TUM Data Innovation Lab

Munich Data Science Institute

Technical University of Munich

&

Alyne GmbH

Project: Enterprise Risk Autopilot

Authors	Ekrem Alper Kesen, Ky Thoai Pham, Nikita Okorokov, Syed Husain Mustafa
Mentor(s)	M.Sc. Egor Labintcev, M.Sc. Bhushan Chaudhari, M.Sc. Shraddha Beedu
Project Lead	Dr. Ricardo Acevedo Cabra (Munich Data Science Institute)
Supervisor	Prof. Dr. Massimo Fornasier (Munich Data Science Institute)

Aug 2022

Abstract

The task of policy-based risk assessment is of vital importance to the financial and legal viability of an organization as well as to its credibility and trust amongst the public at large. In this project, we aim to map reference statements from policies to a risk register and identify the most likely risks that may result from their flawed implementation. For this project, we have made use of a registry of risks provided to us by Alyne GmbH that was crafted by legal experts familiar with governing regulations. In place of policies from a particular firm, we make use of a generic list of policy statements provided to us by Alyne GmbH. The data from these two sets are processed using several approaches, namely, natural language inference, contrastive learning, and approximate nearest neighbor search. The results from these aforementioned approaches are then passed to a gradient boosting algorithm that aggregates relevancy scores from these models to come up with a final ranking for each user-provided reference statement.

Contents

1 Introduction

1.1 Motivation

According to a recent PwC report [1], 86% of global firms expect GDPR non-compliance to negatively affect their operations. Due to fines approaching 4% of turnover for most firms in 2019, compliance is of utmost importance. Globally, Ernst & Youngs estimates that GDPR compliance costs $7.8 billion for the 500 largest corporations [2]. As challenging as the GDPR and other nation-specific regulations are to a firm's profitability, these regulations also present a great opportunity to automate the risk assessment process [3]. Following the financial crisis in 2008, firms disproportionately focused on market and credit risks, while operations risks were handled by separate business units [4]. As a result, it becomes difficult to determine how risks relate in real-time. By automating risk assessment, firms can learn risk interdependencies, whilst focusing their efforts on their core business objectives.

1.2 Project goals

This project aims to identify relevant risks associated with a given policy statement. Using Alynes' risk registry, we identify which risks qualify as "relevant risk" from multiple perspectives. As a first step, we establish a few baselines using cosine similarity and Meilisearch. After developing our baselines, each of us assessed the suitability of several state-of-the-art approaches to our task. Our final report includes the approaches that outperform our baseline models on a set of metrics, while the appendix contains approaches that did not meet this criterion. Following our approaches, we apply Light Gradient Boosting (LightGBM) which ranks risks in order of relevance for a given policy statement.

1.3 Prior works

We seek to solve a multilabel problem where each risk is a label, and each reference can be mapped to several risks. A 2020 survey report by Qaraei et. al [5] highlights how most extreme multilabel classification models achieve a performance below linear classifiers. Although recent models ([6], [7]) produce superior results, the difficulty of training such models without GPUs and the inability to incorporate risk attributes have kept us from investing much time and effort in this area. The Appendix presents results based on simpler multilabel models. Instead, we used lexical matching and approximate nearest neighbor search techniques. We make use of TF-IDF embeddings, sentence transformers, and embeddings from FastText models trained on GDPR texts. Contrastive learning methods such as SimCSE[8] and natural language inference models are utilized considering the risk mapping problem as a text similarity problem between policy statements and risk descriptions. Additionally, we employ a state-of-the-art approximate nearest neighbors search algorithm based on Navigable Small Worlds.

2 Data Analysis

2.1 Data description and statistical data exploration

Before starting the project, we investigated and analyzed the data provided to us by Alyne GmbH to understand patterns and characteristics.

Risk database Alyne GmbH provided us with two datasets. The first dataset contains a risk database which is describing 1041 risks from the Alyne GmbH registry.

Field	Description
riskId	Unique identifier of the risk
type	Type of the risk
en_US_description	Description of the risks in US
en_GB_description	Description of the risks in GB
en_DE_description	Description of the risks in DE
impact	Score of the risk. Represents its cruciality.
riskLinks	Links from the current risk to other risks
isCore	Label for the core risks
pathsToLeaves	Paths from other risks to the current risk
pathsToCore	Paths from current risk to core risks
isLeaf	Label for the leaf risks

Table 1: Description of the fields of risk database

Distribution of risks per risk type is given below:

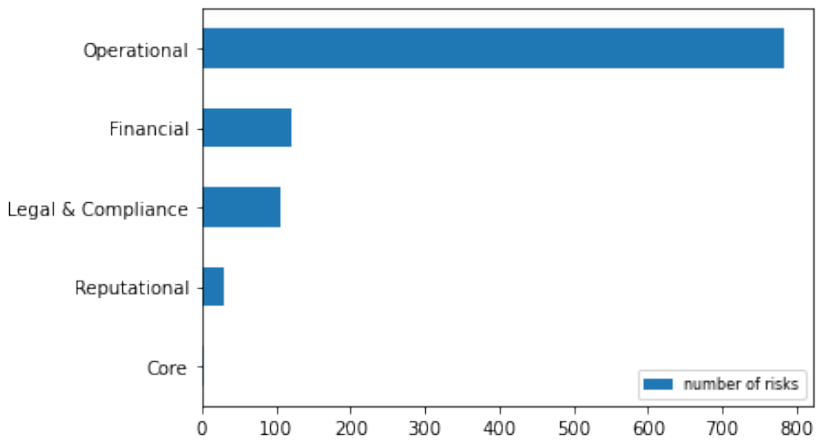

Figure 1: Distribution of risk counts per type in the risk database

The most common types of risks are Operational, Financial and Legal Compliance. The least common types are Core risks and Reputational risks.

We can also look at the distribution of the word counts in risk descriptions.

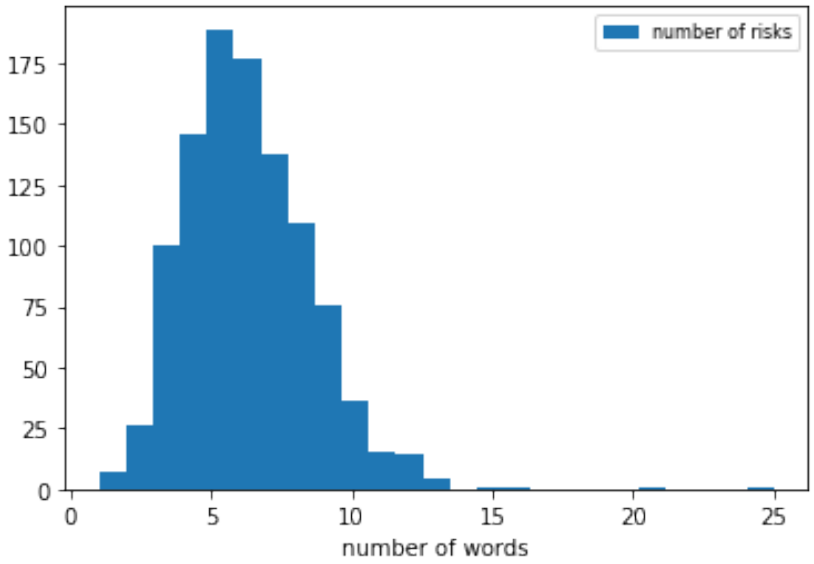

Figure 2: Distribution of word counts in risk descriptions in the risk database

As we can see from the plot, on average the risk is described by six or seven words.

Reference dataset The second dataset consists of 3265 references mapping to control statements and corresponding risks.

Field	**Description**
reference	Text of the reference
control statement	Text of the control statement
riskId	Unique identifier of the risk
type	Type of the risk
en_US_description	Description of the risks in US
en_GB_description	Description of the risks in GB
en_DE_description	Description of the risks in DE
impact	Score of the risk. Represents its cruciality.
riskLinks	Links from the current risk to other risks
isCore	Label for the core risks
pathsToLeaves	Paths from other risks to the current risk
pathsToCore	Paths from current risk to core risks
isLeaf	Label for the leaf risks

Table 2: Description of fields of reference dataset

There are six references which don't have any mapping to any risk, so we excluded these references from further analysis and worked with the rest of 3259 references. Also, it appears that only 554 out of 1041 risks are presented in the second dataset. Distribution of risks per type is given below:

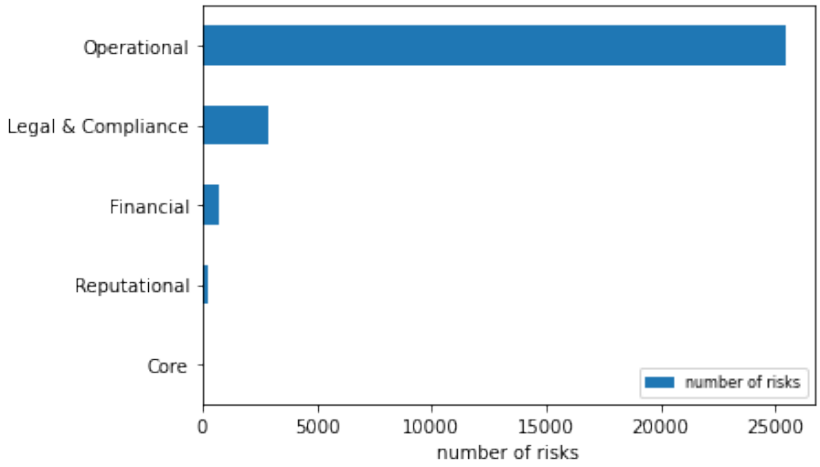

Figure 3: Distribution of risk counts per type in the reference dataset

From the plot, we can see that in the second dataset, types of risks are distributed almost in the same way as in the risk database. As our task is to map a text of the reference to one or multiple risks, we also need to look at the distribution of risk counts per reference in the second dataset:

