information

Article

Multi-Keyword Classification: A Case Study in Finnish Social Sciences Data Archive

Erjon Skenderi [1,*], Jukka Huhtamäki [1] and Kostas Stefanidis [2]

[1] Faculty of Management and Business, Tampere University, 33100 Tampere, Finland; jukka.huhtamaki@tuni.fi
[2] Faculty of Information Technology and Communication Sciences, Tampere University, 33100 Tampere, Finland; konstantinos.stefanidis@tuni.fi
[*] Correspondence: erjon.skenderi@tuni.fi

Abstract: In this paper, we consider the task of assigning relevant labels to studies in the social science domain. Manual labelling is an expensive process and prone to human error. Various multi-label text classification machine learning approaches have been proposed to resolve this problem. We introduce a dataset obtained from the Finnish Social Science Archive and comprised of 2968 research studies' metadata. The metadata of each study includes attributes, such as the "abstract" and the "set of labels". We used the Bag of Words (BoW), TF-IDF term weighting and pretrained word embeddings obtained from FastText and BERT models to generate the text representations for each study's abstract field. Our selection of multi-label classification methods includes a Naive approach, Multi-label k Nearest Neighbours (ML-kNN), Multi-Label Random Forest (ML-RF), X-BERT and Parabel. The methods were combined with the text representation techniques and their performance was evaluated on our dataset. We measured the classification accuracy of the combinations using Precision, Recall and F1 metrics. In addition, we used the Normalized Discounted Cumulative Gain to measure the label ranking performance of the selected methods combined with the text representation techniques. The results showed that the ML-RF model achieved a higher classification accuracy with the TF-IDF features and, based on the ranking score, the Parabel model outperformed the other methods.

Keywords: multi-label classification; supervised learning; text representation; text feature extraction

check for updates

Citation: Skenderi, E.; Huhtamäki, J.; Stefanidis, K. Multi-Keyword Classification: A Case Study in Finnish Social Sciences Data Archive. *Information* **2021**, *12*, 491. https://doi.org/10.3390/info12120491

Academic Editor: Arkaitz Zubiaga

Received: 21 Ooctober 2021
Accepted: 20 November 2021
Published: 25 November 2021

1. Introduction

The Finnish Social Science Data Archive (FSD) is a Finnish entity that stores and provides access to digital social sciences research data. The research data are comprised of studies that are structured according to the Data Documentation Initiative standard. FSD translates the metadata of every study to English and implements a manual study labelling process through which a variable-length set of relevant labels are assigned to each respective study. Manually labelling the studies in their archive has become more challenging and expensive with the increasing amount of digital research data. In this work, we considered the task of identifying and selecting a label recommending approach to facilitate the study labelling process at FSD.

The task of assigning relevant labels to an entity based on its corresponding features is known as Multi-label Classification (MLC). The MLC methodologies have been applied in various domains. Loza Mencía and Fürnkranz introduced the EUR-Lex database of legal documents of the European Union where each document is assigned a set of labels [1]. EUR-Lex has been used as a benchmark dataset for various MLC techniques [2–4]. Papanikolaou et al. implemented an ensemble of algorithms to perform MLC on a dataset comprised of about 12 million biomedical papers [5]. News articles published online are another domain of application for MLC task. Al-Salemi et al. proposed a benchmark news dataset for the Arabic language and used it to test the performance of various classification techniques [6]. Fiallos et al. applied MLC methods in the domain of social networks [7]. The authors proposed a methodology to train a multi-label model on a dataset obtained from the Reddit online forum and use the trained model to predict the topics of interest on a Twitter dataset.

Our data snapshot consists of 1484 Finnish study metadata and the 1484 corresponding English-translated study metadata. The metadata of every study includes information such as the title, abstract, author(s), set of labels etc. We selected the abstract and the set of labels of each study to test various MLC methods. We used the information contained within the abstract field as an input for the MLC methods because it provides a summary for each respective study. The goal of the MLC methods was to learn any patterns and correlations emerging between the abstract and the set of labels of every study. The information contained within the original form of the abstract field of every study is not computable by the machine learning classification models. We transformed the textual information into a machine-readable format by employing four text feature extraction techniques: Bag of Words (BoW), TF-IDF, FastText-based and Bert-based.

We use the extracted features to train and test a selection of MLC methods which consist of Naive approach, Multi-label k Nearest Neighbours (ML-kNN), Multi-label Random Forest (ML-RF), X-BERT and Parabel. We computed the accuracy and the ranking scores for the technique combination with the feature extraction techniques.

The main contributions of our work are summarized in the following list:

- We introduce the FSD dataset as a benchmark for the MLC task. The properties of this dataset, such as size and the bi-lingual metadata, make it particular in the field of MLC.
- We evaluate the performance of several textual MLC approaches in the domain of social sciences. We used a Naive approach, and the ML-KNN, ML-RF, X-BERT and Parabel techniques to tackle the MLC problem.
- We compare the performance of a set of text representation techniques by combining them with the selected MLC methods. This quantitative evaluation provides an objective comparison of the text representation techniques.

The rest of this work is organised as follows. In Section 2, we discuss the task and our workflow analysis. In Section 3, we provide details on the dataset and discuss the approach we employed to split it into two parts to be used during the training and testing of the selected models. In Section 4, we report the feature extraction techniques we used and, in Section 5, we describe the classification models that we applied and the respective reasoning of selection for each. Finally, in Section 6, we describe the experimental setup and report the results that we obtained.

2. FSD and Analysis of Workflow

The Finnish Social Science Data Archive (FSD) maintains and provides access to an archive of Finnish social science studies. The studies are indexed and their metadata are translated to English. FSD assigns labels to all the studies in their archive to facilitate the process of organising, retrieving and disseminating the data. The labelling process is manually performed separately for the Finnish and English metadata of each study. Our task at FSD focuses on identifying and selecting a technique that would be used to facilitate the labelling process. The selected method should recommend a list of labels to be assigned to each new study that is added to the archive. The list of recommended labels would then be validated and updated manually by the responsible clerk at FSD. This semi-automated process would speed up the labelling process and would help maintaining a consistent label distribution among all the studies thus minimizing the potential human error that would result in labels not being included or included erroneously.

We consider the task of Multi-label text classification in the domain of social sciences by using the FSD data. We summarize the workflow of our task into four stages: Dataset Preparation, Train-Test Split, Feature Extraction and Model Training & Evaluation. In Figure 1, we provide a diagram illustrating the pipeline of our task. Each box illustrates the respective technique that was used during the corresponding stage.

