new model improves BERT in two ways: 1) it uses MLM to note the position information of all tokens, 2) and uses PLM to note the dependency between predicted tokens (Song et al., 2020).

5) State-of-the-art: SPECTER

SPECTER is a transformer that uses document-level embeddings instead of word or sentence-level embeddings (Cohan et al., 2020). By using document-level embeddings, the model allows for the comparison of documents and not just pieces of text, which could be preferable when wishing to find relevant scientific articles. In fact, the model is trained with scientific research in mind - it uses SciBERT, a version of BERT trained on scientific text, as the base of the model (Cohan et al., 2020). The model uses citations to compare similarity between documents and

further trains the model on hundreds of thousands of scientific paper titles, abstracts, and citations (Cohan et al., 2020).

A summary of all the models' advantages and disadvantages listed above can be seen in the table below. The only model architecture that fulfills all four parameters examined is the sentence transformer.

	Model	**Syntax**	**Semantics**	**Context**	**Out-of-vocabulary (OOV)**
Classical	**Tf-idf**	X	X	X	X
Classical	**Doc2Vec**	✓	✓	X	X
State-of-the-art	**Transformer Models (RoBERTa, MPNet, SPECTER)**	✓	✓	✓	✓

Table 1: A table summarizing the main differences between the feature extraction models

Transformer Model Selection:
Past literature shows that the RoBERTa model has a consistently high performance on various text classification tasks. In a study comparing the performance of BERT, DistilBERT, RoBERTa, XLNet, and ELECTRa as language models for an emotion recognition task (Reddit comments labeled with emotion categories), RoBERTa outperformed the other pre-trained models (Cortiz, 2021). In a large-scale study on transformer-based word embeddings, transformer feature extraction models were paired with a CNN classifier in order to detect sarcasm and irony in social media datasets (Twitter and Reddit). Amongst all of the transformer-based models, RoBERTa performed the best (Ahuja & Sharma, 2021). Additionally, a study on political science text classification compared the language models ULMFIT, RoBERTa, and XLNet (in combination with the SVM, logistic regression, and random forest classifiers). RoBERTa had a higher performance, especially for the smallest datasets tested, which was attributed to the fact that RoBERTa is trained longer and on larger corpora (Terechshenko et al., 2020). Additionally, the study compared transformer models to classical methods such as bag-of-words and word2vec and concluded that transformer language models performed the best.

Two other sentence transformers will be implemented in this study: SPECTER and MPNet. The SPECTER model seems to be well-aligned to ASReview goals: the fact that it is trained on scientific paper titles and abstracts coincides with the data that ASReview most often reviews. MPNet, the default of ASReview's SBERT feature extractor, will also be included in the comparison. The model is another interesting sentence transformer to examine because it has different advantages than RoBERTa and SPECTER: while RoBERTa carries the advantage of being trained on large corpora, and SPECTER has been trained on scientific corpora, MPNet's advantage is that it models the complexities of human language more accurately.

1.3) Aims and Research Question
Previous research has shown that transformer models, such as variations of SBERT, are a powerful new method of feature extraction. However, although many studies compare the performance of various transformer models, not many studies directly compare classical vs. state-of-the-art feature extraction models, especially in the context of systematic reviews. There is a gap in the research when it comes to comparing both the theoretical background and performance of various feature extractor types/architectures for systematic reviews.

This study focuses on the following research question: *Can new feature extraction methods, i.e., transformers (i.e, RoBERTa, SPECTER, MPNet), outperform more classical feature extraction techniques (i.e., tf-idf, Doc2Vec) in ASReview?*

This question will be examined in the context of ASReview, the systematic review software that uses machine learning to find relevant papers. The purpose of answering this question is two-fold: Firstly, there exist many theoretical reasons why certain text representation models should outperform other models. It is important to examine how these theoretical reasons play out in an empirical setting in order to reach optimal performance in the empirical setting. Secondly, the research question is meant to assist the target audience: the end-user of ASReview software. Since feature extractor selection is an important step the user performs, it is important for the user to understand the performance of the different types of feature extractors. Picking the right feature extractor could increase model performance: the user will find more relevant articles in a shorter amount of time. This study aims to shed light on which feature extraction methods ASReview should implement in the future.

2. Data

ASReview offers multiple benchmark datasets for users to run simulations with. For this study, the PTSD Trajectories dataset by Van de Schoot et al. was chosen. This dataset deals with PTSD trajectories and uses articles about longitudinal studies examining posttraumatic stress after trauma (van de Schoot et al., 2017; van de Schoot et al., 2018). Finding relevant articles for a systematic review is an imbalanced problem; in most cases, there will be a lot more irrelevant than relevant articles (O'Mara-Eves, 2015). The benchmark dataset used in this study is no exception. The dataset contains 6,189 studies that were extracted from Pubmed, Embase, PsychInfo, and Scopus, and as this is a labeled dataset, it is known that 43 of those studies are considered "relevant" (included in the systematic review).

3. Methods

3.1) Methodology Steps
To examine the question of whether or not there is a difference in classical and state-of-the-art feature extractor performance, the process was broken down into three main steps: 1) Model Selection, 2) Implementation and Simulation, and 3) Performance Metrics.

First of all, the literature overview was used to select the models with the best expected performance. The model selection step began afterwards, and in this step the focus was on

examining the models found on Hugging Face. The implementation step involved running the selected Hugging Face models (since many variations of models exist) from the sentence-transformers library using the ASReview Python API and simulation settings. The simulation mode in ASReview contains a SBERT feature extractor that can be adjusted for different models. This study made use of these settings to implement the various Hugging Face models. Performance metrics, calculated using the ASReview "statistics" package, were used to examine how well the combination of the sentence transformer with the different classifiers managed to find all the relevant articles. Further explanation of all of the steps will be given in the sections below.