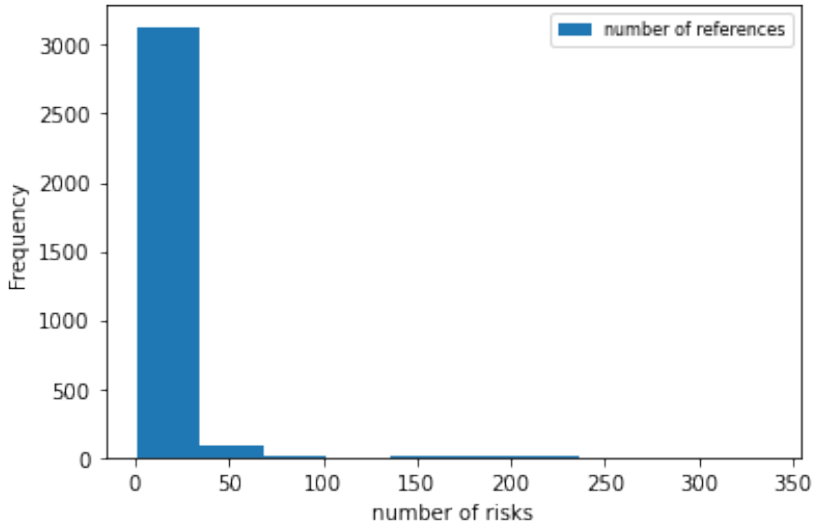

Figure 4: Distribution of risk counts per reference in the reference dataset

We see that the distribution of risk counts per reference is very skewed to the left. Most references have less than 50 risks, but some of the references have more than 100 risks. On average, we have 9 risks per reference. The minimum number of risks per reference is 1 and the maximum number of risks per reference is 337. The median is 4 risks per reference. It is also interesting to look at the distribution of reference counts per risk:

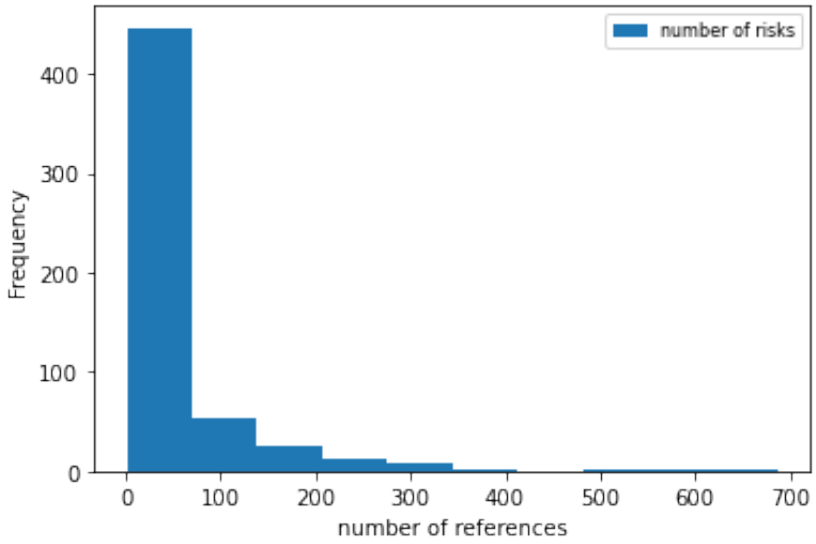

Figure 5: Distribution of reference counts per risk in the reference dataset

As we can see from the plot, the distribution of reference counts per risk is also skewed to the left. On average, there are 53 references per risk. The minimum is 1 reference per risk, the maximum is 687 references per risk. The median is 24 references per risk.

Additionally, we looked at the distribution of the word counts per reference:

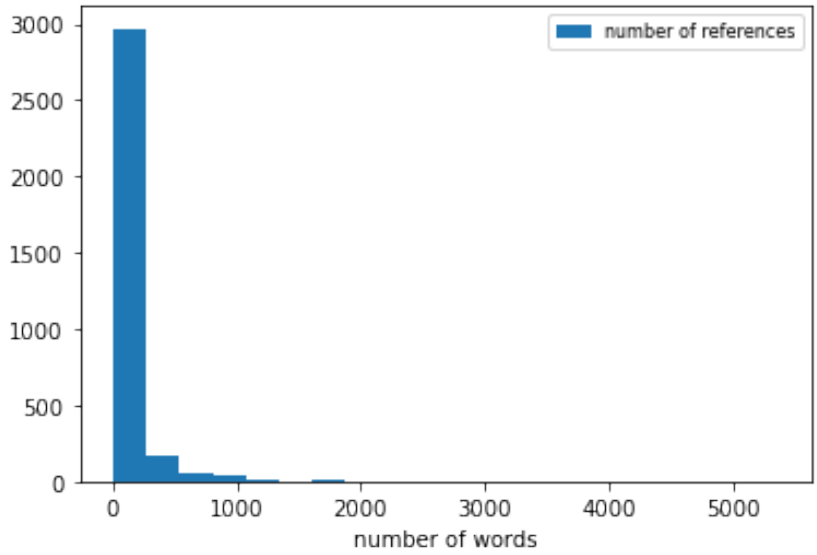

Figure 6: Distribution of word counts per reference in the reference dataset

We can see that on average a reference is described by 115 words. The minimum number of words in reference is 2 and the maximum number is 5234. Also, we looked at the distribution of word counts per control statement:

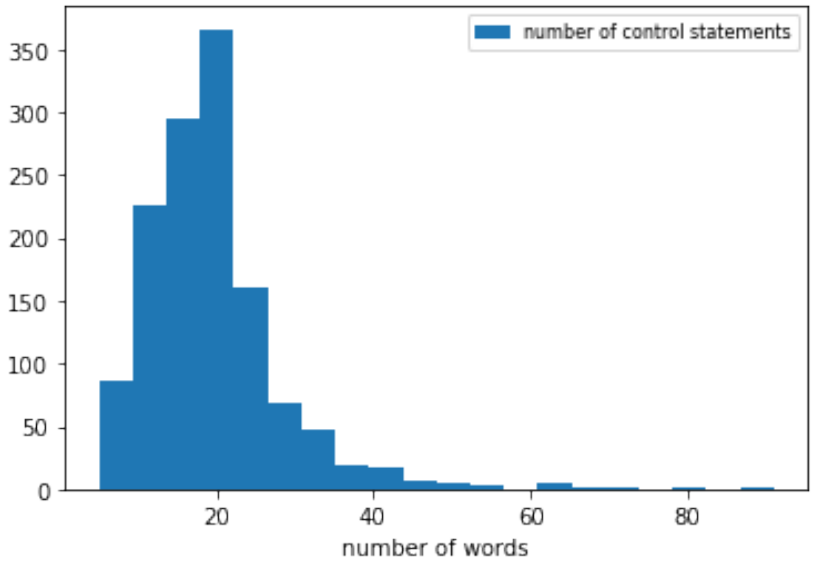

Figure 7: Distribution of word counts per control statement in the reference dataset

On average, a control statement is described by 19 words. The minimum number of words in the control statement is 5 and the maximum number is 91.

2.2 Data splitting

To evaluate our models objectively, we created fixed train, validation, and test sets. We split our data such that references in the test set are not in the train and validation set. Additionally, each reference in the validation and test set has less than 10 risks. In the end, we have 2513 references for the train set, 246 references for the validation, and 500 references for the test set. The train set contains 16369 risks and includes 552 unique risks out of 554 total unique risks in the reference dataset and each reference has 6.51 risks on average. The validation set contains 787 risks of 229 unique risks and each reference has 3.19 risks on average. The test set contains 1568 risks of 362 unique risks and each reference has 3.13 risks on average.

2.3 Metrics

To evaluate and compare our approaches, we considered several metrics such as precision (P@K), recall (R@K), mean average precision (MAP@K), and r-precision (RP@K) scores for top-K prediction. For our problem, we mostly considered R@K and MAP@K scores since the number of predictions (K) is most of the time higher than the number of true risks. RP@K score equals R@K when K is chosen as a number that is higher than 10 since the number of risks for each reference in the test set is always less than 10. The calculation of each of these metrics is given below:

$$R@K(\text{Recall}) = \frac{1}{T}\sum_{t=1}^{T}\frac{S_t(K)}{R_t}$$

$$P@K(\text{Precision}) = \frac{1}{T}\sum_{t=1}^{T}\frac{S_t(K)}{K}$$

$$RP@K(\text{Recall-Precision}) = \frac{1}{T}\sum_{t=1}^{T}\frac{S_t(K)}{\min(K, R_t)}$$

$$AP_t@K = \frac{1}{K}\sum_{i=1}^{K}\frac{S_t(K)}{K}$$

$$MAP@K = \frac{1}{T}\sum_{t=1}^{T} AP_t@K$$

where T is the total number of test documents, K is the number of labels to be selected per document, $S_t(K)$ is the number of correct labels among those ranked as top K for the t-th document, and R_t is the number of gold labels for each document.

3 Methods

3.1 Elasticsearch

Elasticsearch [9] is a distributed, RESTful search and analytics engine and document store which allows storing, searching and analyzing huge volumes of data quickly. In Elasticsearch, documents are stored as structured data encoded in JSON instead of rows like in relational databases. Indices are a collection of documents that have similar characteristics such as logically related documents and similar to databases in relational databases. The inverted index is a data structure that stores a mapping from content to its locations in documents. Instead of searching text directly, it searches an index. Mapping each unique token to a list of documents containing that word makes it possible to locate documents within given keywords very quickly. There are many alternative search engines such as Meilisearch [10], Typesense [11], Algolia [12], and Redisearch [13].

Meilisearch is another open-source alternative to Elasticsearch. Although it provides fewer features, it is more lightweight than Elasticsearch, works out-of-the-box well, it can return search results very quickly.