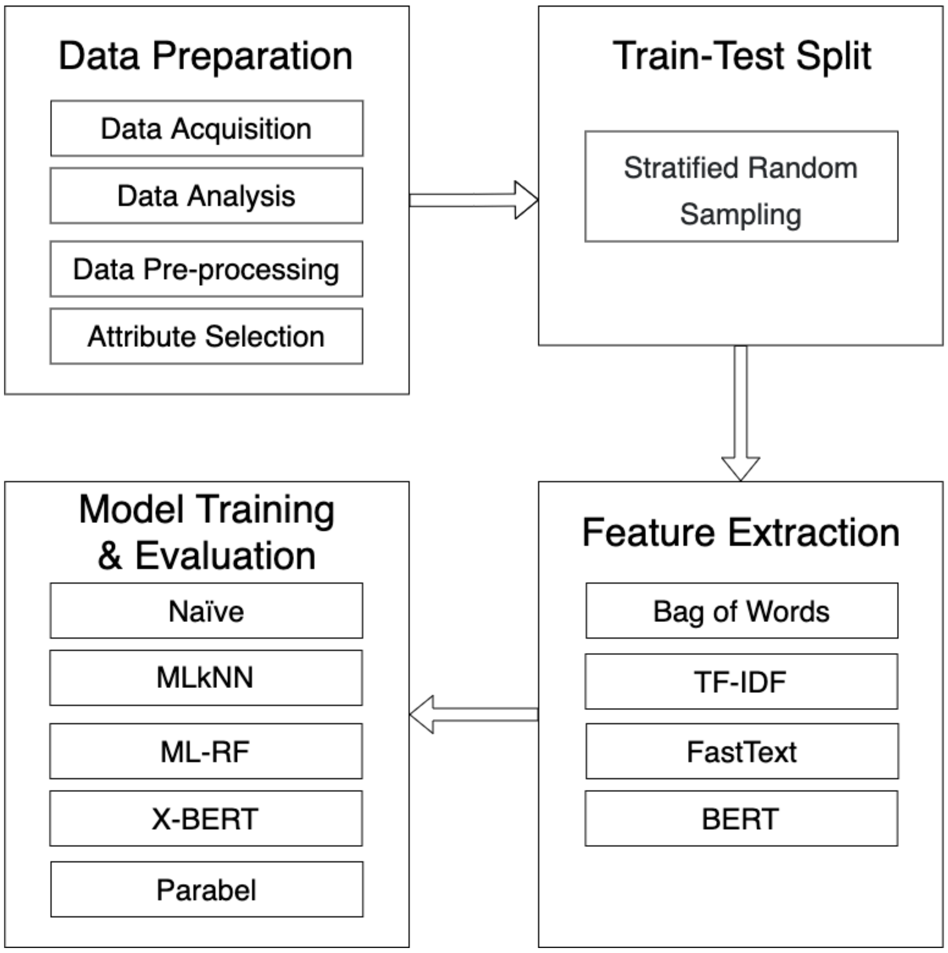


Figure 1. The diagram description of our work. Each box represents a specific phase of the project pipeline.

The first stage of our work focuses on data acquisition and pre-processing. During the first step we acquire the data from FSD. The metadata of the studies they have in their archive is released under Creative Commons Attribution 4.0. The data were obtained as a compressed file and uploaded into a NoSQL database server to ease the process of data analysis and pre-processing. Through the pre-processing step, we transform the original data into a format that could be input into our selected classification models. For our task, this step consisted of transforming the textual information of the metadata into lowercase and lemmatizing the labels. Since the goal of our models is to predict the set of labels assigned to a study, given the corresponding study's abstract, we select the abstract and the labels fields out of the available metadata fields for every study in our dataset. The pre-processed and filtered dataset is then propagated to the next phase of our work.

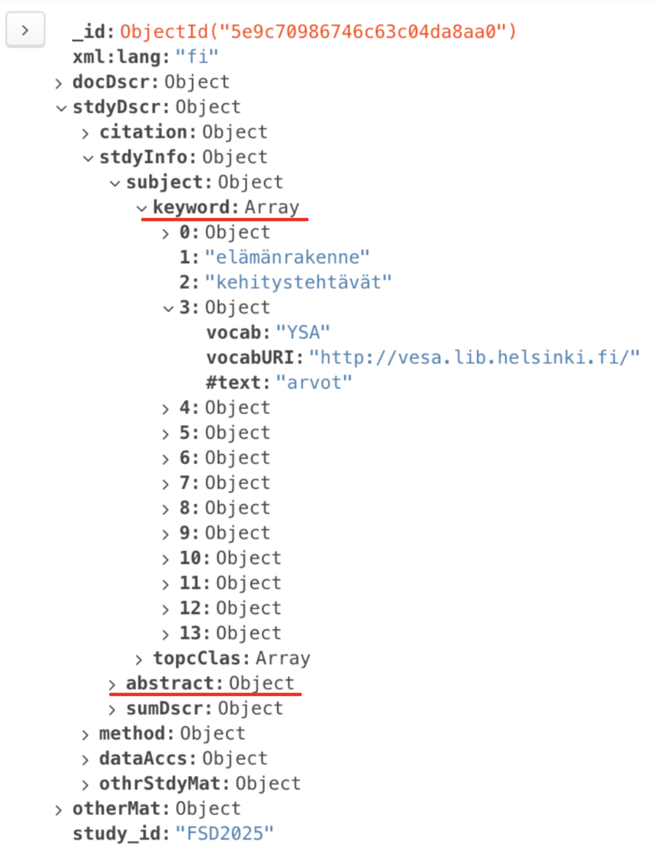
The classification models that we use learn the patterns from a given training dataset on a supervised manner. The supervised machine learning models are prone to the problem of overfitting which arises when the models learn the patterns from the training set too well. Therefore, testing such a model's performance on the same training data could be misleading. By evaluating the performance of the classification models on unseen data, we can assess their level of overfitting. Our approach to address this issue consists of splitting the dataset in two mutually exclusive subsets: Train and Test. Every model was trained with the training data and evaluated with the testing data. Ideally, the distribution of labels in both samples should be equal i.e., the frequency of every keyword would be the same in both subsets. The small number of studies relative to the number of available labels in our dataset pose a challenge to the simple random sampling. We addressed that issue by employing a stratified random sampling method to minimize the under and over representation of specific labels in both subsets.

The textual data contained within the abstract attribute of every study is not computable by our selected classifying models in its original form. To address this issue, the textual information is transformed into a machine-readable format such as a vector of numerical real values. This text transformation process is generally referred to as a feature extraction process. During the third stage of our work, we employed 4 techniques to perform the feature extraction. Each technique transforms the abstract of every study into a vector of numbers that are used to train and test our selected classification models. The differences between the techniques consist of their ability to encapsulate the syntactic and semantic meaning of text.

One of the goals of this work includes the measuring of the performance of the selected classification models on predicting a set of labels for an unseen study's abstract. During the last phase of our work, we trained and evaluated each of the selected classification models. The performance of the models is then tested on the testing data and evaluated on the testing data using the F1 score and the Normalized Discounted Cumulative Gain. We analysed the outcome and reported the results.

3. Dataset

The studies in the FSD archive are structured according to the Data Documentation Initiative standard and represented using the XML (Extensible Markup Language) format. Each study's metadata consists of a subset of defined study-attributes that include the title, the abstract and a variable-length set of manually assigned labels which describe the study's scope. In Figure 2, we provide an illustration of the metadata structure for an example study.



```
   _id: ObjectId("5e9c70986746c63c04da8aa0")
   xml:lang: "fi"
 › docDscr: Object
 ⌄ stdyDscr: Object
   › citation: Object
   ⌄ stdyInfo: Object
     ⌄ subject: Object
       ⌄ keyword: Array
         › 0: Object
           1: "elämänrakenne"
           2: "kehitystehtävät"
         ⌄ 3: Object
             vocab: "YSA"
             vocabURI: "http://vesa.lib.helsinki.fi/"
             #text: "arvot"
         › 4: Object
         › 5: Object
         › 6: Object
         › 7: Object
         › 8: Object
         › 9: Object
         › 10: Object
         › 11: Object
         › 12: Object
         › 13: Object
       › topcClas: Array
     › abstract: Object
     › sumDscr: Object
   › method: Object
   › dataAccs: Object
   › othrStdyMat: Object
 › otherMat: Object
   study_id: "FSD2025"
```

Figure 2. The attributes included in the metadata structure of an example study. The abstract and the keyword attributes are underlined in red.

Our data snapshot is comprised of 2969 studies' metadata in total. We filtered our data to include the title, abstract and the list of labels from the metadata of every study. These attributes were stored on a MongoDB database to ease the process of querying and analysing the data.

In Figure 3, we show a histogram of the distribution of labels among the studies. The number of labels assigned to the studies follow a normal distribution for the Finnish and English metadata. On average, the number of labels appearing in the English-translated metadata is smaller than the number of labels used in the original, Finnish metadata. This change is explained by the linguistic differences between the two languages.