3.2) Model Selection

The model selection step was based on the results of the literature review, where RoBERTa and SPECTER proved to be promising transformer language models (along with MPNet, the ASReview default). The model selection process carried out by investigating Hugging Face models.

Hugging Face:

All sentence transformer models in this study were implemented using the sentence-transformers library on Hugging Face. Hugging face (https://huggingface.co/) is a website containing open-source machine learning models and allows users to upload and download models for free, including a wide variety of sentence transformer models (Hugging Face, n.d.). Using the knowledge from background research, which indicated RoBERTa was the model of interest, three different transformer models were selected.

Model version and the number of downloads were both examined when selecting which model to implement from the sentence-transformers page (in other words, the model version with higher downloads was preferred since high popularity generally indicates a more recent version of the model). Model versions differ due to the fact that as part of the open-source community, any user can download, fine-tune, and then upload the updated version of a model. Following past research, a "distil" (all-distilroberta-v1) and "base" (stsb-roberta-base-v2) version of the RoBERTa model were selected from the sentence-transformers library. The "base" version is the larger base version of the model, while the "distilled" version of the model is a lighter and faster version with half the number of layers and a little over half of the parameters of the "base" model. The "distilled" version runs twice as fast (Hugging Face, n.d.).

Implemented Feature Extractors and Classifiers:

The **classifiers** used in this study (along with their implementations in ASReview) are as follows: 1) SVM, 2) Logistic Regression, 3) Random Forest using the sklearn library, 4) Naive Bayes using the sklearn Multinomial Naive Bayes classifier, and 5) NN2 classifier (a fully connected neural network with 2 hidden layers, dense and of the same size) (ASReview., 2022). Each of the classifiers was run in combination with each of the six feature extractors (with the exception of Naive Bayes, which was only run with tf-idf).

The **feature extractor** implementations are: 1) The Doc2Vec implementation using the genism library and 2) Tf-idf implemented using the sklearn library (ASReview., 2022). The sentence transformers are: 3) Distil-RoBERTa, 4) RoBERTA-base, 5) Allenai-SPECTER, and 6) All-

mpnet-base-v2. The transformers were extracted from the Hugging Face sentence-transformers library (Hugging Face, n.d.) and implemented using the ASReview code for SBERT (ASReview., 2022). All-mpnet-base-v2 is the current default sentence transformer used by ASReview. Further details about the architecture of the chosen transformer models can be found in Table 2.

Sentence Transformer	Architecture	Corpus model was trained on	Documentation
all-distilroberta-v1	6-layer, 768-hidden, 12-heads, 82M parameters (distilled from RoBERTa-base)	Book-Corpus, CC-News, Open-Web-Text, Stories, Wiki	https://huggingface.co/sentence-transformers/all-distilroberta-v1
RoBERTa-base	12-layer, 768-hidden, 12-heads 125M parameters	Book-Corpus, CC-News, Open-Web-Text, Stories, Wiki	https://huggingface.co/sentence-transformers/stsb-roberta-base-v2
Allenai-specter	12-layer, 768-hidden, 12-heads 110M parameters	SciBERT corpus	https://huggingface.co/sentence-transformers/allenai-specter
All-mpnet-base-v2	12-layer, 768-hidden, 12-heads 110M parameters	Book-Corpus, CC-News, Open-Web-Text, Stories, Wiki	https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Table 2: Sentence transformer models selected from Hugging Face

3.3) Implementation and Simulation

To examine the performance of different feature extractor and classifier combinations for finding relevant articles, the simulation mode of the ASReview software was used (version v0.19.3) (van de Schoot et al., 2021). The simulation mode allows for customization of feature extractors and classifiers. A total of 25 combinations and simulations were run; all of the implemented feature extractors and classifiers can be viewed in Figure 3 (further details are also given in the next section). The simulations follow the ASReview pipeline mentioned in the section above, with the feature extraction run on the Schoot et al. dataset. Figure 3 shows a basic overview of the simulation and the steps of the analysis.