3.2 Cosine similarity with TF-IDF

TF-IDF The term frequency-inverse document frequency (TF-IDF) is a popular weighting scheme that is used to improve the simple count-based data from the bag of words model. It is the product of two values, the term frequency and the inverse document frequency

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \text{IDF}(t)$$

where $\text{TF}(t, d)$ is term frequency which is the number of times term t appears in a doc d and $\text{IDF}(t)$ is the inverse document frequency, which is estimated by

$$\log \frac{1+n}{1+df(d,t)} + 1$$

where n is # of documents and $df(d, t)$ is document frequency of the term t

Cosine Similarity Given two vectors of attributes, A and B, the cosine similarity, $\cos(\theta)$, is represented using a dot product and magnitude as

$$cos(\theta) = \frac{A \cdot B}{||A||\ ||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

3.3 Sentence Transformer

Sentence Transformers is a framework for calculating dense vector representations for sentences and paragraphs. The models are based on transformer networks like BERT [14], RoBERTa [15], XLM-RoBERTa and achieve state-of-the-art performance in various tasks. We can use this framework to compute sentence embeddings for given texts. These embeddings can then be compared with measures of similarity to find groups of sentences

which are similar in meaning. In this project, we mainly use sentence-transformers based on MPNet models.

3.4 Natural Language Inference

Natural language inference task is given sentence pairs which are called premise and hypothesis, the model predicts whether the hypothesis about the premise is true (entailment), false (contradiction), or undetermined (neutral). Before the success of deep learning approaches, symbolic and statistical approaches such as deterministic rules which exploit handcrafted lexical features and word matching [16], decision trees [17], and naive Bayes classifiers [18] were applied to solve the natural language inference problem and comparable approaches were applied on Recognizing Textual Entailment (RTE) [19] challenges. Benchmarks with a large amount of data allowed researchers to address the problem with deep neural networks where non-contextual word embeddings models such as GloVe [20], word2vec [21] or pretrained contextual representations such as ELMo [22], BERT embeddings were used together with LSTMs, CNNs or transformer-based architectures [23]. Additional to RTE, benchmark datasets include SNLI [24], MNLI [25], SciTail [26] datasets. Please check the survey paper for more details [27].

3.5 Contrastive Learning

Contrastive learning (CL) is a paradigm that allows models to learn about data by gathering similar examples closer and pushing different examples further away without the use of labels [28]. It greatly enhances the performance of the model by training in a self-supervised way and is successfully applied for computer vision [29], and natural language processing (NLP) tasks [21], [8]. For the training of contrastive learning, negative samples and positive samples should be determined. We can define contrastive loss as below:

$$L_{contrastive} = -\log \frac{\exp(sim(q, k_+)/\tau)}{\exp(sim(q, k_+)/\tau) + \exp(sim(q, k_-)/\tau)}$$

where q is reference sample, k_+ is positive sample, k_- is negative sample, $sim()$ is similarity function, τ is temperature coefficient. Contrastive loss is low when easy negative samples are chosen and high when hard negative samples are chosen. Therefore, choosing negative samples properly has an impact on the performance of the model after training.

SimCSE SimCSE [8] is a contrastive learning model which enhances state-of-the-art sentence embeddings by encoding the same data twice. It achieved a better Spearman's correlation score than SBERT on semantic textual similarity (STS) tasks. It makes use of NLI datasets where entailment pairs are used as positive samples and contradiction pairs are used as negative samples. It can be used both in supervised and unsupervised data. An example of the SimCSE approach is given below:

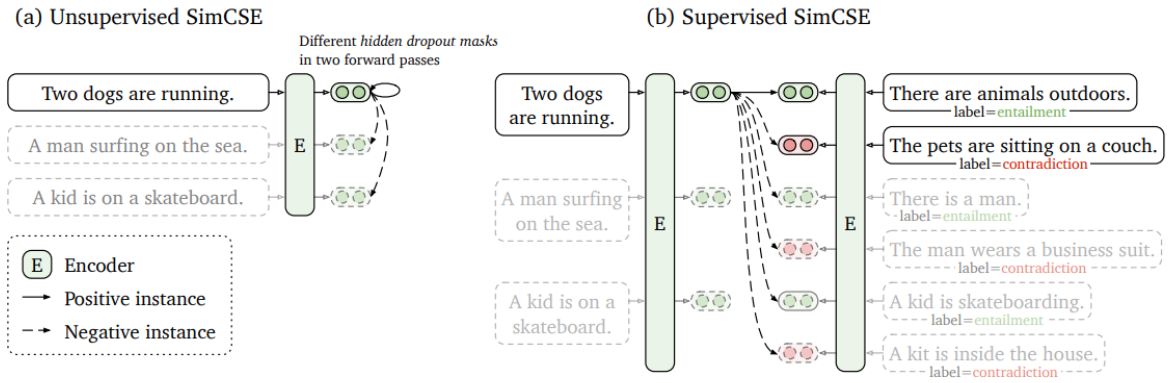

Figure 8: Unsupervised and supervised SimCSE approach

3.6 Approximate Nearest Neighbor Search

A K Approximate Nearest Neighbor (K-ANN) approach is referred to as Hierarchical Navigable Small Worlds Graph [30] is used to perform semantic similarity search.

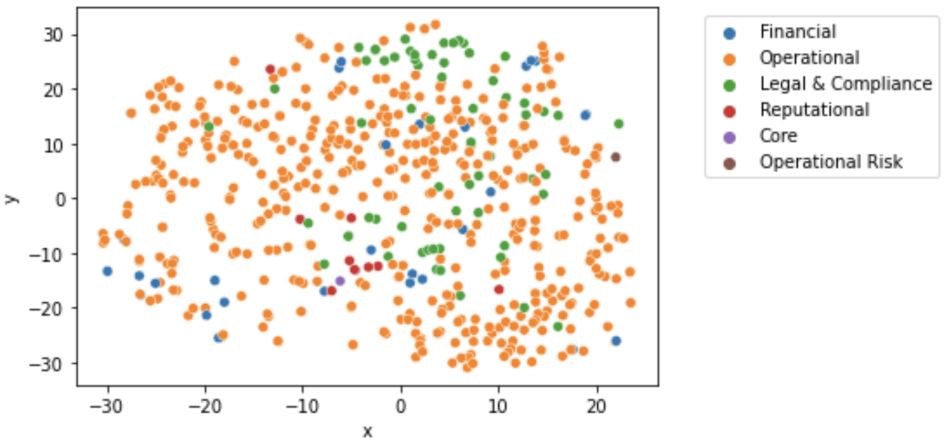

Figure 9: t-SNE Scatter Plot of 554 Risks

Traditionally, K-NN has been used to retrieve semantically similar documents [31]. However, the KNN approach does not scale well for very large datasets; furthermore, the K-NN approach assumes a defined distance function, which does not work for graphs with documents at varying distance scales. For large datasets, an exact KNN search is also associated with intractable time complexities.
For these reasons, we have considered using a K-ANN search algorithm. Since risks in our registry are clustered very closely together, we opt for the HNSW approach instead of the greedy proximity graph method that performs poorly on such data [32]–[40]. According to Malkov et al., HNSW improves search complexity by using zoom-in and zoom-out navigation levels. In HNSW each layer has links of a certain scale (inspired by probabilistic skip list structure [41]).

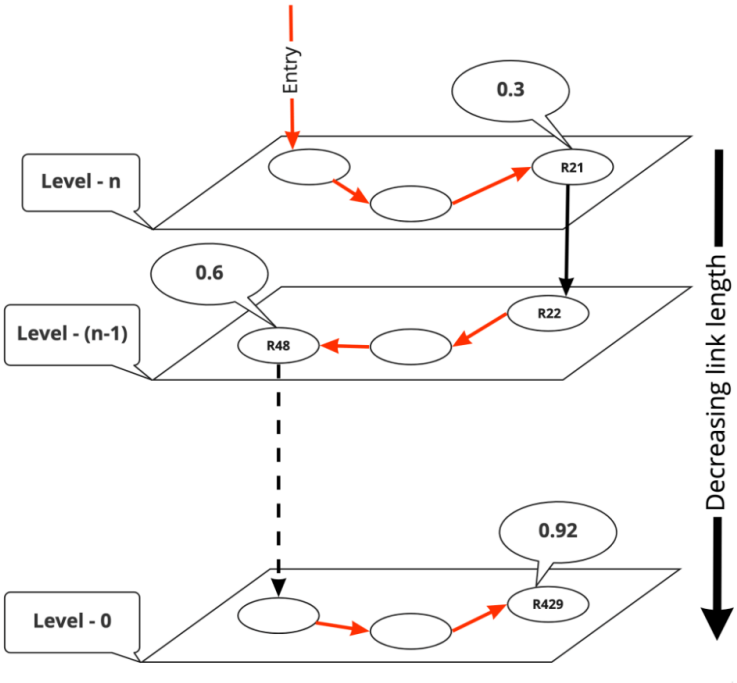

Figure 10: Hierarchical Navigable Small Worlds Algorithm

Fig. 10 illustrates the search process. The query embedding greedily navigates to “R21” in the top layer, then from there to “R48” in the second layer, until it reaches the most similar risk embedding in the zeroth layer, i.e.; “R429” with the score “0.92”.

4 Implementation

4.1 Baselines

4.1.1 Meilisearch

For the integration of the Meilisearch document store into our task, initially we launched the Meilisearch in development mode with a predefined master key. After establishing a connection with the server and creating an index, documents are created using id, reference text, and the corresponding list of risk ids given training data. For the prediction of relevant risks, the target reference is searched through the index with a limit of 4000 documents, and documents that include lexically similar references are ranked with Meilisearch's built-in ranking rules. Lists of risks from these similar references are ranked accordingly. There were two issues with this approach. Meilisearch could not return all risks (or documents). To overcome this problem, scores for unmatched risks are set to zero. Secondly, returning the internal score for matches is still not implemented. Thus, relevancy scores for matched risks are given between 0 and 1 depending on the ranking of the document. In this approach, risks that are in the same list for a specific reference have the same relevancy score. Finally, top-K risks are chosen from the ranking of risks for the prediction.