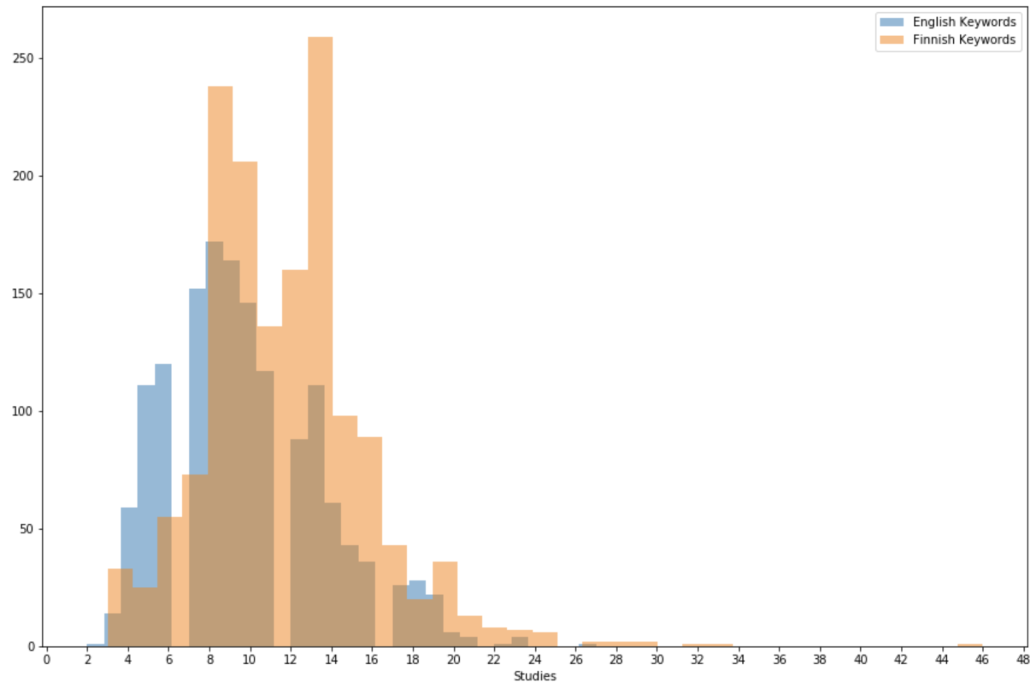


Figure 3. Keyword Frequency Distribution for the English and Finnish studies' metadata.

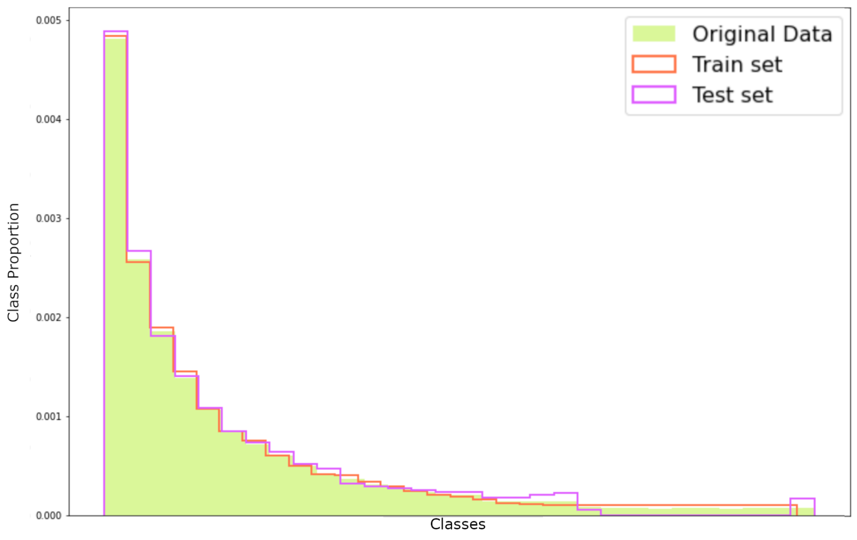
Next, we proceeded with the data pre-processing step. This process consists of transforming the information to make it machine-readable and adapt it for our specific use case. We converted the abstract and the set of labels of every study into lowercase. We observed that some labels were very similar semantically and syntactically but did not match due to the manual labelling of the studies. We applied the lemmatization technique to the labels of every study. Lemmatization consists of reducing a word to its root form by removing the prefixes and suffixes. For example, the words *'financing'* and *'financed'* are both converted into *'finance'* lemma. Lemmas match to an actual word in a dictionary.

In Table 1, we provide further details for the snapshot of the data after the pre-processing step. After the keyword lemmatization, the ratio between the number of distinct labels and the number of studies is 1.95 for the Finnish metadata and 1.15 for the English metadata.

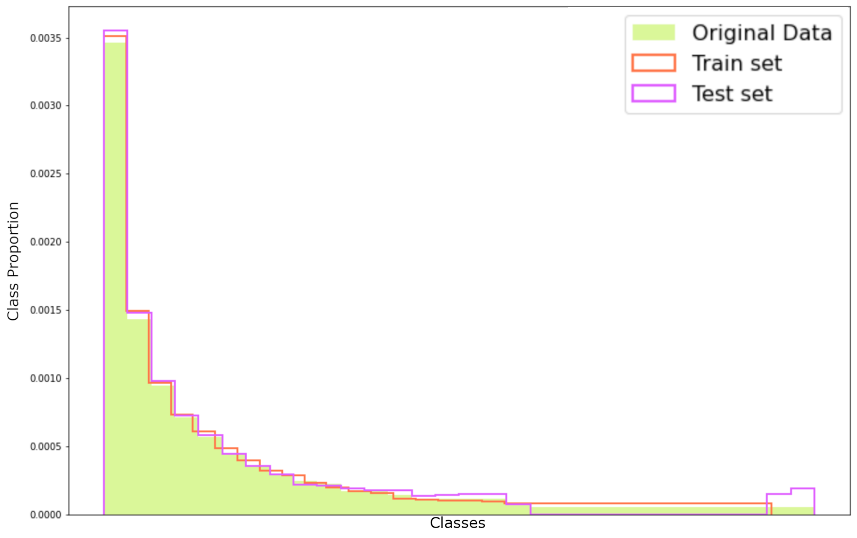
Table 1. Details of our snapshot of data.

	Finnish Metadata	**English Metadata**	**Total**
Number of Studies	1484	1484	2968
Distinct Labels	2896	1709	4605
Labels/Studies	1.95	1.15	1.55

The supervised machine learning models in the task of MLC learn patterns from a given dataset. Such models are evaluated by analysing their performance on an unseen set of data. Following various related work [4,8], we split our dataset into train and test sets for both languages. We employed an iterative stratified sampling method given the small size of our dataset versus the large number of available labels. This approach was proposed by Sechidis et al. [9] and minimizes the over/underrepresentation of labels in both sets of data obtained from the split. The train and test datasets are sampled without replacement from the original dataset while attempting to keep the same distribution of labels. The method consists of sampling the entities that are labelled with the most underrepresented labels in an iterative way. For both the Finnish and English metadata, we sampled 70% of the dataset for the training set and used the remaining 30% for the testing set. In Figure 4, we illustrate the label distribution of the generated samples of data.



(**a**) English metadata



(**b**) Finnish metadata

Figure 4. The sorted label distributions on the train and test data sets for the English and Finnish studies' metadata. The labels are sorted along the horizontal axis, based on their occurrences in the original data. The vertical axis represents the relative class proportion of each respective label.

4. Feature Extraction

The textual information contained within the original form of the abstract attribute of every study is not computable by our selected classification models. We addressed this issue by transforming the textual information into a machine-readable format consisting of vectors of numerical real values. The outcome of this transformation process is referred to as text representation. We used the Bag of Words (BoW), TF-IDF term weighting and Pretrained Word Embeddings techniques to generate the representations for every study's abstract field.

4.1. Bag of Words

The Bag of Words is one of the simplest text representation techniques that we used. This approach does not take into consideration the grammar nor the order of the words in a text. The technique consists of assigning a word to every position in the representation vector, without a specific order. The value of the corresponding position in the vector equals the number of occurrences of the associated word in the represented text. A BoW representation of a text d has the following form:

$$R_{BoW}(d) = (c_{w1}, c_{w2}, \ldots, c_{wn}), \tag{1}$$

where c_{wn} is the number of times the term w appears on a document and n is the size of the dictionary over the corpus. In this work [10], the authors developed a BoW model to represent textual information obtained from a list of publications in the fall-detection area.