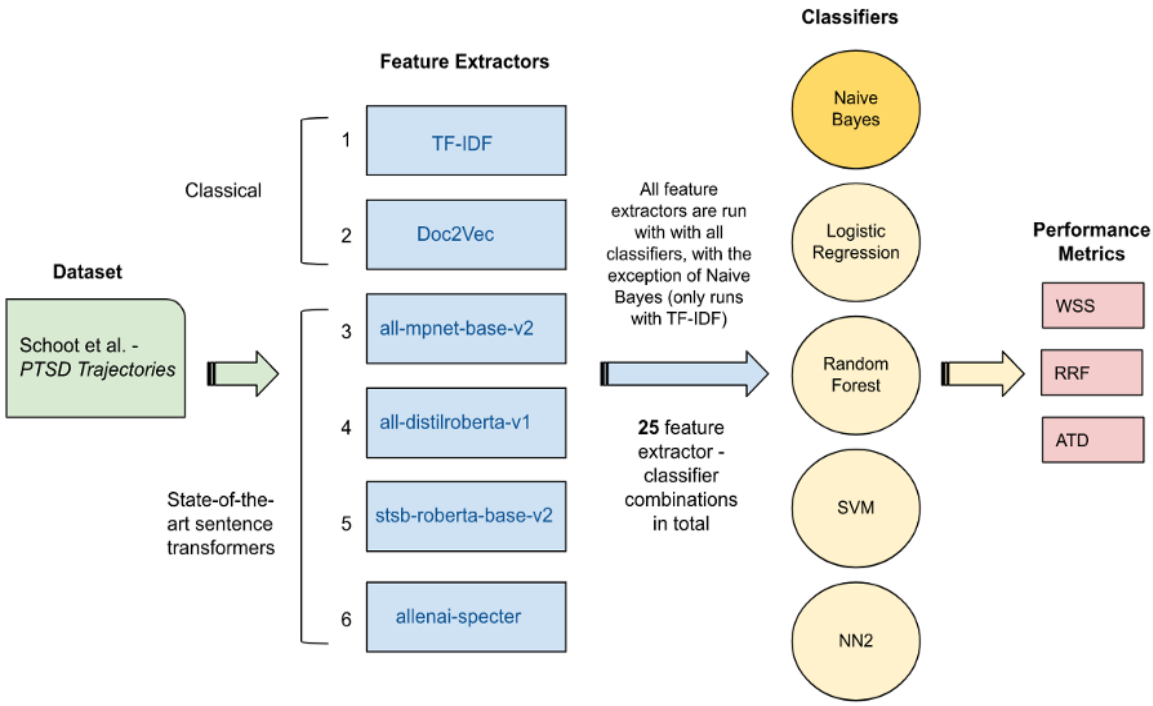

Figure 3. Simulation Steps Overview - Shows the feature extractors and classifiers used during the simulation runs. A total of 25 simulations were run.

The simulations were run on Google Colab Pro, using a high-RAM GPU to speed up the run-time of the simulations (Colab Pro can reach ~25 GB of RAM) using the ASReview Python API and command line interface (ASReview., 2022). Only the feature extractor and/or classifier were changed during the simulations in order to keep performance results as unbiased as possible. The default settings used were as follows:

Setting Name	**Chosen Value**	**Definition**
Query strategy	Max query (default)	Model chooses record with highest relevance score (article most likely to be relevant)
Balance strategy	Double balance (default)	Rebalancing strategy that super-samples 1's based on the number of 0's and the total number of studies in the dataset
Number of instances	10	The number of studies queried during each query
Initial seed	10 (can be any fixed number)	Setting the seed to a fixed number controls for the randomness in each simulation run so that performance can be compared in an unbiased manner
Prior knowledge	1 relevant, 1 irrelevant	Starting training set

Table 3: Simulation settings for the ASReview simulation mode (v0.19.3)

This study chose to use default settings as much as possible in order to emulate the settings that most users will use. The query strategy determines how to choose the next record the researcher needs to label. The balance strategy helps deal with the imbalance problem systematic review datasets commonly have (van den Brand & van de Schoot, 2021). The initial seed is necessary in order to minimize bias due to randomness. The prior knowledge refers to the articles labeled by the researcher (through random or keyword selection). The size of the prior knowledge determines the starting training set that will be used to train the first model in the active learning cycle. In this study, the prior knowledge was set as one relevant and one irrelevant article in order to also see how well the models perform in the beginning of the training, when the training set is small (i.e., when the researcher does not label many articles). It should be noted that changing these simulation settings could impact the performance of the models, which is why they are kept constant for all the simulations in this study.

3.4) Performance metrics

To assess the performance of the different models, the recall curves were plotted. The recall plots show two metrics - Work Saved Over Sampling (WSS) at 95% recall and Relevant References Found (RRF) at 10% recall. WSS@95 means that at 95% recall, the given percentage is the amount of work that can be saved (in this case, how many less studies need to be screened). A higher WSS@95 is preferable since it indicates how much work the researcher can save by using machine learning instead of manual screening (to find 95% of all relevant articles) (O'Mara-Eves, 2015; van den Brand & van de Schoot, 2021).

RRF@10 is the number of relevant articles found after screening 10% of the dataset. A higher RRF@10 score is preferable because it means that more relevant records have been found after screening only 10% of the articles. If the RRF does not change from RFF@5 to RRF@10 it could indicate that some of the relevant articles are hard to find and are taking longer to be found (van den Brand & van de Schoot, 2021).

An additional metric, Average Time to Discovery (ATD), was also utilized. ATD refers to the average time it takes to find a relevant article, expressed as a percentage/proportion of all articles in the dataset being screened (van den Brand & van de Schoot, 2021). This metric is useful for examining how much of the dataset needs to be screened in order to find a relevant article, and thus a lower ATD means the model is more efficient at discovering a relevant article.

Unlike traditional recall plots where the axes are precision and recall, in the recall plots used in this study, the x-axis shows the percentage/proportion of studies that have been screened. The y-axis shows the percentage/proportion of relevant studies that have been found. The grey diagonal line indicates random screening, which is the rate of finding relevant articles when performing manual (random) screening. Thus, horizontal lines in the plot illustrate the RRF@10 at a specific point in the screening process, while the vertical lines indicate the WSS@95 (the number of studies that needs to be screened so that a certain level of recall is reached) (van den Brand & van de Schoot, 2021).

4. Results

The results indicate that the best performing combination of feature extractor and classifier is tf-idf and Naive Bayes. This coincides with the past ASReview simulation results, which also indicated that the tf-idf and Naive Bayes combination has the best performance (van de Scoot et al., 2021). Recall plots were examined to determine which models performed the best, along with WSS@95, RRF@10, and ATD.