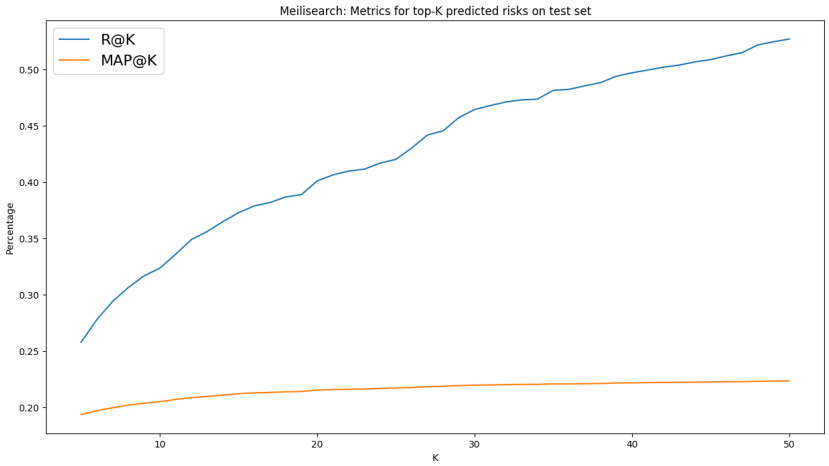

Figure 11: Metrics for reference to reference similarity with Meilisearch on the test set

4.1.2 Cosine similarity with TF-IDF

To represent references and risks as a vector, we fitted and transformed reference texts and risk descriptions to a vector of token counts using TfidfVectorizer. Secondly, using the TF-IDF feature matrix, we applied a transformation to the examples from the test set. Then, we computed cosine similarity for each reference and risk description pair in the test set. Finally, we took K risks with the highest cosine similarity score and returned them as our prediction. The results for this approach are given below:

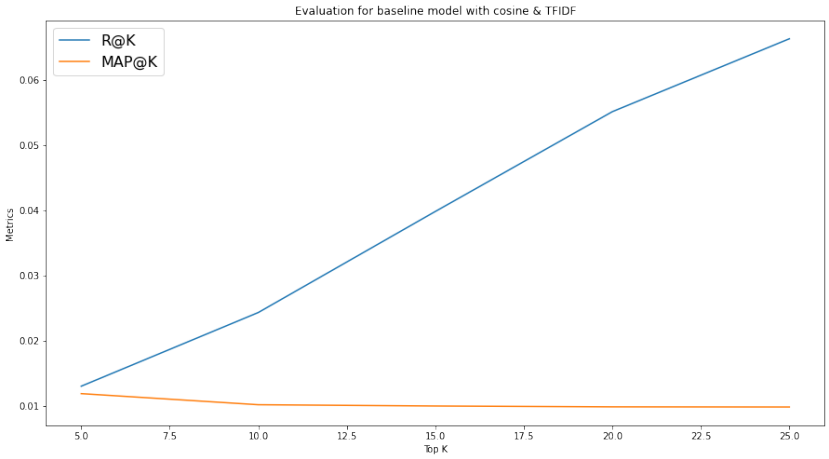

Figure 12: Metrics for cosine similarity with TF-IDF on the test set

4.2 Sentence Transformer

First, we chose several pretrained sentence-transformer MPNet models such as "all-mpnet-base-v1", "paraphrase-mpnet-base-v2", "multi-qa-mpnet-base-cos-v1", "all-mpnet-base-v2" and then transformed each reference text and risk description to 768-dimensional dense vectors. For each reference and risk pair, we computed mean embeddings from each of our previously chosen models. We calculated cosine similarity between reference embeddings and risk description embeddings and sorted them by their score. Finally, we selected top-K risks as our prediction.

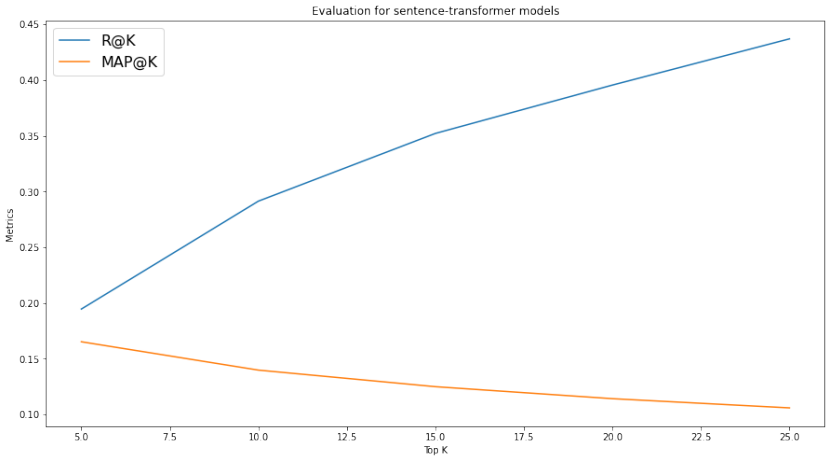

Figure 13: Metrics for mean sentence-transformer embedding on the whole data

4.3 Natural Language Inference

For the adaptation of natural language models to the risk mapping task, we use reference text as the premise and risk description as the hypothesis. If correct risk description is used as a hypothesis, the model should return a high entailment score, otherwise, it should return a low entailment score. You can see the simplified model architecture below:

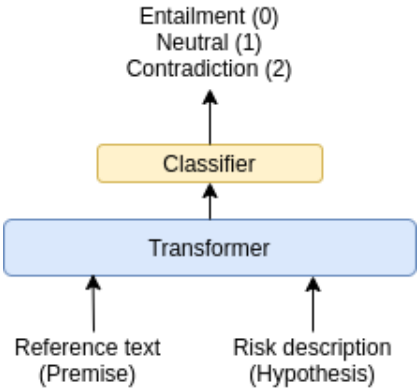

Figure 14: Risk mapping as natural language inference problem

ELECTRA [42] model which was trained on MNLI benchmark is used for the inference without additional training. For each reference text, entailment scores for all risks are calculated using risk description from the risk register. Risks are sorted by entailment scores and top-K risks with high entailment scores are returned as the prediction of the model.

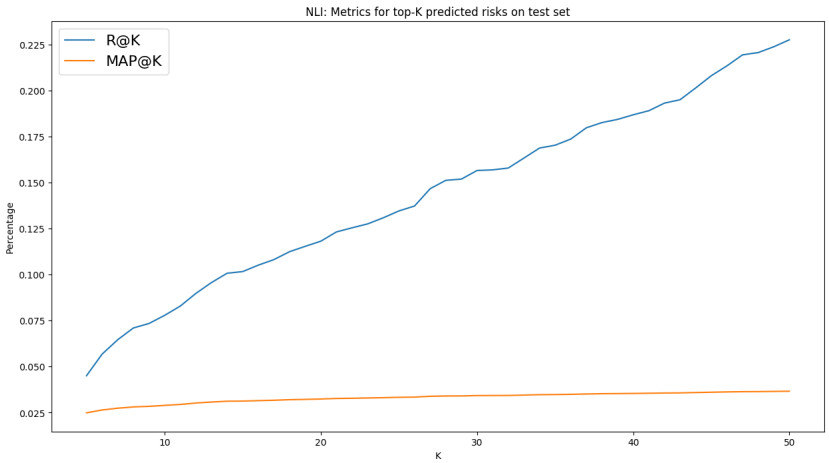

Figure 15: Metrics for natural language inference on the test set

4.4 Mean Reference Embeddings

In this approach, we used 'legal-bert-base-uncased' which is a BERT model previously trained on legal data such as contracts and legislations. We created reference embeddings from the train set and references are grouped by risks. To create our risk embeddings, we compute mean reference embeddings for each risk. Finally, we computed cosine similarity scores between reference embeddings in the test set, and risk embeddings (mean reference embeddings) and rank them accordingly. Finally, we choose the highest top-K risks as our prediction.

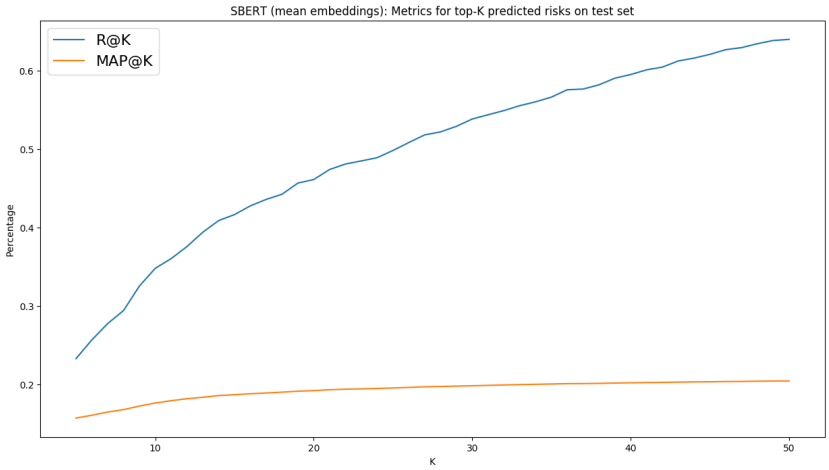

Figure 16: Metrics for mean reference embeddings on test set

We also checked the performance of this approach by varying the number of references used to encode the risk.

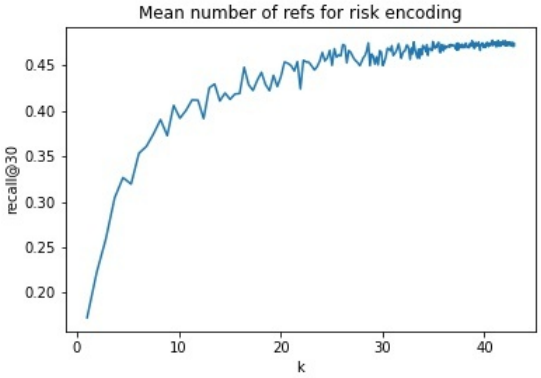

Figure 17: Mean number of references for risk encoding

From the plot, we can see that for the value of k greater than 25 the performance of this approach converges to R@K = 47%.