4.2. Term Frequency—Inverse Document Frequency

TF-IDF is a standard term weighting approach. Given the set of all documents, the words that appear in fewer documents will be given a higher weight and vice versa. Like the BoW representation, TF-IDF text representation does not take into consideration the grammar nor the order of the words in a text. Given a word w, a set of documents D and a document d such that $d \in D$, the TF-IDF score is calculated as follows:

$$\text{TFIDF}_w = freq_{(w,d)} * log(\frac{|D|}{freq_{(w,D)}}), \tag{2}$$

where $freq_{(w,d)}$ is the number of occurrences of the word w in the document d and $freq_{(w,D)}$ is the number of documents where the word w appears at least once. The TF-IDF representation of a document d would have the following form:

$$R_{\text{TF-IDF}}(d) = (tfidf_{w1}, tfidf_{w2}, \ldots, tfidf_{wn}), \tag{3}$$

where $tfidf_{wn}$ is the TF-IDF score of w and n is the size of the dictionary over the corpus. We limited the TF-IDF representation to only include words that appear at least 3 times over all the studies' abstracts in the training data.

4.3. Pretrained Word Embeddings

The representations obtained from the BoW and TF-IDF techniques do not manifest the semantic meaning of the underlying text. The encapsulation of the syntactic and semantic meaning of a word into a representation vector is known as word embedding. The word embeddings are generated by training a specific language model on large amounts of textual data. We generated the word embeddings using pretrained models since it was not feasible to train such models from scratch given the small size of our dataset. The language models are usually implemented using the neural network architecture. During the training process, the goal of the model is to learn the joint probability function of a sequence of words in a language [11]. Mikolov et al. introduced Word2vec model that employs two learning approaches, CBOW and Skip-Gram [12]. Ali et al. trained a Skip-Gram Word2vec model to represent the semantic meaning of individual user-generated text in social media related to the issues in the transportation industry. Word2vec model can then be used to generate word embeddings, but its main limitation is its inability to generate representations for unseen words.

Fasttext is a model that learns the representations in the word and character level [13]. The FastText model will represent words as the sum of its corresponding n-gram representations. In this work [14], the authors used a FastText embedding model to transform formal and informal words from social networking text into low-dimensional vector representations. Mikolov et al. trained high-quality word vector representations using FastText for different languages including Finnish and English [15]. We use the Finnish and English pretrained models to generate the respective vector representations for every abstract field of our study. The Fasttext vector representation of a document d has have the following form:

$$R_{Fasttext}(d) = (v_1, v_2, \dots, v_m), \tag{4}$$

where m is the size of the vector generated by the model and v is a value in the vector.

Devlin et al. proposed BERT [16], a bidirectional encoder representation model that is based on the transformer architecture [17]. The previous models consume text in a left-to-right fashion, but the BERT model trains on an unlabelled text on both left and right context simultaneously. We use the following BERT pretrained models: *bert-base-cased*, *bert-base-multilingual-cased* and *bert-base-finnish-cased-v1*. The *bert-base-finnish-cased-v1* pretrained model was introduced by Virtanen et al. [18]. The model was trained from scratch on Finnish textual data and performed better than the *bert-base-multilingual-cased* model on several natural language processing tasks. All the pretrained models that we selected consist of 12 layers i.e., transformer blocks. Each layer forwards a single dimension vector representation of size 768 to the next layer. The model we use can provide 12 different representations but there are many approaches to sample a single representation vector [19]. In this work, we generate the vector representation by averaging the representation vectors that we obtain from the 12 individual layers. Similar to the FastText representation, the Bert-based vector representation of a document d is defined as:

$$R_{bert}(d) = (v_1, v_2, \dots, v_{768}), \tag{5}$$

where v is a real numerical value in the vector.

5. Multi-Label Classification Methods

The approaches to the MLC problem can be categorized in two groups: Problem transformation and algorithm adaptation. The methods can be combined with textual feature extraction techniques to handle multi-label text classification.

The transformation of an MLC problem into a single-label classification consists of predicting the probability of each label being assigned to an entity and then combining all predictions into a single output. One of the first methods following this approach is the binary relevance [20]. The method consists of building n binary classifications where n is the total number of labels available. The single-label classification problems are solved independently, and the outputs are merged. Read et al. pointed out that binary relevance method learns the labels independent to each other and provided Classifier Chains method which addressed the issue [21]. The Classifier Chains method randomly sorts the single-label classifiers into a chain such that the output of a classifier is used as an input for the next one in the chain. An alternative approach to the multi-label problem transformation named Pairwise Comparison was introduced by Hüllermeier et al. [22]. This method consists of generating $L(L-1)/2$ pairs of labels where L is the total number of labels, and sampling the original data such that each data sample is labelled with at least one of the respective label pair. Tsuomakas et al. introduced the Label Powerset method [23]. This method consists of generating the powerset of the available labels and run a classifier for every element of the list. The size of the powerset equals 2^L where L is the number of available labels to learn. Through this approach, the total number of classifiers needed would exponentially increase as the number of available labels increases. Given a relatively small size dataset, some combinations in the powerset may not co-appear in any entity making the respective classifier redundant.

Algorithm adaptation approaches consists of modifying single-label classification methods to support multiple label classification. Schapire et al. proposed two modifications to of the original AdaBoost algorithm [24] by introducing AdaBoost.MH and AdaBoost.MR [25]. Both modified algorithms train by generating sets of weak learners on feature pairs and test their performance. AdaBoost.MH assigns weights to every learner's output based on their contribution to minimize the hamming distance between the predicted and actual labels. The AdaBoost.MR will instead optimize the learners based on a ranking metrics. The conventional k'th Nearest Neighbours algorithm has also been adapted for MLC task on previous work such as BRkNN [26] and IBLRML [27]. In this work, we use an adaptation of the kNN algorithm that was introduced by Zhang et al. named ML-kNN [28]. ML-kNN managed to outperform well known approaches at that time, including the AdaBoost and SVM-based alternatives.

We defined two main objectives when selecting the classification models. The first one is to identify easy to implement MLC models that would perform well given the relatively small size of our dataset. The second objective consisted of identifying the recently proposed methods which performed well on related tasks and implement them on our task. The evaluation of the models provided us with quantitative proof to help the decision-making process on selecting which method would meet the requirements for deployment in the FSD system and similar classification tasks.

5.1. Naive Approach

Our implementation of the Naive approach was inspired by the manual labelling process that took place at FSD. The clerk would inspect the target study and identify relevant labels that could be used to describe it best. We defined the Naive approach to recommend assigning specific labels if they appear in the textual content of the abstract of the target study as is. This method is included as a baseline to contrast its performance against the other methods.

5.2. Multi-Label k Nearest Neighbours

The abstract sections of the studies in our dataset are written in a short and concise manner. This limits the performance of the Naive approach that we defined initially. The second model that we employed is the ML-kNN classifier that was introduced by Zhang et al.. This model is based on the kNN (k'th Nearest Neighbours) algorithm [28]. The feature representation vectors that we defined in Section 4 can be used by this method to measure the similarity between the studies' abstracts. On the training phase, the train set studies are grouped into several clusters K using the Euclidean distance metrics. The label vector L of a study x in the training set is represented as $L(x) = (1, 0, \ldots, 1)$. These vectors are used to count the number of neighbours that have each respective label assigned. The prior and likelihood probabilities are then computed for each label. During the prediction phase, the k'th nearest neighbours of the unseen study are identified. The statistical information that was collected during the training phase is used to generate the predictions for the new study by using the maximum a posteriori (MAP) principle. MAP estimates the probabilities for the most probable labels to be assigned to a study based on the observed label distribution from the k nearest neighbours combined with the prior information on the label distribution over all the dataset.

5.3. Multi-Label Random Forest

Another alternative approach that can use the feature representation techniques that we defined in Section 4 is based on the Random Forest (RF) model. The RF model is a general-purpose machine learning algorithm that is comprised of decision trees [29].