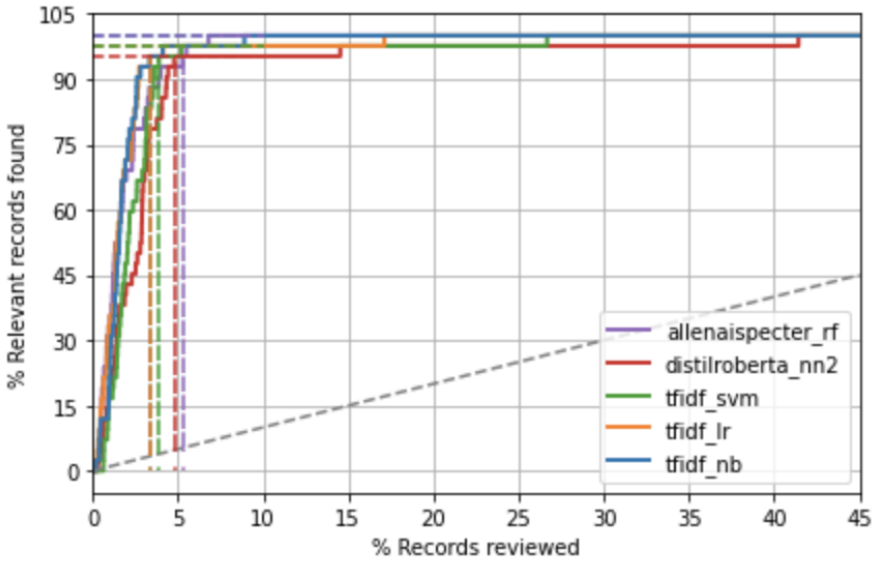

Figure 4: Recall plot of top five best performing models (based on WSS@95). Models are listed in order - SPECTER with RF is the worst of the top five models; Tf-idf with Naive Bayes has the best performance amongst all of the models

Model	WSS@95	RRF@10	ATD
Tf-idf and NB	91.89%	100.00%	0.0177
Tf-idf and LR	91.86%	97.62%	0.0189
Tf-idf and SVM	91.33%	97.62%	0.0276
Distill-RoBERTa and NN2	90.44%	95.24%	0.0356
SPECTER and RF	89.95%	100.00%	0.0185

Table 4 - Performance metrics of top five models

WSS@95 was used as the primary metric to assess model performance, but an optimally performing model is one that has high WSS@95, high RRF@10, and low ATD. The WSS@95 for tf-idf and Naive Bayes is 91.89%, which means that the model saves the researcher 91.89% of work when used instead of manual screening. However, it is also interesting to note that the Distil-RoBERTa and NN2 model is the top fourth model (with a WSS@95 value of 90.44%),

while the SPECTER and random forest model is the top fifth model (with a WSS@95% value of 89.95%). The sentence transformer models may not be performing quite as well as tf-idf, but they do show some promise. Their WSS@95 scores are not drastically lower than those of tf-idf and Doc2Vec (except for RoBERTa-base). Distil-RoBERTa and SPECTER are the most promising transformer feature extractors, outperforming the ASReview default MPNET.

Figure 4 shows the top five feature extractor and classifier combinations based on WSS@95. Note, the x-axis is cut off at 45% reviewed because all models have reached 95% recall by that point. Tf-idf takes the top three spots when in combination with the logistic regression, SVM, and Naive Bayes models, making it the best performing feature extractor overall. This means that this model saves the most time for the researcher (WSS), finds the most relevant articles while only needing to screen a small amount of the dataset (RRF), and needs a shorter amount of time to find a relevant article (ATD).

Figure 5: RRF plot of top five models (for RRF@1, RRF@2, RRF@5, and RRF@10). Models are listed in order - SPECTER with RF is the worst of the top five models; Tf-idf with Naive Bayes has the best performance amongst all of the models

As expected, the tf-idf and Naive Bayes model has the highest RRF@10 at 100.00%. This means that the model found all relevant articles after only screening 10% of the dataset. The RRF@10 scores of the rest of the top five models are all close to 100%. One interesting thing to note is that the Distil-RoBERTa and NN2 model has a slightly lower RRF@10 score (95.24%) than the SPECTER and RF model (100.00%), but still has a slightly higher WSS@95 score. Tf-idf with Naive Bayes has the lowest ATD at 0.0177. The ATD of the models generally increases as the WSS@95 score increases, which is to be expected (see appendix for full results). The higher the WSS@95 score, the more work the reviewer saves, and thus it is expected that it would take less time for a better performing model to find a relevant article (i.e., a smaller ATD).

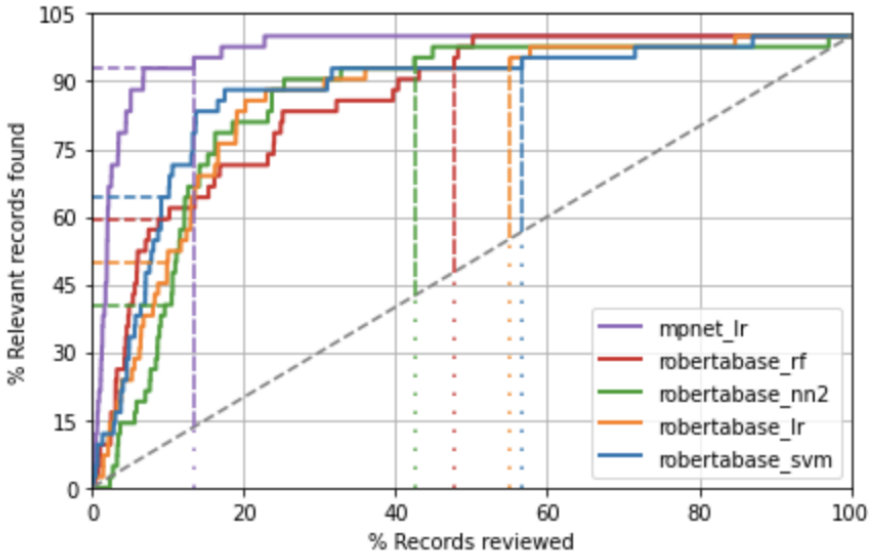

Figure 6: Recall plot of bottom five (the worst performance) models (based on WSS@95). Models are listed in order - MPNet with LR is the best of the bottom five models; Roberta-base with SVM has the worst performance of all models