4.5 Contrastive Learning

For the contrastive learning approach, the supervised SimCSE model which uses RoBERTa-large model (24 layers, 1024 hidden representation, 16 heads) and is trained on NLI datasets (SNLI, MNLI) is chosen due to its performance in semantic textual similarity tasks.

Creating NLI dataset To finetune the model, NLI datasets which include target, positive and negative samples are required. Reference texts are chosen for the target samples, positive samples are chosen from the risk descriptions of true risks. For finding hard negatives, cosine similarity between reference text and all risk descriptions are computed and ranked using the pretrained SimCSE model. The risk descriptions which have high cosine

similarity for that specific reference and not true risks are chosen as hard negatives for the NLI dataset. The number of hard negatives is chosen as five for references that have less than five risks and set to the number of true risks when the number of true risks for the reference is more than five.

Training Model is trained for 3 epochs with a batch size of 32 using pretrained RoBERTa model and NLI dataset. The learning rate is set to 5e-5 and the temperature coefficient is 0.05.

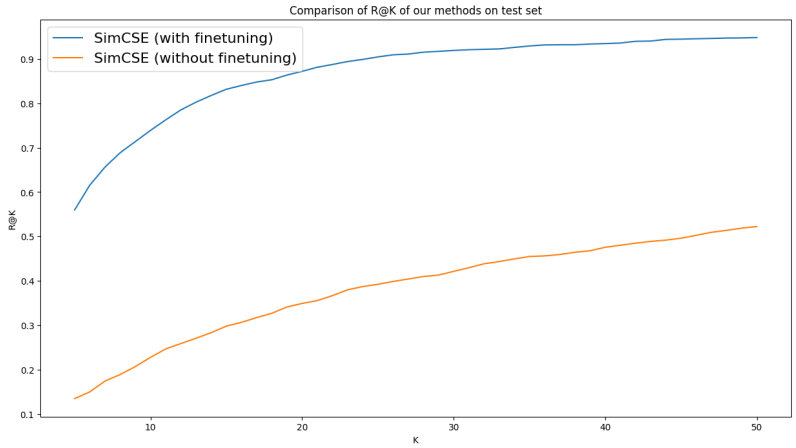

Figure 18: Comparison of recall on the test set with and without finetuning SimCSE

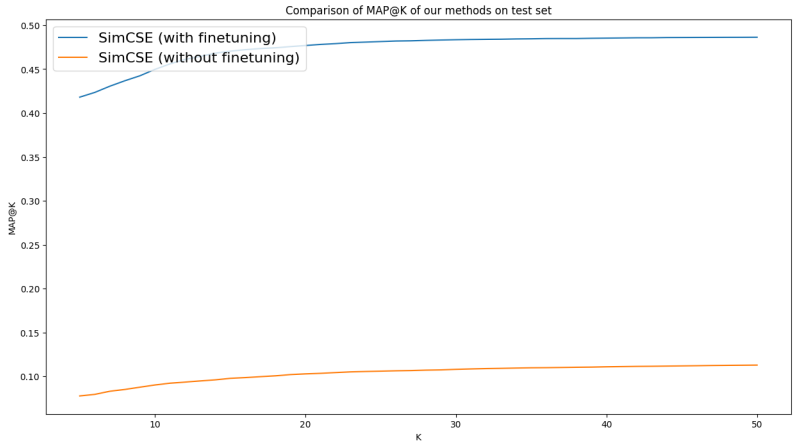

Figure 19: Comparison of mean average precision on the test set with and without finetuning SimCSE

4.6 Qdrant Neural Search

Qdrant neural search is a vector similarity search engine based on hierarchical navigable small world indexing graph. Neural search finds applications in areas where queries cannot be formulated precisely. Qdrant works by generating a neural database using vector data and payload data.

We represent each risk as the mean embedding of all of the references in the train set that it maps to. To vectorize the reference text, we use two approaches, namely the MPNetv2 sentence transformer [43] and a FastText-based model pretrained on GDPR text.

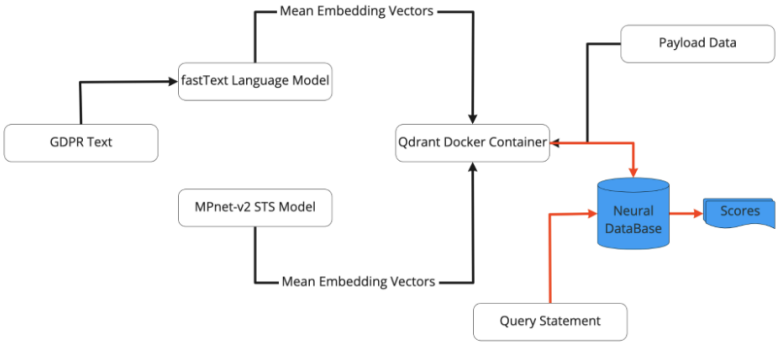

Figure 20: Qdrant Neural Search Engine Components

In the above diagram, "Payload Data" refers to the risk attributes and vector embeddings, which are used to generate a vector index. The vector index helps to query several vectors similar to the target vector.In Fig. 21 the x-axis depicts the top-K predictions that we consider for each of the 500 reference policy statements from the test set. These predictions are compared against the true set of risks associated with the test set reference statements. In this plot, the red curve represents the MPNetv2 model and the blue represents the GDPR-based model. Examining the plot, it is clear that the MPNetv2 model performs better for increasing values of top-K predicted risks. Therefore, we can conclude that a general sentence textual similarity (STS) model is more suited to working with reference and risk statements that can have large variations in their vocabulary and themes. The GDPR-based model is specific to data protection policies, hence it is unable to generalize well enough to different types of risks. For this reason, the reference-to-risk similarity scores obtained from the MPNetv2 model were used in the LightGBM method.

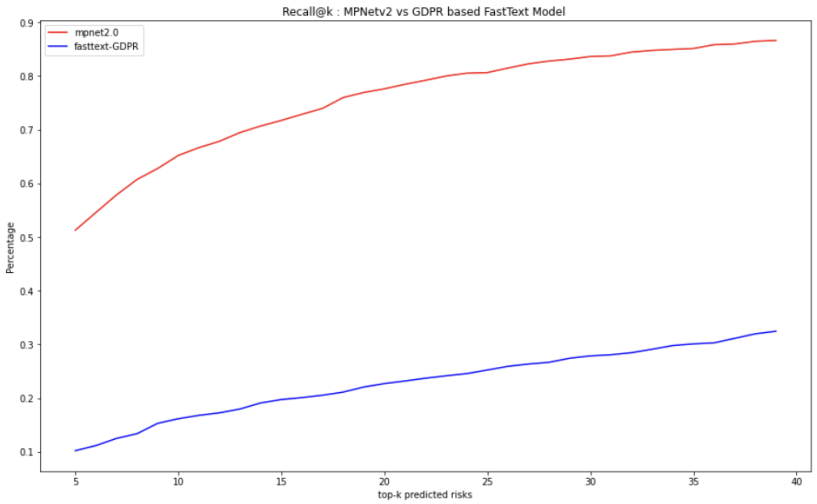

Figure 21: MPNetv2 vs. FastText GDPR for R@K

5 Results

5.1 LightGBM

In the previous sections, we described our approaches and their results. As a final step, we decided to rank the scores of our models with the help of the LightGBM ranker and get the final ranking. LightGBM is a framework developed by Microsoft which is based on the ensemble of decision trees, which are trained in sequence using the gradient boosting method. In addition to classification and regression tasks, it is also able to do ranking. During ranking, we used LambdaRank as our objective function and as an evaluation metric, we used normalized discounted cumulative gain (NCDG).

$$DCG@k = \sum_{n=1}^{k} \frac{relevance_i}{log_2(i+1)}$$

where $relevance_i$ is the relevance score of risk i.

$$IDCG@k = \sum_{n=1}^{K_{ideal}} \frac{relevance_i^{ideal}}{log_2(i+1)}$$

where $relevance_i^{ideal}$ is the relevance score of risk i. Ideal DCG score - score when we recommend the most relevant items first.

$$NDCG@k = \frac{DCG@K}{IDCG@K}$$

We trained LightGBM ranker on scores of our models from validation set, finetuned on scores of our models from validation set, and finally checked performance on the test set.

5.2 Error Analysis

We first consider the risks that are most often wrongly predicted for a given reference. Across all our approaches we compiled a list of the top 50 most wrongly predicted risks at R@30. There are 116 such risks, and 8 amongst them are common across all methods.

	Risk-Scores	Intersection-List Scores	Union-List Scores
Count	554	8	116
Mean	38.9%	48.6%	45.5%
Std	6.7%	1.2%	3.9%
Min	10.4%	46.3%	31.8%
25%	34.2%	48%	44.3%
50%	40.1%	48.7%	46.4%
75%	44.3%	49.3%	47.9%
Max	52.9%	50.2%	52.9%

We observe that the top 75% of risks in the union list lie in the top 25% of all risks by similarity score. Based on this we can ascertain that 50R@30 risks share a high degree of semantic similarity across most references. This suggests that there is some deeper

sense of similarity between references and risks that is overlooked by our approaches. We hypothesize that a lack of domain knowledge contributes to these wrong predictions. To explore this hypothesis we consider a subset of 29 risks that are unique to the Qdrant approach. We represent these risks using the MPNetv2 and FastText GDPR model and store how often they are correctly and wrongly predicted across all references.