The RF algorithm consists of extracting n equally sized sub-samples from the dataset where n corresponds to the number of decision trees that comprise the model. On a random forest, the decision tree models are constrained to select optimum splitting points out of a random subset of features. Using these two constraints, RF imposes an inter-tree diversity, making it a robust machine learning model. The output of a random forest model is obtained through averaging the individual outputs of all the n decision trees, a process known as voting. Figure 5 provides a diagram of the prediction process that takes place on the Random Forest algorithm. The RF algorithm has been modified and applied to the MLC tasks [30,31]. Such models are not able to learn from any dependencies within the labels and are expensive in terms of space and time during training and evaluation [29].

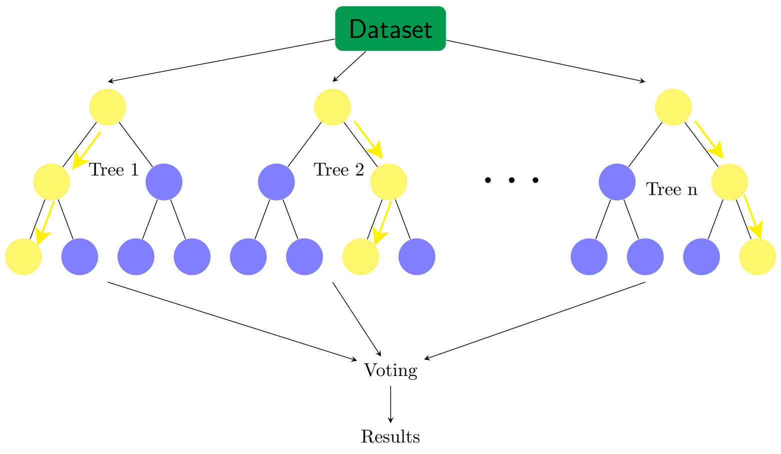


Figure 5. Random Forest model prediction pipeline.

5.4. X-BERT

Multiple deep learning-based MLC methods have been proposed. Such methods use several neural network architectures to perform the classification task. The BP-MLL was one of the first deep-learning-based method proposed by Zhang et al. [32]. This approach consists of a fully connected neural network that is trained by minimizing the pairwise ranking loss of the predicted output compared to the actual labels. Kurata et al. proposed an MLC method that is based on the Convolutional Neural Network (CNN) architecture [33]. Yang et al. introduced an application of the Recurrent Neural Network (RNN) to assign the probabilities for every label available [34]. X-BERT is an MLC method, proposed by Chang et al. [35]. The Naive, ML-kNN and ML-RF models that we presented previously are not able to learn any dependencies within the labels. X-BERT tackles this issue by combining the pretrained transformer language models with label clustering and, a linear ranking algorithm. Initially, the labels are clustered based on their syntactic and semantic meaning using ELMo [36], a deep learning word representation technique. The pretrained models are fine-tuned to assign each study to a cluster of labels. A linear ranker is then trained to order the labels within each cluster of studies by minimizing the error of the output of the trained transformer model compared to the ground truth. The pretrained models allowed us to apply such deep learning-based method to our dataset despite the small size of the training set.

5.5. Parabel

The FSD archive is likely to grow gradually in the future. Increasing the number of studies and labels available would result in an increased space and time complexity of the approaches introduced above. These issues are addressed by a subfield of MLC known as Extreme Multi-label Classification (XMC). One approach to the XMC problem consists of organising labels as leaves on a tree structure. The probability of each leaf or label is calculated by using the scores of nodes that are part of the path traveled to reach the specific leaf from the root of the tree. The Hierarchical SoftMax model was one of the first proposed approaches that was-based in the label tree structure and applied in the domain of natural language processing [37]. Applications of label tree-based approaches to the XMC problem such as DiSMEC [38] and SLICE [38] consist of training one linear classifier for each label. These methods are accurate but computationally expensive during the training and the prediction phase. The method that we selected to train and evaluate on our dataset is Parabel [4]. This approach addresses the computational issues of previous label tree-based methods and performs more efficiently by partitioning the label space through a balanced binary tree of labels. We combine this approach with the TF-IDF features extracted from the textual abstract field to represent each study.

6. Experimental Evaluation

Our task targets at identifying the optimal methodology to implement an online keyword recommendation tool for new studies listed at FSD. The factors motivating the methodology selection process included performance, space and time complexity, feasibility, and scalability of all the selected methods. We set up an experiment to quantify each factor for the selected methods and compared the results.

6.1. Experimental Setup

The approaches that we evaluated consist of Naive approach, ML-kNN, ML-RF, X-BERT, and PARABEL. The selected models were trained and evaluated with the respective training and testing data that was obtained from the stratified random splitting technique, for both English and Finnish languages.

The Naive approach that we introduced as a baseline model uses the BoW representation to represent the textual information of the abstract field of every study. Our implementation of this method consists of representing each study's abstract with a BoW feature vector. During the testing phase we assign a label to a test study if the specific label's respective counter in the abstract representation vector is larger than zero.

The ML-kNN model is defined with the parameter k. We tested a set of integer values for this parameter ranging from 1 to square root of n where n is the size of the training dataset. We tested these values of k for both language training sets combined with each text representation method. For each value of k we measured the sum of absolute errors at predicting the labels in the training set. The absolute error was lower at $k = 3$ and increased proportionally as the value of k increased. We obtained the same results on all 8 language-to-feature-extraction-method combinations that we evaluated.

Similar to the ML-kNN evaluation, the ML-RF model performance was trained and evaluated for each language's training set combined with each text representation methods that we used. The multi-label random forest model is configured through multiple parameters. Searching for the optimal set of parameters to initialize the model for our specific problem, similar to what we did for the ML-kNN model, may result in the model learning 'too well' and overfitting. Ideally, another set of data would be used to validate the results and assess the level of overfitting while optimizing the parameters. It was not feasible to conduct such parameter optimization given the small size of our data. We initialized the random forest model with the default values [29] for each the 8 language-to-feature-extraction-method combinations that we evaluated.

We trained the X-BERT model using the pretrained *bert-base-multilingual-uncased* model for the training datasets of each language [16]. The X-BERT algorithm includes a preprocessing step of data, so we provided it with the raw abstract field and the labels of the training data. We tested various initialization configurations for the model to identify the optimal training settings. With a learning rate equal to 1e-5 and the training batch size to 8 the model achieved better convergence rate with smaller error on training data, for both languages. The model converged in two epochs due to the small size of our training datasets.

The PARABEL model uses 4 parameters during the initialization. We found that updating the default settings did not improve the performance of the model during the training phase. We used the following values selected as default from the method's authors: number-of-trees = 3, maximum-number-of-paths-traversable-in-a-tree = 10, maximum-number-of-labels-in-a-leaf-node = 100, misclassification-penalty-for-internal-and-leaf-node-classifiers = 10. The TF-IDF features extracted from the abstract fields were used to represent each respective study in the training data. Two separate model instances were trained and evaluated on the respective English and Finnish data.

6.2. Evaluation Metrics

The predicted labels obtained from an MLC model are used to implement a semi-automatic labelling system for FSD. The model will predict a set of labels for a new study and the responsible clerk will inspect the output and update it accordingly. An ideal classification model would minimize the amount of input from the clerk. Our evaluation metrics selection process was guided from our specific use case.

The methods were evaluated using the Precision, Recall, F1 score and the Normalized Discounted Cumulative Gain denoted as nDCG@*k* where $k \in (1{,}2{,}3)$ [39].

Precision, Recall and F1 score measure the performance of a model at predicting all the labels, ignoring their order. The ranking-based nDCG@*k* evaluation metric [39] is used in the related MLC tasks to measure the label ranking performance of a model [3,40,41].

Table 2 provides an illustration of the confusion matrix mapping the predictions of a classification model versus the ground truth.

Table 2. The classification confusion matrix.