Model	WSS@95	RRF@10	ATD
RoBERTa-base and SVM	38.54%	64.29%	0.131
RoBERTa-base and LR	40.09%	50.00%	0.148
RoBERTa-base and NN2	52.75%	40.48%	0.153
RoBERTa-base and RF	47.49%	59.52%	0.142
MPNet and LR	81.14%	92.86%	0.0347

Table 5 - Performance metrics of bottom five models

An examination of the bottom five models based on WSS@95 (Figure 6) shows that RoBERTa-base is consistently the worst performing feature extractor. The RoBERTa-base feature extractor has a drastically lower WSS@95 in comparison to all other feature extractors (see Table 5), and the worst performing model is RoBERTa-base and SVM with a WSS@95 value of 38.54%. The next worst feature extractor is the MPNet because when it is in combination with the logistic regression classifier it achieves a WSS@95 of 81.14% (however, this value is still much higher than the WSS@95 values of the RoBERTa-base models). To see the results of all 25 model combinations please refer to the appendix.

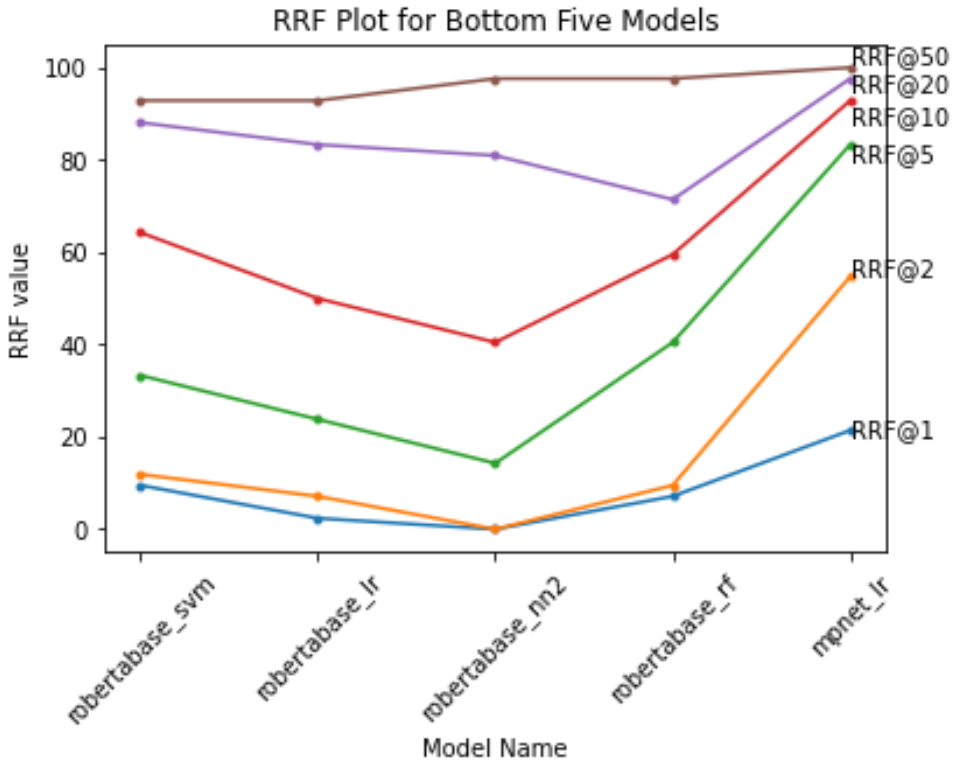

Figure 7: RRF plot of bottom five models (for RRF@1, RRF@2, RRF@5, RRF@10, RRF@20, and RRF@50). Models are listed in order - MPNet with LR is the best of the bottom five models; Roberta-base with SVM has the worst performance of all models

The RRF@10 scores of the bottom five models show more variation than the top five models. For all RRF values (RRF@1, RRF@5, etc.) the MPNET model performs better than RoBERTa-base, and RoBERTa-base combined with the NN2 classifier tends to have the lowest RRF values (see Table 10 in the appendix). A direct comparison of the RFF values shows that the worst RoBERTa-base model based on WSS@95 (RoBERTa-base and SVM) has a RRF@10 of 64.29%, while the RoBERTa-base and NN2 has a RRF@10 of 40.48%. This indicates that when the RoBERTa-base feature extractor is paired with the NN2 classifier it needs to screen a larger proportion of the dataset before it can perform as well as other models (it has worse performance in the beginning of the screening process). ATD scores confirm this because RoBERTa-base with NN2 has the highest ATD at 0.153, which means that it takes longer to find relevant articles.

In fact, all of the RoBERTa-base models have much lower RRF@10 scores in comparison to all other feature extractors (a difference of about ~30-40% in most cases), suggesting that these models struggle to find a relevant article as efficiently as the other feature extractors. This suggests that the RoBERTa-base feature extractor is not suitable for this task or may require some fine-tuning in order for its performance to increase to a comparable level.