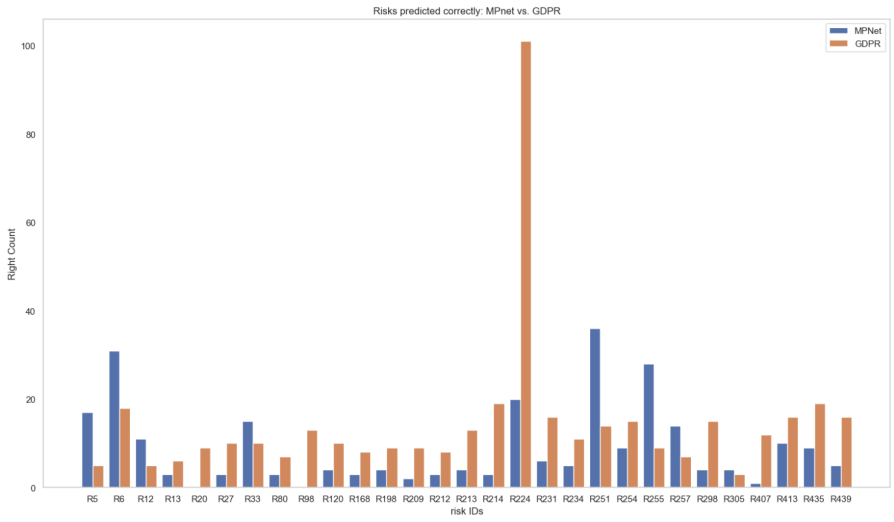

Figure 22: Number of correct predictions: MPNetv2 vs FastText GDPR

In fig. 22, we observe that 21 of the most wrongly predicted risks are predicted correctly more often by the FastText GDPR model, with 15 risks containing terms commonly found in GDPR. Risk R224 contains many such terms and is predicted correctly 4 times more often by the GDPR model. This trend is reversed in fig. 23.

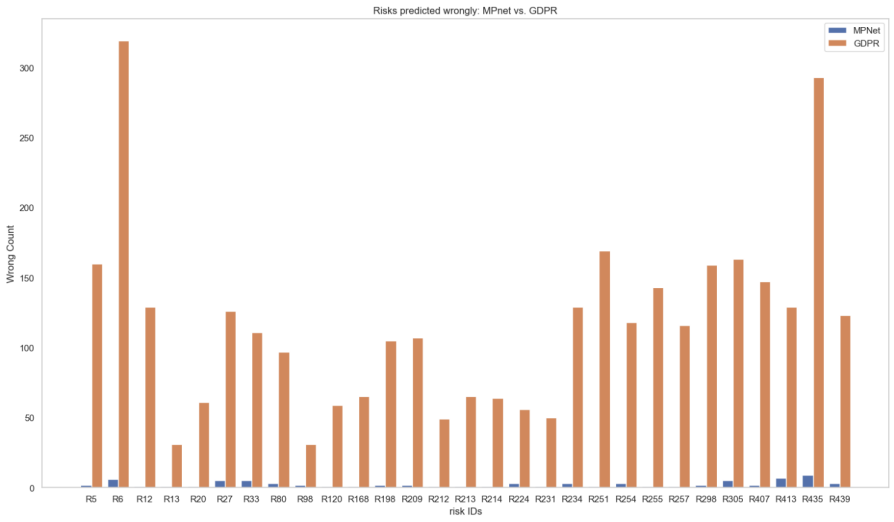

Figure 23: Number of wrong predictions: MPNetv2 vs. FastText GDPR

We reason that a model with some domain knowledge can correctly predict some risks, while its specificity limits it from predicting others, resulting in many wrong predictions.

5.2.1 Quantitative Analysis

	Recall			Mean Average Precision		
	R@10	R@20	R@30	MAP@10	MAP@20	MAP@30
Meilisearch	32.36	40.10	46.43	20.48	21.51	21.95
Natural Language Inference	7.78	11.81	15.65	2.89	3.24	3.43
Mean Reference Embeddings	34.78	46.11	53.83	17.59	19.17	19.79
Qdrant	65.18	77.53	83.57	43.80	45.84	46.50
SimCSE	73.91	87.22	91.93	44.96	47.68	48.33
LightGBM (ndcg)	77.72	90.02	93.61	52.19	54.74	55.31
LightGBM (map)	**81.32**	**91.71**	**94.53**	**57.11**	**58.59**	**59.72**

Table 3: Comparison of metrics of our methods on test set

Our best performing model was SimCSE which acquired recall of 91.93 for top-30 prediction since it is finetuned on reference dataset and contrastive learning approaches work well with small amount of data. Following SimCSE, the Qdrant model achieved an R@30 score of 83.57. The natural language inference model was the worst performing model compared to other models we used since the model is used without additional training and lacks domain knowledge. We compared the performance of LightGBM to other models using R@30. We can see that LightGBM improved the result of our best model (SimCSE) by ranking predictions of all our models. Comparison of all our models has given below:

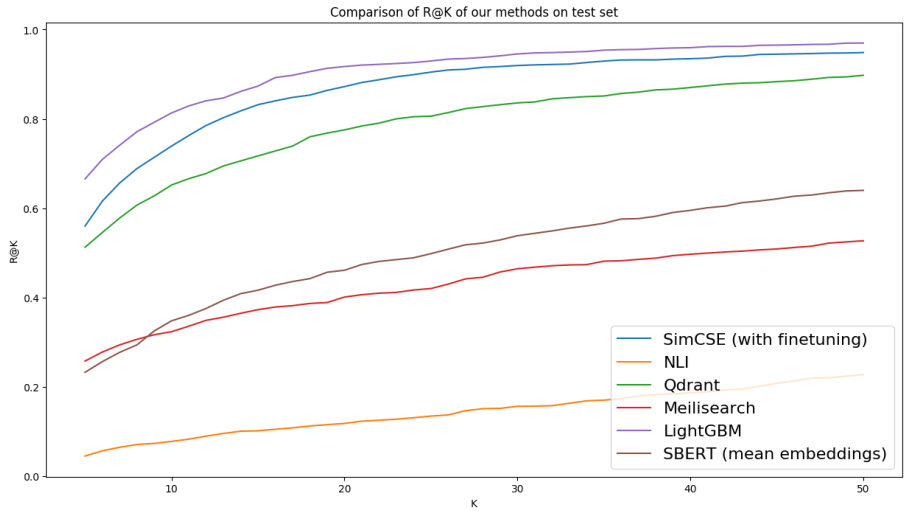

Figure 24: Comparison of our models in terms of recall on the test set

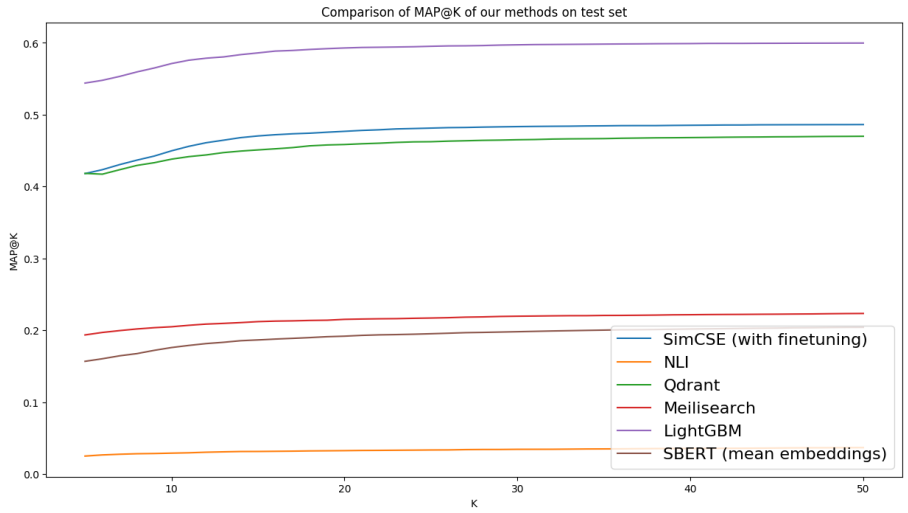

Figure 25: Comparison of our models in terms of mean average precision on the test set

5.2.2 Qualitative Analysis

For the qualitative analysis, we can investigate several reference examples and top-K predictions from each model.

Reference: "7. to ensure that personal data are protected from accidental destruction or loss (availability control),"

SimCSE	Meilisearch	NLI	Qdrant	Mean Embeddings
R198	R79	R167	R55	R55
R55	R76	R338	R54	R271
R52	R7	R583	R33	R357
R54	R6	R29	R193	R193
R190	R55	R680	R197	R9

Table 4: Top-5 prediction for different models on reference example #1. The green background represents that risk is in the true list of risks. The red background represents that risk is not one of the true risks.

In the first example, R55 ('physical data being destroyed by natural influences.') is successfully recognized by SimCSE, Meilisearch, Qdrant and Mean Embeddings approach. Only the natural language inference model fails to predict R55. Qdrant manages to include R33 ('data retention and deletion requirements not being met.') in its top-K prediction, unlike other models. Furthermore, Meilisearch can predict R79 ('continuity measures being defined for non-critical functions.') where other models fail to identify it. Meilisearch finds lexical matches using words such as *personal*, *protect*, *ensure*, *data*, and *accidental*, returns similar references which contain these words and use their corresponding risks (including R79) as its prediction.

Figure 26: Similarity matrix between reference example #1 and risk description (R55) using SimCSE

On the other hand, SimCSE model captures the relation between the reference text and risk description of R55 with words such as *data*, *destroyed* (destruction, loss), *natural* (accidental).

Reference: "'Art. 5 Principles relating to processing of personal data1. Personal data shall be:(a) processed lawfully ... be able to demonstrate compliance with, paragraph 1 ('accountability')."'

SimCSE	Meilisearch	NLI	Qdrant	Mean Embeddings
R223	R79	R222	R226	R221
R224	R76	R250	R223	R226
R225	R7	R194	R225	R225
R11	R6	R404	R239	R359
R222	R55	R226	R899	R220

Table 5: Top-5 prediction for different models on reference example #2. The green background represents that risk is in the true list of risks. The red background represents that risk is not one of the true risks.