		Ground Truth	
		Positive(1)	**Negative(0)**
Predicted	Positive(1)	True Positive	False Positive
	Negative(0)	False Negative	True Negative

The total true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN) values are counted for each prediction. These values are used to calculate the precision and recall. Precision is the ratio of true positives among the positive predictions and it is calculated as:

$$Precision = \frac{TP}{TP + FP}. \tag{6}$$

The Recall score is calculated as the ratio of true positives among all the positive labels:

$$Recall = \frac{TP}{TP + FN}. \tag{7}$$

A classification model may have a high precision score by only predicting the positive labels that the model is more confident on. Conversely, a model can achieve a high recall score by predicting many labels as positive, with a lower confidence. $F1$ score is a classification model evaluation metric that balances the precision and recall score using the following equation:

$$F1_{score} = 2 * \frac{Precision * Recall}{Precision + Recall}. \tag{8}$$

The ranking-based *nDCG@k* evaluation metric [39] measures the label ranking performance of a model [3,40,41]. Given our use case, labels with higher *nDCG@k* score are considered more likely to be assigned to the respective study. Let the label vector L of an entity x in the training set be represented as $y = (1, 0, \ldots, 1)$ and the predicted values as $\hat{y} = (0.7, 0.1, \ldots, 0.9)$. The *DCG@k* score is computed as follows:

$$DCG@k = \sum_{i \in rank_k(\hat{y})} \frac{y_i}{log_2(i+1)}, \tag{9}$$

where the $rank_k(\hat{y})$ function returns the top k labels with the highest positive prediction confidence. The ideal *DCG@k* score is calculated by ranking the labels according to the ground truth and is defined as:

$$iDCG@k = \sum_{i=1}^{min(k,\|y\|_0)} \frac{y_i}{log_2(i+1)}, \tag{10}$$

Finally, the normalized *DCG* score is obtained from the ratio of *DCG@k* and *iDCG@k* score:

$$nDCG@k = \frac{DCG@k}{iDCG@k}. \tag{11}$$

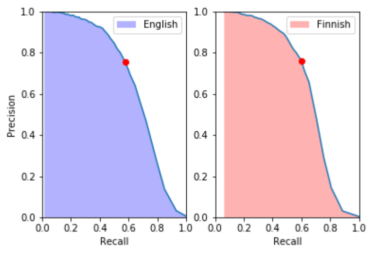
6.3. Results

We evaluated the selected models combined with the feature extraction methods on the English and Finnish testing datasets. Besides the Naive method which gives its predictions in a vector of 0-1 integers, all the approaches we selected can provide a probability distribution for the target labels i.e., $Preds_M(s) = [0.1, 0.5, \ldots, 0.02]$, where M is the model providing the predictions and s is the target study. Deciding whether to assign a label to a study given the predicted probability distribution can be solved by arbitrarily selecting a threshold value. The labels with the corresponding probability value greater than or equal to the threshold are assigned to the target study. With a threshold t = 0.3, the previous example would be converted to $Preds_M(s) = [0, 1, \ldots, 0]$. Selecting the threshold value results in a tradeoff between the precision and recall score. We tested 100 different threshold values spaced equally within the range [0, 1] for every combination of our selected models with the feature representation methods and plot the precision-recall curves, respectively.

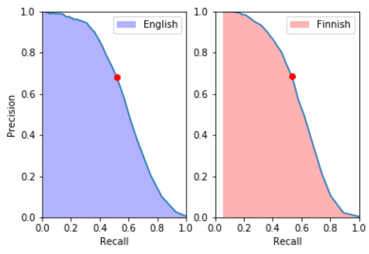
In Figure 6, we provide the precision-recall curves that resulted by combining the ML-kNN model with the feature extraction methods. The ML-kNN model's predictions are inferred from the k'th most similar studies. Our selected value of k = 3 explains the fragmentation of the precision-recall curves for this model. The plots suggest that the performance of the model on the English data is consistently higher, but the differences are small. Combining the ML-kNN model with the TF-IDF features provides higher prediction performance in terms of precision and recall. We plotted the precision-recall curve for each combination of our feature extraction techniques with the ML-RF model in Figure 7. The MLRF provides smoother precision-recall curves compared to the ML-kNN model. The resulting plots obtained from evaluating ML-RF for studies on both languages' test data are similar.

We plot the precision-recall curves for the predictions obtained from X-BERT and PARABEL in Figure 8. There are minor differences between the predictions obtained for each language. The predictions of PARABEL method provide a higher recall on average which explain the wider distribution of the area under the plot along the recall score axis.

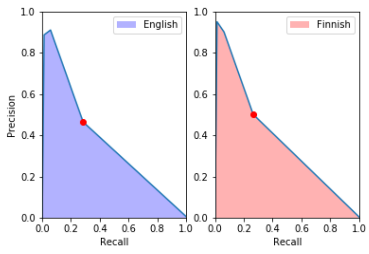
For the Naive approach, we did not select any threshold since its prediction output consists of a vector with 0 and 1 binary values. For the other methods we considered the optimal threshold values which maximized the F1 score for each respective combination of the models with the features. The highest obtained F1 score values are marked with a red dot in each respective precision-recall curve presented in Figures 6–8.



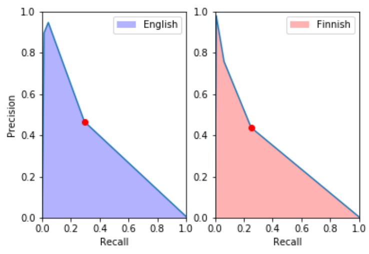
(**a**) Using TF-IDF features.



(**b**) Using FastText features.

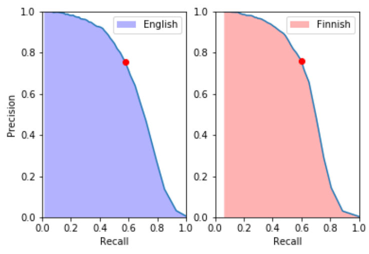


(**c**) Using BERT features.

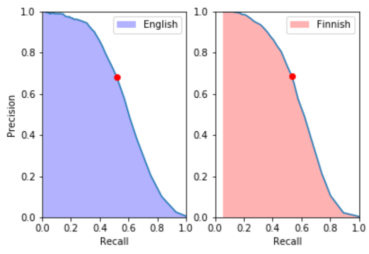


(**d**) Using Multi-BERT features.

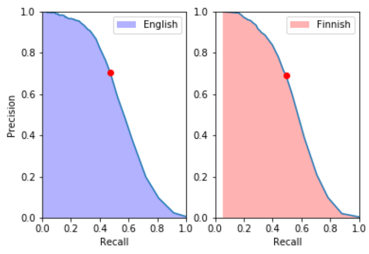
Figure 6. Precision-Recall curves obtained from evaluating the combination of the ML-kNN model with the feature extraction methods. The red dots represent the point where the corresponding maximum F1 score was measured for every plot.



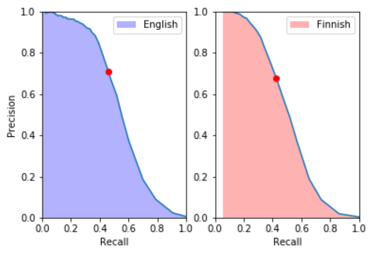
(**a**) Using TF-IDF features.



(**b**) Using FastText features.

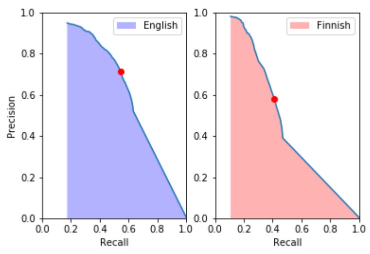


(**c**) Using BERT features.

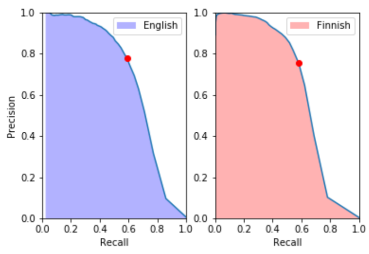


(**d**) Using Multi-BERT features.

Figure 7. Precision-Recall curves obtained from evaluating the combination of the ML-RF model with the feature extraction methods. The red dots represent the point where the corresponding maximum F1 score was measured for every plot.



(**a**) X-BERT method



(**b**) PARABEL method

Figure 8. Precision-Recall curves obtained from the evaluation of X-BERT and Parabel methods where the red dots represent the corresponding maximum F1 score for each model.