When individually examining the results of each classifier (table of all results is in the appendix), the following feature extractor is the best for each classifier (based on WSS@95 and RRF@10):

Classifier	Best performing feature extractor (based on WSS@95)	Best performing feature extractor (based on RRF@10)
Logistic Regression	Tf-idf	Tf-idf
Random Forest	SPECTER	SPECTER
SVM	Tf-idf	Tf-idf
NN2	Distil-RoBERTa	Tf-idf
Naive Bayes	Tf-idf → only feature extractor that can be used with NB	Tf-idf → only feature extractor that can be used with NB

Table 6 - Best performing feature extractor (based on WSS@95 and RRF@10) for each classifier

It can be seen that tf-idf is dominating as the best performing feature extractor across most classifiers. Even when the sentence-transformer outperforms tf-idf on a certain classifier, the margin is a small amount. For example, in the case of SPECTER and random forest the WSS@95 is 89.95%, which is only slightly above the tf-idf and random forest WSS@95 of 89.05%.

This shows that the answer to the research question is not clear-cut or simple. In some cases (combinations with the classifier) the classical feature extractors perform better, and in other cases the sentence transformers manage to perform a little better. Overall, the results indicate that classical feature extraction methods such as tf-idf are powerful in their own right and should not be underestimated, but that transformer language models also have the potential to perform feature extraction well.

5. Discussion

5.1) Main Findings

Past research indicated that RoBERTa models should have a high performance. In fact, the study by Terechshenko et al. suggested that on the whole, transformer models should perform well as feature extractors. However, the performance results show some discrepancies with the theoretical research: In general, the sentence transformers are not performing better than the classical feature extractors. Instead, they are performing about the same or even at a lower performance level. This could be due to several reasons: 1) the more complex transformer models may be overfitting on the data, and/or 2) the datasets the transformers were pre-trained on may not be ideal for systematic reviews (although the SPECTER model slightly contradicts this point).

A closer look at the architecture and theoretical background of the different feature extractors suggests another possible reason. Recall that tf-idf focuses on the frequency of a word within a text. When searching for relevant scientific articles, there may be certain keywords that are very frequent and are present within all of the documents. Moreover, it is important to remember that in the context of ASReview, the search for relevant articles is often being performed based on

keyword(s) given by the researcher. This means that the chosen keyword(s) have a high importance, and so placing a high importance/weight on distinctive keywords may be the reason that tf-idf performs so well. In addition, tf-idf is a word-based feature extractor, rather than a sentence-based feature extractor. Tf-idf may be benefitting from focusing on words instead of sentences.

However, when examining only the transformer feature extractors, one finding is very clear: The ASReview default transformer model is not the transformer model with the highest performance. Distil-RoBERTa and SPECTER tend to outperform MPNET. Therefore, it could be beneficial to consider using either the Distil-RoBERTa or SPECTER model as the default transformer model. The Distil-RoBERTa model would be preferable if generalizability (working with all types of text, including non-scientific text) needs to be considered.

5.2) Study Limitations

Study limitations include the fact that not all possible sentence transformer (and classical feature extractor) models were tested, since it is not feasible. In addition, the statistical power of the results could be improved by running more simulation of all the models. Furthermore, the models were only run on one dataset, but performance could change if the models were run on a dataset of a different topic. Pre-trained models such as sentence transformers could have greatly varying performance if the data type is one the model has never been exposed to.

Another study limitation to consider is the fact that the transformer models were not fine-tuned, which could potentially significantly change performance. The lack of hyperparameter tuning and training on additional data could potentially be used to explain RoBERTa-base's low performance, but unfortunately no conclusions can be drawn within the scope of this study. The significantly lower performance of the RoBERTa-base feature extractor could also indicate that transformers are too complex of a tool for a fairly simple task. Of all the sentence transformer models, the RoBERTa-base model is the most complex at 12 layers and 125 million parameters (compared to Distil-RoBERTa's 6 layers and 82 million parameters) (Hugging Face, n.d.). For a task as straightforward as finding article relevance with a non-noisy dataset, such a complex feature extractor may be unnecessary.

5.3) Future Research

Future research on feature extractors could focus on testing the performance of transformer models on noisy datasets. With a noisy dataset, the performance of transformers may increase, and they may even outperform the simpler models like tf-idf and Doc2Vec. The more complex architecture of transformers lends itself to more challenging tasks, since these pre-trained models are able to pick up on subtleties that the more classical feature extractors may not analyze (such as synonyms, differences due to context, etc.). As part of this research, one could implement a wider range of models - not only transformer models but also additional classical feature extractors to broaden the comparison.

Another suggestion is for future research to compare changes in performance due to increased parameters versus fine-tuning the transformer models. RoBERTa-base has the most parameters, but also the worst performance in this study. This suggests that a large number of parameters may not translate into better performance. Instead, the fact that SPECTER and Distil-RoBERTa

can perform almost as well as tf-idf when paired with certain classifiers, suggests that the pre-training of the transformer model may be more important. Recall, SPECTER's advantage is that it is trained on scientific text (so the model has domain knowledge) and Distil-RoBERTa is trained on the largest corpus (so the model has a wide range of knowledge). Additionally, MPNET is not performing particularly, which is not too surprising under this logic, considering the fact that its advantage is that it uses a different structure for its pre-training task (but the corpus it is pre-trained on is not particularly large or relevant to scientific research).

A more focused approach would be to choose to focus only on fine-tuning the transformer models. The SPECTER model has already been pre-trained on scientific articles but did not perform significantly better from the Distil-RoBERTa model. Further exploration of various tuning methods, along with training on new datasets (such as scientific articles in different domains and news articles), could demystify the effects of fine-tuning and potentially increase the performance of transformer models.