In the second example, although SimCSE can find the risks R222 ('inaccurate personally identifiable information being stored and processed.'), R223 ('unnecessary data storage and archiving cost.'), R224 ('inappropriate use of personally identifiable information.'), R225 ('excessive collection of personally identifiable information.'), it fails to predict R7 ('the organization violating data privacy legal requirements.') or R226 ('data being used for purposes other than the purpose for collection.') in its top-5 prediction. However, NLI, Qdrant and Mean Embeddings model include R226 in their top-5 predictions and Meilisearch manages to find R7 in its prediction.

6 Conclusion

For this project, we worked on the problem of risk identification which results from unfulfilled legal requirements. We have investigated the relationship between policy statements and risks from multiple points of view taking into consideration semantic, lexical, and logical relations. We explored several approaches such as cosine similarity with sentence transformers, contrastive learning, Meilisearch, natural language inference, recommendation systems, neural search engine, and mean reference embeddings approach. We chose our Meilisearch and cosine similarity with TF-IDF vectors as our baselines. SimCSE which is a contrastive learning framework achieved the best performance among all of our models due to additional training on the reference dataset and the convenience of working with a small amount of data. Finally, we combined the relevancy scores of our models and gave these scores to the LightGBM ranker as inputs to re-rank the risks. As a result, this learning-to-rank approach has greatly improved final metrics including recall and mean average precision. In conclusion, we achieved our goal to improve the results from baseline and explored numerous approaches to tackle the problem of risk identification.

In the future, the natural language inference model should be fine-tuned on the reference dataset using larger models which would improve the metrics. Data split should be reconsidered. It should be handled such that each train, validation, and test set include examples from all risks, and examples should be distributed more appropriately. For hyperparameter tuning, the SimCSE model should be experimented with a different number of hard negatives and different temperature coefficients. LightGBM parameters such as maximum tree depth and the number of leaves should be tuned to improve the quality of the final ranking. Relevancy scores from Meilisearch should be received from the internal scoring mechanism when this feature is implemented. Furthermore, in error analysis, we observed that domain knowledge improves the total number of correct predictions, hence we believe this should be incorporated into future approaches as well.

References

[1] Veritas, *GDPR Infographic*, https://uk.insight.com/content/dam/insight/EMEA/blog/2017/06/GDPR-Infographic-design-final.pdf, Accessed: 2022-07-23.

[2] J. Kahn, S. Bodoni, and S. Nicola, *It'll cost billions for companies to comply with europeâs new data law*, https://www.bloomberg.com/news/articles/2018-03-22/it-ll-cost-billions-for-companies-to-comply-with-europe-s-new-data-law, Accessed: 2022-07-23.

[3] C. Howe and B. Benton, *Internal controls and the shifting wave of focus*, https://www.alyne.com/de/blog/articles/internal-controls-and-the-shifting-wave-of-focus/, Accessed: 2022-07-23.

[4] B. Benton, *Real-time operation risk management in financial institutions*, https://www.alyne.com/de/blog/articles/real-time-operational-risk-management-in-financial-institutions-part-1/, Accessed: 2202-07-23.

[5] M. Mohammadnia Qaraei, S. Khandagale, and R. Babbar, "Why state-of-the-art deep learning barely works as good as a linear classifier in extreme multi-label text classification," English, ser. ESANN 2020 - Proceedings, 28th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning; 2020, p. 6, ISBN: 9782875870742. [Online]. Available: http://urn.fi/URN:NBN:fi:aalto-2021120110556.

[6] R. Zhang, Y.-S. Wang, Y. Yang, D. Yu, T. Vu, and L. Lei, "Long-tailed extreme multi-label text classification with generated pseudo label descriptions," *ArXiv*, vol. abs/2204.00958, 2022.

[7] A. Mittal, K. Dahiya, S. Agrawal, *et al.*, "Decaf: Deep extreme classification with label features," *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021.

[8] T. Gao, X. Yao, and D. Chen, *Simcse: Simple contrastive learning of sentence embeddings*, 2021. DOI: 10.48550/ARXIV.2104.08821. [Online]. Available: https://arxiv.org/abs/2104.08821.

[9] Elastic, *What is Elasticsearch?* https://www.elastic.co/what-is/elasticsearch.

[10] *Meilisearch*, https://www.meilisearch.com.

[11] *Typesense: Fast, typo-tolerant, open source search engine*, https://www.typesense.org.

[12] *How it works?* https://www.algolia.com/products/search-and-discovery/hosted-search-api/.

[13] *Redisearch: Full-text search engine for nosql database*, https://redis.com/modules/redis-search/.

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2018. DOI: 10.48550/ARXIV.1810.04805. [Online]. Available: https://arxiv.org/abs/1810.04805.

[15] Y. Liu, M. Ott, N. Goyal, *et al.*, *Roberta: A robustly optimized bert pretraining approach*, 2019. DOI: 10.48550/ARXIV.1907.11692. [Online]. Available: https://arxiv.org/abs/1907.11692.

[16] E. Charniak, Y. Altun, R. de Salvo Braz, *et al.*, "Reading comprehension programs in a statistical-language-processing class," in *ANLP-NAACL 2000 Workshop: Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, 2000. [Online]. Available: https://aclanthology.org/W00-0601.

[17] H. T. Ng, L. H. Teo, and J. L. P. Kwan, "A machine learning approach to answering questions for reading comprehension tests," in *2000 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, Hong Kong, China: Association for Computational Linguistics, Oct. 2000, pp. 124–132. DOI: 10.3115/1117794.1117810. [Online]. Available: https://aclanthology.org/W00-1316.

[18] O. Glickman, *Applied textual entailment*. Citeseer, 2006.

[19] I. Dagan, O. Glickman, and B. Magnini, "The pascal recognising textual entailment challenge," in *Machine learning challenges workshop*, Springer, 2005, pp. 177–190.

[20] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[21] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *Advances in neural information processing systems*, vol. 26, 2013.

[22] M. E. Peters, M. Neumann, M. Iyyer, *et al.*, *Deep contextualized word representations*, 2018. DOI: 10.48550/ARXIV.1802.05365. [Online]. Available: https://arxiv.org/abs/1802.05365.

[23] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[24] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, *A large annotated corpus for learning natural language inference*, 2015. DOI: 10.48550/ARXIV.1508.05326. [Online]. Available: https://arxiv.org/abs/1508.05326.

[25] A. Williams, N. Nangia, and S. R. Bowman, *A broad-coverage challenge corpus for sentence understanding through inference*, 2017. DOI: 10.48550/ARXIV.1704.05426. [Online]. Available: https://arxiv.org/abs/1704.05426.

[26] T. Khot, A. Sabharwal, and P. Clark, "Scitail: A textual entailment dataset from science question answering," in *AAAI*, 2018.

[27] S. Storks, Q. Gao, and J. Y. Chai, *Recent advances in natural language inference: A survey of benchmarks, resources, and approaches*, 2019. DOI: 10.48550/ARXIV.1904.01172. [Online]. Available: https://arxiv.org/abs/1904.01172.

[28] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, *A survey on contrastive self-supervised learning*, 2020. DOI: 10.48550/ARXIV.2011.00362. [Online]. Available: https://arxiv.org/abs/2011.00362.

[29] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, *A simple framework for contrastive learning of visual representations*, 2020. DOI: 10.48550/ARXIV.2002.05709. [Online]. Available: https://arxiv.org/abs/2002.05709.

[30] Y. A. Malkov and D. A. Yashunin, "Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, pp. 824–836, 2020.

[31] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman, "Indexing by latent semantic analysis," *J. Am. Soc. Inf. Sci.*, vol. 41, pp. 391–407, 1990.

[32] S. Arya and D. M. Mount, "Approximate nearest neighbor queries in fixed dimensions," in *SODA '93*, 1993.

[33] J. Wang, J. Wang, G. Zeng, R. Gan, S. Li, and B. Guo, "Fast neighborhood graph search using cartesian concatenation," *2013 IEEE International Conference on Computer Vision*, pp. 2128–2135, 2013.

[34] J. Wang and S. Li, "Query-driven iterated neighborhood graph search for large scale indexing," *Proceedings of the 20th ACM international conference on Multimedia*, 2012.

[35] Z. Jiang, L. Xie, X. Deng, W. Xu, and J. Wang, "Fast nearest neighbor search in the hamming space," in *MMM*, 2016.

[36] E. Chávez and E. S. Tellez, "Navigating k-nearest neighbor graphs to solve nearest neighbor searches," in *MCPR*, 2010.

[37] K. Aoyama, K. Saito, H. Sawada, and N. Ueda, "Fast approximate similarity search based on degree-reduced neighborhood graphs," in *KDD*, 2011.

[38] G. Ruiz, E. Chávez, M. Graff, and E. S. Tellez, "Finding near neighbors through local search," in *SISAP*, 2015.

[39] R. Paredes, "Graphs for metric space searching," 2008.

[40] Y. Malkov, A. Ponomarenko, A. Logvinov, and V. Krylov, "Scalable distributed algorithm for approximate nearest neighbor search problem in high dimensional general metric spaces," in *SISAP*, 2012.

[41] W. W. Pugh, "Skip lists: A probabilistic alternative to balanced trees," in *CACM*, 1990.

[42] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, "ELECTRA: Pre-training text encoders as discriminators rather than generators," in *ICLR*, 2020. [Online]. Available: https://openreview.net/pdf?id=r1xMH1BtvB.

[43] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, "Mpnet: Masked and permuted pre-training for language understanding," *ArXiv*, vol. abs/2004.09297, 2020.