The calculated maximum F1 scores along the respective optimal precision and recall values for each model are shown in Table 3. The Naive approach predictions resulted in the smallest accuracy scores that we calculated. For the English metadata, the PARABEL method achieved the highest accuracy according to the precision, recall and F1 score metrics. The next highest performance was obtained from the combination of the ML-RF method with the TF-IDF features according to our metrics. In the Finnish test data, PARABEL was outperformed by a small margin from the combination of ML-RF model with TF-IDF features. The ML-kNN method attained higher F1 score when combined with the TF-IDF features. Besides the ML-RF method, all the other approaches that we tested achieved smaller scores on the Finnish data. This is explained by the larger number of labels available in the Finnish data compared to the English data. The models achieved higher accuracy scores when combined with the FastText representations compared to their respective combinations with BERT and Multi-BERT extracted features. In the Finnish data, the models combined with BERT extracted features provided higher scores consistently compared to their respective combinations with Multi-BERT features.

In Table 4, we report the evaluation results in terms of ranking accuracy. The PARABEL algorithm outperformed the other methods based on the ranking nDCG scores for top 1, 3 and 5 labels, in English and Finnish data. X-BERT provided the second-best-ranking scores for the English data but was outperformed by the combination of ML-RF with the TF-IDF features in the Finnish data. The *bert-base-multilingual-uncased* pretrained model that we used to initialize X-BERT has been trained in textual data from multiple languages where English data were dominant [16]. For ML-kNN and ML-RF the combination with language specific BERT features yielded better ranking scores than the combination with Multi-BERT features.

Table 3. Accuracy scores obtained from the evaluation of the models in the English and Finnish test datasets.

	Features	Precision	Recall	F1 Score
English metadata				
Naive	BoW	0.19	0.13	0.32
ML-kNN	TF-IDF	0.31	0.57	0.4
	FastText	0.30	0.52	0.38
	BERT	0.28	0.46	0.35
	Multi-BERT	0.29	0.47	0.36
ML-RF	TF-IDF	0.58	0.75	0.65
	FastText	0.52	0.68	0.59
	BERT	0.47	0.70	0.57
	Multi-BERT	0.46	0.71	0.56
X-BERT	-	0.54	0.71	0.62
PARABEL	TF-IDF	**0.59**	**0.78**	**0.67**

Table 3. *Cont.*

	Features	Precision	Recall	F1 Score
Finnish metadata				
Naive	BoW	0.15	0.11	0.24
ML-kNN	TF-IDF	0.28	0.56	0.38
	FastText	0.27	0.49	0.35
	BERT	0.27	0.50	0.35
	Multi-BERT	0.25	0.44	0.32
ML-RF	TF-IDF	**0.60**	**0.76**	**0.67**
	FastText	0.53	0.69	0.60
	BERT	0.49	0.69	0.58
	Multi-BERT	0.43	0.68	0.52
X-BERT	-	0.41	0.58	0.48
PARABEL	TF-IDF	0.58	0.75	0.66

Table 4. Ranking scores obtained from the evaluation of the models in the English and Finnish test datasets.

	Features	nDCG@1	nDCG@3	nDCG@5
English metadata				
ML-kNN	TF-IDF	0.65	0.60	0.58
	FastText	0.60	0.56	0.54
	BERT	0.57	0.53	0.51
	Multi-BERT	0.54	0.51	0.49
ML-RF	TF-IDF	0.77	0.74	0.71
	FastText	0.71	0.66	0.64
	BERT	0.65	0.62	0.60
	Multi-BERT	0.62	0.60	0.58
X-BERT	-	0.79	0.75	0.72
PARABEL	TF-IDF	**0.83**	**0.78**	**0.76**
Finnish metadata				
ML-kNN	TF-IDF	0.65	0.61	0.60
	FastText	0.60	0.57	0.55
	BERT	0.59	0.57	0.56
	Multi-BERT	0.54	0.52	0.50
ML-RF	TF-IDF	0.78	0.75	0.73
	FastText	0.75	0.71	0.69
	BERT	0.72	0.68	0.66
	Multi-BERT	0.64	0.61	0.58
X-BERT	-	0.69	0.64	0.60
PARABEL	TF-IDF	**0.80**	**0.76**	**0.74**

7. Conclusions

In this work, we applied MLC techniques for textual data in the domain of social sciences. Our dataset was collected from the Finnish Social Science Data Archive and was comprised of 2968 research studies conducted in the domain of social sciences. We extracted features from the abstract attribute of each study by using the following text representations methods: the Bag of Words (BoW), TF-IDF term weighting and pretrained word embeddings obtained from FastText and BERT models. The dataset was split into train and test sets using an iterative stratified sampling method to maintain a similar label distribution among each set.

We defined a Naive approach as a baseline MLC method and selected the following MLC techniques to train and evaluate on our dataset: ML-kNN, ML-RF, X-BERT and

PARABEL. We combined the selected models with the feature extraction techniques and trained them on the training data. We measured the performance of the selected methods on the test dataset by calculating the Precision, Recall and F1 score metrics of the respective predictions. We calculated the nDCG score to measure the label ranking performance of each method.

We deployed a Multi-label Random Forest classifier using the TF-IDF features in the label recommending tool at FSD. We justified our decision by considering the evaluation results, the size of our dataset and the system running the tool. According to the accuracy scores that were measured during the evaluation process, the ML-RF combined with the TF-IDF features achieved best performance on the Finnish metadata and was outperformed by the PARABEL model on the English metadata by 1-3% difference based on the Precision, Recall and F1 scores. The ranking scores provide similar evidence in the performance of our selected method compared to the Parabel and X-BERT models. Albeit Parabel and X-BERT are intended to perform better in the Extreme Multi-label Classification task, they achieved top scores on our task.

The implementation of the ML-RF method was technically more feasible, given the constraints set by the system running the study labelling tool. For our task, the marginal accuracy difference between ML-RF and the other selected methods did not justify the other implementation costs. These models may be considered when the number of studies available at FSD archive increases, thus increasing the accuracy gain at predicting the set of labels for new studies.

Author Contributions: Conceptualization, E.S., J.H. and K.S.; methodology, E.S., J.H. and K.S.; software, E.S.; validation, E.S.; formal analysis, E.S., J.H. and K.S.; investigation, E.S., J.H. and K.S.; resources, E.S.; data curation, E.S.; writing—original draft preparation, E.S., J.H. and K.S.; writing—review and editing, E.S., J.H. and K.S.; visualization, E.S., J.H. and K.S.; supervision, J.H. and K.S.; All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Academy of Finland (319319).

Data Availability Statement: The data used for this work is available at the official website of the Finnish Social Science Data Archive .

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Loza Mencía, E.; Fürnkranz, J. Efficient pairwise multilabel classification for large-scale problems in the legal domain. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5212, pp. 50–65. [CrossRef]
2. Chalkidis, I.; Fergadiotis, M.; Malakasiotis, P.; Aletras, N.; Androutsopoulos, I. Extreme Multi-Label Legal Text Classification: A case study in EU Legislation. *arXiv* **2019**, arXiv:1905.10892.
3. You, R.; Zhang, Z.; Wang, Z.; Dai, S.; Mamitsuka, H.; Zhu, S. AttentionXML: Label Tree-based Attention-Aware Deep Model for High-Performance Extreme Multi-Label Text Classification. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 5820–5830.
4. Prabhu, Y.; Kag, A.; Harsola, S.; Agrawal, R.; Varma, M. Parabel: Partitioned label trees for extreme classification with application to dynamic search advertising. In Proceedings of the World Wide Web Conference (WWW), Lyon, France, 23–27 April 2018; Association for Computing Machinery, Inc.: New York, NY, USA, 2018; pp. 993–1002. [CrossRef]
5. Papanikolaou, Y.; Tsoumakas, G.; Laliotis, M.; Markantonatos, N.; Vlahavas, I. Large-scale online semantic indexing of biomedical articles via an ensemble of multi-label classification models. *J. Biomed. Semant.* **2017**, *8*, 1–13. [CrossRef] [PubMed]
6. Al-Salemi, B.; Ayob, M.; Kendall, G.; Noah, S.A.M. Multi-label Arabic text categorization: A benchmark and baseline comparison of multi-label learning algorithms. *Inf. Process. Manag.* **2019**, *56*, 212–227. [CrossRef]
7. Fiallos, A.; Jimenes, K. Using reddit data for multi-label text classification of twitter users interests. In Proceedings of the 2019 6th International Conference on eDemocracy and eGovernment, ICEDEG, Quito, Ecuador, 24–26 April 2019; pp. 324–327. [CrossRef]
8. Chang, W.C.; Yu, H.F.; Zhong, K.; Yang, Y.; Dhillon, I. Taming Pretrained Transformers for Extreme Multi-label Text Classification. In Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 3163–3171.
9. Sechidis, K.; Tsoumakas, G.; Vlahavas, I. On the stratification of multi-label data. In Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Athens, Greece, 5–9 September 2011; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6913, pp. 145–158. [CrossRef]