6. Conclusion

The selection of a feature selector can have a great impact on classifier performance. The answer to the question of whether or not the new transformer feature extractors can outperform classical models in ASReview is: not quite yet, but perhaps with more tuning. This study shows that simply because a feature extractor model is new and state-of-the-art, does not mean that it is the best model for every situation. Although the performance of three out of four (Distil-RoBERTa, MPNet, and SPECTER) of the sentence transformers was promising, the classical tf-idf feature extractor still performed the best overall. The results of this study illustrate the importance of not underestimating older, classical models. However, this does not mean that transformer models should be completely discounted. Rather, the fairly good performance of some of the transformer models shows the potential of focusing on fine-tuning and improving these models in future research.

7. Appendix

Although performance relies on the combination of the feature extractor with the classifier, in order to better examine the feature extractor performance on its own, the average WSS@95 was computed across all classifiers for each feature extractor. There is variation when the feature extractor is combined with a different classifier, which is why standard deviation was also noted. These results further reinforce the fact that the tf-idf performs best on average across all classifiers, while RoBERTa-base performs the worst:

Feature Extractor	Mean WSS@95 across all classifiers	Standard Deviation (SD)
Distil-RoBERTa	85.66%	4.0061
RoBERTa-base	44.71 %	5.7398
Allenai-specter	87.34%	2.4656
All-mpnet-base-v2	82.72%	1.3278
Tf-idf	90.90%	1.0771
Doc2Vec	85.93%	1.1725

Table 7 - Average WSS@95 and SD across all classifiers for each feature extractor

Schoot et al. Dataset Complete Simulation Results:

Feature Extractor	Classifier	WSS@95	WSS@100	RRF@5	RRF@10	Run time	ATD
Distil-RoBERTa	SVM	80.55%	40.02%	80.95%	92.86%	15m 35s	0.0492
	Random Forest	83.07%	82.08%	90.48%	92.86%	15m 1s	0.0302
	Logistic Regression	88.58%	69.63%	88.10%	97.62%	4m 52s	0.0347
	NN2	90.44%	58.56%	95.24%	95.24%	1h 51m 41s	0.0356
RoBERTa-base	SVM	38.54%	12.91%	33.33%	64.29%	14m 58s	0.131

	Random Forest	47.49%	49.78%	40.48%	59.52%	12m 47s	0.142
	Logistic Regression	40.09%	15.23%	23.81%	50.00%	8m 50s	0.148
	NN2	52.75%	2.89%	14.29%	40.48%	1h 50m 52s	0.153
Allenai-specter	SVM	87.33%	66.17%	88.10%	97.62%	11m 11s	0.0347
	Random Forest	89.95%	93.18%	92.86%	100.00%	13m 12s	0.0185
	Logistic Regression	88.69%	77.74%	85.71%	97.62%	9m 49s	0.0294
	NN2	83.38%	86.77%	60.47%	93.02%	1h 45m 31s	0.0561
All-mpnet-base-v2	SVM	84.28%	73.10%	83.33%	90.48%	18m 25s	0.0366
	Random Forest	81.69%	79.55%	88.10%	92.86%	12m 36s	0.301
	Logistic Regression	81.14%	75.90%	85.71%	92.86%	4 min 45s	0.0311
	NN2	83.75%	71.20%	83.33%	92.86%	1h 52m 4s	0.0359
Tf-idf	SVM	91.33%	73.32%	95.24%	97.62%	35m 43s	0.0276
	Random Forest	89.05%	71.44%	90.48%	95.24%	12m 24s	0.0333
	Logistic Regression	91.86%	82.87%	97.62%	97.62%	37.3s	0.0189
	Naive Bayes	**91.89%**	91.08%	97.62%	100.00%	13.5s	0.0177
	NN2	90.36%	79.42%	95.24%	97.62%	3h 6m 41s	0.0262
Doc2Vec	SVM	85.91%	84.13%	59.52%	95.24%	1h 4m 28s	0.0496

	Random Forest	85.25%	87.55%	52.38%	95.24%	13min 57s	0.0524
	Logistic Regression	87.82%	88.46%	64.29%	95.24%	4m 2s	0.0454
	NN2	84.72%	78.58%	76.19%	92.86%	1h 49m 1s	0.0389

Table 8 - Complete Simulation Results: WSS@95, WSS@100, RRF@5, RRF 10, runtime, and ATD for all feature extractor and classifier combinations; 25 simulations run in total

Model	**RRF@1**	**RRF@2**	**RRF@5**	**RRF@10**
Tf-idf and NB	21.428571	71.428571	97.619048	100.000000
Tf-idf and LR	33.333333	71.428571	97.619048	97.619048
Tf-idf and SVM	16.666667	47.619048	95.238095	97.619048
Distil-RoBERTa and NN2	19.047619	42.857143	95.238095	95.238095
Allenai-specter and RF	33.333333	69.047619	92.857143	100.000000

Table 9 – RRF values of top five models

Model	**RRF@1**	**RRF@2**	**RRF@5**	**RRF@10**	**RRF@20**	**RRF@50**
RoBERTa-base and SVM	9.523810	11.904762	33.333333	64.285714	88.095238	92.857143
RoBERTa-base and LR	2.380952	7.142857	23.809524	50.000000	83.333333	92.857143
RoBERTa-base and NN2	0.000000	0.000000	14.285714	40.476190	80.952381	97.619048
RoBERTa-base and RF	7.142857	9.523810	40.476190	59.523810	71.428571	97.619048
MPNET and LR	21.428571	54.761905	83.333333	92.857143	97.619048	100.000000

Table 10 – RRF values of bottom five models

Code and additional visualizations can be found on: https://github.com/acaklovic/Comparison-of-feature-extractors-using-ASReview/tree/main

References

Acheampong, F. A., Nunoo-Mensah, H., & Chen, W. (2021). Transformer models for text-based Emotion Detection: A review of Bert-based approaches. *Artificial Intelligence Review*, *54*(8), 5789–5829. https://doi.org/10.1007/s10462-021-09958-2

Adoma, A. F., Henry, N. M., & Chen, W. (2020, December). Comparative analyses of bert, roberta, distilbert, and xlnet for text-based emotion recognition. In *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)* (pp. 117-121). IEEE.