10. Thakur, N.; Han, C.Y. Country-Specific Interests towards Fall Detection from 2004–2021: An Open Access Dataset and Research Questions. *Data* **2021**, *6*, 92. [CrossRef]
11. Bengio, Y.; Ducharme, R.; Vincent, P.; Jauvin, C. A Neural Probabilistic Language Model. J*Ournal Mach. Learn. Res.* **2003**, *3*, 1137–1155. [CrossRef]
12. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. In Proceedings of the 1st International Conference on Learning Representations, ICLR 2013—Workshop Track Proceedings, Scottsdale, AZ, USA, 2–4 May 2013.
13. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
14. Ali, F.; Ali, A.; Imran, M.; Naqvi, R.A.; Siddiqi, M.H.; Kwak, K.S. Traffic accident detection and condition analysis based on social networking data. *Accid. Anal. Prev.* **2021**, *151*, 105973. [CrossRef]
15. Mikolov, T.; Grave, E.; Bojanowski, P.; Puhrsch, C.; Joulin, A. Advances in Pre-Training Distributed Word Representations. In Proceedings of the LREC 2018—11th International Conference on Language Resources and Evaluation, Miyazaki, Japan, 7–12 May 2018; pp. 52–55.
16. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the NAACL HLT 2019—2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies—Proceedings of the Conference, Minneapolis, MI, USA, 2–7 June 2019; pp. 4171–4186.
17. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Neural Information Processing Systems Foundation: San Francisco, CA, USA 2017; Volume 2017, pp. 5999–6009.
18. Virtanen, A.; Kanerva, J.; Ilo, R.; Luoma, J.; Luotolahti, J.; Salakoski, T.; Ginter, F.; Pyysalo, S. Multilingual is not enough: BERT for Finnish. *arXiv* **2019**, arXiv:1912.07076.
19. Khattak, F.K.; Jeblee, S.; Pou-Prom, C.; Abdalla, M.; Meaney, C.; Rudzicz, F. A survey of word embeddings for clinical text. *J. Biomed. Inform.* **2019**, *100*, 100057. [CrossRef]
20. Boutell, M.R.; Luo, J.; Shen, X.; Brown, C.M. Learning multi-label scene classification. *Pattern Recognit.* **2004**, *37*, 1757–1771. [CrossRef]
21. Read, J.; Pfahringer, B.; Holmes, G.; Frank, E. Classifier chains for multi-label classification. *Mach. Learn.* **2011**, *85*, 333–359. [CrossRef]
22. Hüllermeier, E.; Fürnkranz, J.; Cheng, W.; Brinker, K. Label ranking by learning pairwise preferences. *Artif. Intell.* **2008**, *172*, 1897–1916. [CrossRef]
23. Tsoumakas, G.; Vlahavas, I. Random k-Labelsets: An Ensemble Method for Multilabel Classification. In Proceedings of the European Conference on Machine Learning, Warsaw, Poland, 17–21 September 2007; Springer: Berlin/Heidelberg, Germany, 2007; Volume 4701, pp. 406–417. [CrossRef]
24. Freund, Y.; Schapire, R.E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* **1997**, *55*, 119–139. [CrossRef]
25. Schapire, R.E.; Singer, Y. Improved boosting algorithms using confidence-rated predictions. *Mach. Learn.* **1999**, *37*, 297–336. [CrossRef]
26. Cheng, W.; Hüllermeier, E. Combining Instance-Based Learning and Logistic Regression for Multilabel Classification. In Proceedings of the LWA 2009—Workshop-Woche: Lernen-Wissen-Adaptivitat—Learning, Knowledge, and Adaptivity, Darmstadt, Germany, 21–23 September 2009; p. 6. [CrossRef]
27. Spyromitros, E.; Tsoumakas, G.; Vlahavas, I. An Empirical Study of Lazy Multilabel Classification Algorithms. In Proceedings of the Hellenic Conference on Artificial Intelligence, Syros, Greece, 2–4 October 2008; Springer: Berlin/Heidelberg, Germany, 2008; Volume 5138, pp. 401–406. [CrossRef]
28. Zhang, M.L.; Zhou, Z.H. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognit.* **2007**, *40*, 2038–2048. [CrossRef]
29. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
30. Agrawal, R.; Gupta, A.; Prabhu, Y.; Varma, M. *Multi-Label Learning with Millions of Labels*; Association for Computing Machinery (ACM): New York, NY, USA 2013; pp. 13–24. [CrossRef]
31. Wu, X.; Gao, Y.; Jiao, D. Multi-label classification based on Random Forest algorithm for non-intrusive load monitoring system. *Processes* **2019**, *7*, 337. [CrossRef]
32. Zhang, M.L.; Zhou, Z.H. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Trans. Knowl. Data Eng.* **2006**, *18*, 1338–1351. [CrossRef]
33. Kurata, G.; Xiang, B.; Zhou, B. Improved neural network-based multi-label classification with better initialization leveraging label co-occurrence. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL HLT 2016, San Diego, CA, USA, 12–17 June 2016; pp. 521–526. [CrossRef]
34. Yang, P.; Sun, X.; Li, W.; Ma, S.; Wu, W.; Wang, H. SGM: Sequence Generation Model for Multi-label Classification. *arXiv* **2018**, arXiv:1806.04822.

35. Chang, W.C.; Yu, H.F.; Zhong, K.; Yang, Y.; Dhillon, I.S. X-BERT: eXtreme Multi-label Text Classification using Bidirectional Encoder Representations from Transformers. *arXiv* **2020**, arXiv:1905.02331.
36. Peters, M.; Neumann, M.; Iyyer, M.; Gardner, M.; Clark, C.; Lee, K.; Zettlemoyer, L. *Deep Contextualized Word Representations*; Association for Computational Linguistics (ACL): Stroudsburg, PA, USA, 2018; pp. 2227–2237. [CrossRef]
37. Morin, F.; Bengio, Y. *Hierarchical Probabilistic Neural Network Language Model*; PMLR: Boulder, CO, USA, 2005; pp. 246–252.
38. Babbar, R.; Shoelkopf, B. DiSMEC - Distributed Sparse Machines for Extreme Multi-label Classification. In Proceedings of the WSDM 2017—Proceedings of the 10th ACM International Conference on Web Search and Data Mining, Cambridge, UK, 6–10 February 2017; pp. 721–729.
39. Järvelin, K.; Kekäläinen, J. Cumulated gain-based evaluation of IR techniques. *ACM Trans. Inf. Syst.* **2002**, *20*, 422–446. [CrossRef]
40. Poerner, N.; Schutze, H.; Schutze, S. Multi-View Domain Adapted Sentence Embeddings for Low-Resource Unsupervised Duplicate Question Detection. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, China, 3–7 November 2019.
41. Eberhard, L.; Walk, S.; Posch, L.; Helic, D. Evaluating Narrative-Driven Movie Recommendations on Reddit. In Proceedings of the 24th International Conference on Intelligent User Interfaces, Los Angeles, CA, USA 17–20 March 2019. [CrossRef]