Ahuja, R., & Sharma, S. C. (2021). Transformer-Based Word Embedding With CNN Model to Detect Sarcasm and Irony. *Arabian Journal for Science and Engineering*, 1-14.

ASReview. (2022). *API reference*. ASReview LAB: Active learning for Systematic Reviews. Retrieved June 21, 2022, from https://asreview.readthedocs.io/en/latest/reference.html

Briggs, J. (n.d.). *Natural language processing (NLP) for Semantic Search*. Pinecone. Retrieved May 2022, from https://www.pinecone.io/learn/nlp/

Cohan, A., Feldman, S., Beltagy, I., Downey, D., & Weld, D. S. (2020). Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Cortiz, D. (2021). Exploring transformers in emotion recognition: a comparison of bert, distillbert, roberta, xlnet and electra. *arXiv preprint arXiv:2104.02041*.

Devika, R., Vairavasundaram, S., Mahenthar, C. S., Varadarajan, V., & Kotecha, K. (2021). A deep learning model based on Bert and sentence transformer for semantic keyphrase extraction on Big Social Data. *IEEE Access*, *9*, 165252–165261. https://doi.org/10.1109/access.2021.3133651

Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Glazkova, A. (2021, May). Identifying topics of scientific articles with BERT-based approaches and topic modeling. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining* (pp. 98-105). Springer, Cham.

Hugging Face. (n.d.). *Sentence-transformers/all-mpnet-base-V2 · hugging face*. sentence-transformers/all-mpnet-base-v2 · Hugging Face. Retrieved June 5, 2022, from https://huggingface.co/sentence-transformers/all-mpnet-base-v2

Hugging Face. (n.d.). *Sentence-transformers/all-distilroberta-v1 · hugging face*. sentence-transformers/all-distilroberta-v1 · Hugging Face. Retrieved June 5, 2022, from https://huggingface.co/sentence-transformers/all-distilroberta-v1

Hugging Face. (n.d.). *Sentence-transformers/allenai-specter · hugging face*. sentence-transformers/allenai-specter · Hugging Face. Retrieved June 5, 2022, from https://huggingface.co/sentence-transformers/allenai-specter

Hugging Face. (n.d.). *Sentence-transformers/stsb-roberta-base-v2 · hugging face*. sentence-transformers/stsb-roberta-base-v2 · Hugging Face. Retrieved June 5, 2022, from https://huggingface.co/sentence-transformers/stsb-roberta-base-v2

Kalyan, K. S., Rajasekharan, A., & Sangeetha, S. (2022). Ammu: A survey of transformer-based biomedical pretrained language models. *Journal of Biomedical Informatics*, *126*, 103982. https://doi.org/10.1016/j.jbi.2021.103982

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

Liu, Q., Kusner, M. J., & Blunsom, P. (2020). A survey on contextual embeddings. *arXiv preprint arXiv:2003.07278*

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Naseem, U., Razzak, I., Khan, S. K., & Prasad, M. (2021). A comprehensive survey on word representation models: From classical to state-of-the-art word representation language models. *ACM Transactions on Asian and Low-Resource Language Information Processing*, *20*(5), 1–35. https://doi.org/10.1145/3434237

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

Qiu, X. P., Sun, T. X., Xu, Y. G., Shao, Y. F., Dai, N., & Huang, X. J. (2020). Pre-trained models for Natural Language Processing: A Survey. *Science China Technological Sciences*, *63*(10), 1872–1897. https://doi.org/10.1007/s11431-020-1647-3

Rajapaksha, P., Farahbakhsh, R., & Crespi, N. (2021). BERT, XLNet or RoBERTa: The Best Transfer Learning Model to Detect Clickbaits. *IEEE Access*, *9*, 154704-154716.

Reimers, N., & Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ruiz-Dolz, R., Alemany, J., Barberá, S. M. H., & García-Fornes, A. (2021). Transformer-based models for automatic identification of argument relations: A cross-domain evaluation. *IEEE Intelligent Systems*, *36*(6), 62-70.

Song, K., Tan, X., Qin, T., Lu, J., & Liu, T. Y. (2020). Mpnet: Masked and permuted pre-training for language understanding. *Advances in Neural Information Processing Systems*, *33*, 16857-16867.

Tabinda Kokab, S., Asghar, S., & Naz, S. (2022). Transformer-based deep learning models for the sentiment analysis of social media data. *Array*, *14*, 100157. https://doi.org/10.1016/j.array.2022.100157

Terechshenko, Z., Linder, F., Padmakumar, V., Liu, F., Nagler, J., Tucker, J. A., & Bonneau, R. (2020). A comparison of methods in political science text classification: Transfer learning language models for politics. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.3724644

van den Brand, S.A.G.E., van de Schoot, R. (2021). ASReview Simulation Mode: Class 101. *Blogposts of ASReview*.

van de Schoot, R., de Bruin, J., Schram, R., Zahedi, P., de Boer, J., Weijdema, F., ... others (2021). An open source machine learning framework for efficient and transparent systematic reviews. Nature Machine Intelligence, 3(2), 125–133.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, *30*.