Appendix

6.1 Recommender System Approach

In this approach we made use of reference-to-reference similarity to predict similarity scores for various risks in the registry. The approach of Collaborative Filtering via matrix completion was pursued using the Singular Value Decomposition (SVD) algorithm. The Surprise Recommender Systems package was used to implement the algorithm. The above

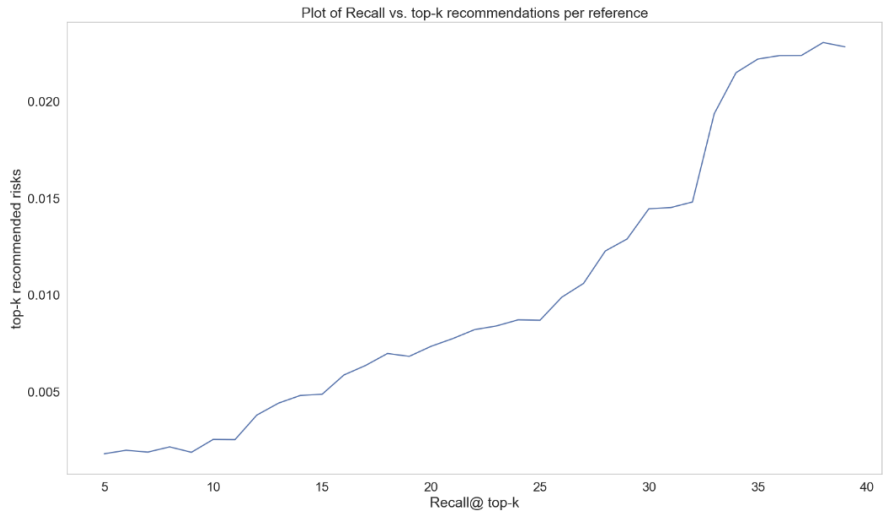

Figure 27: Recall@k: SVD-based Collaborative Filtering

plot demonstrates peculiarly low R@K performance. In examining the reason for this poor performance, we discovered that the model does not predict a consistent number of risks per reference as can be observed in the Fig. 28. We note that the model does not take into

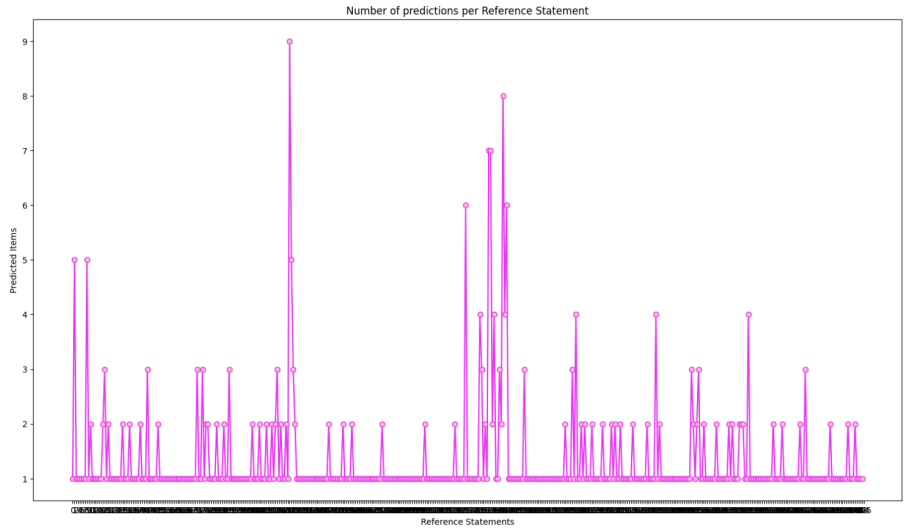

Figure 28: Number of Predictions per Reference Statement

consideration the risk attributes and only operates on a matrix of reference-risk similarity

scores. Moreover, SVD-based matrix completion must fill a matrix of 3259 references by 554 risks, i.e.; 2 million entries while only using scores from some 30k known reference and risk pairs. Due to this extremely sparse starting matrix coupled with some very high similarity scores associated with vectors prepared with the GDPR-based FastText model the resultant scores are an interpolation between a narrow range of values as shown in fig. 29.

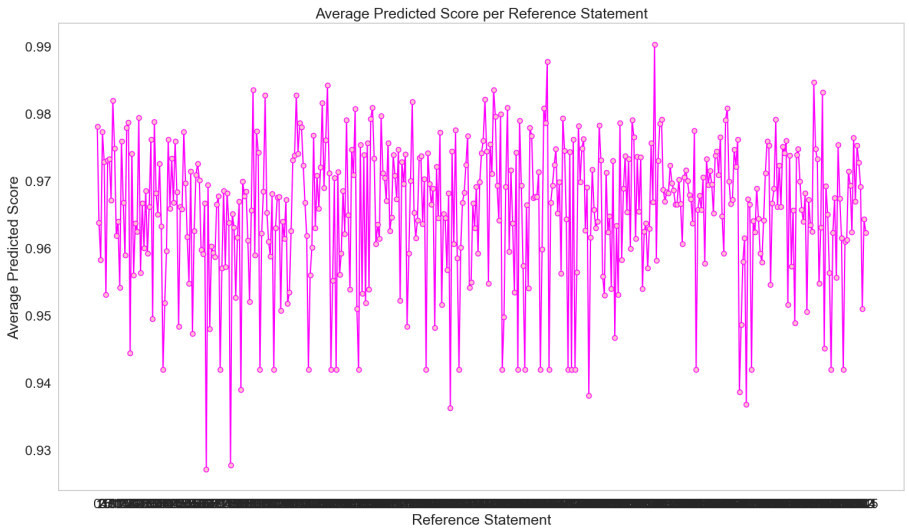

Figure 29: Average Predicted Score per Reference Statement

As most reference-risk pairs receive a very high score regardless of their actual semantic meaning the resultant model fails to produce any valuable results. These limitations of the approach as well as the limited data not allowing for a more populated sparse matrix are the reasons we do not include results from this approach.

6.2 Multilabel classification with basic machine learning algorithms

In this approach, we encoded references using TF-IDF and N-grams and applied basic machine learning algorithms (Decision Trees, Random Forest, Boosting, and SVM) for the task of multilabel classification. The best result was achieved by SVM model. However, this result was below the result of the baseline Meilisearch, therefore we decided not to include results from this approach.

	Multi-Label Models	Macro-Precision	Macro-Recall
1	KNN Classifier	20.8%	16.1%
2	Decision Tree Classifier	25.6%	24.2% %
3	Bagging Classifier	30.6%	14.7%%
4	Random Forest Classifier	26.2%	7.7% %
5	Gradient Boosting Classifier	31.1%	22.7% %
6	Naive Bayes Classifier	0.0%	0.0% %
7	Support Vector Machine Classifier	36.0%	18.0%%

6.3 Additional error analysis

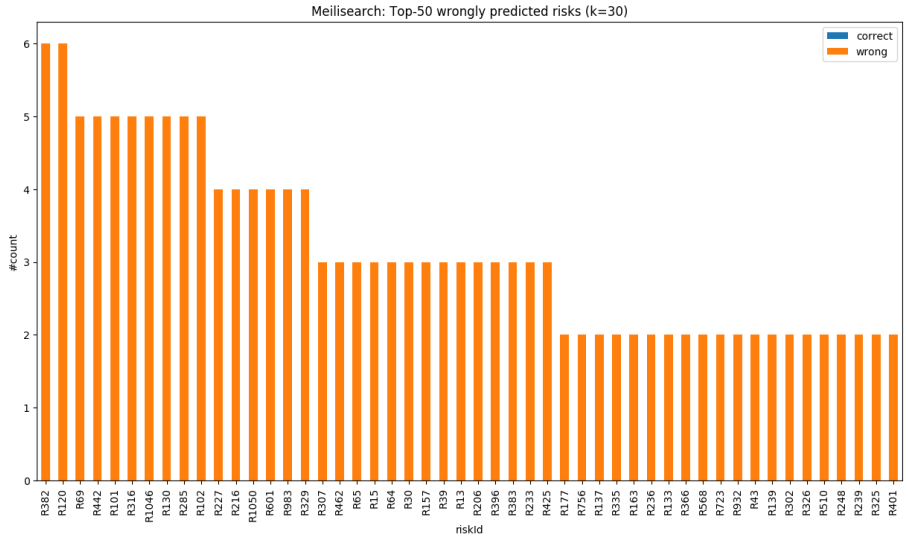

Figure 30: Meilisearch: Top-50 failed risks using Meilisearch on the test set

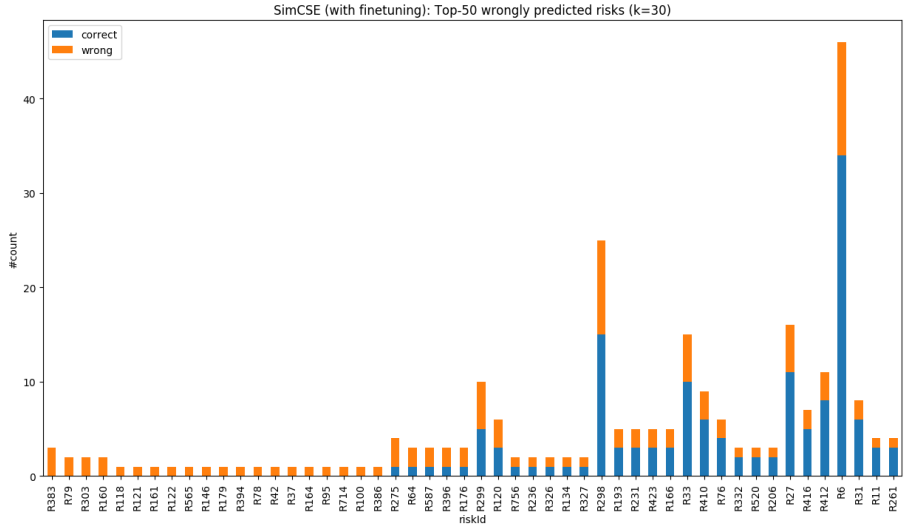

Figure 31: SimCSE: Top-50 failed risks using SimCSE on the test set

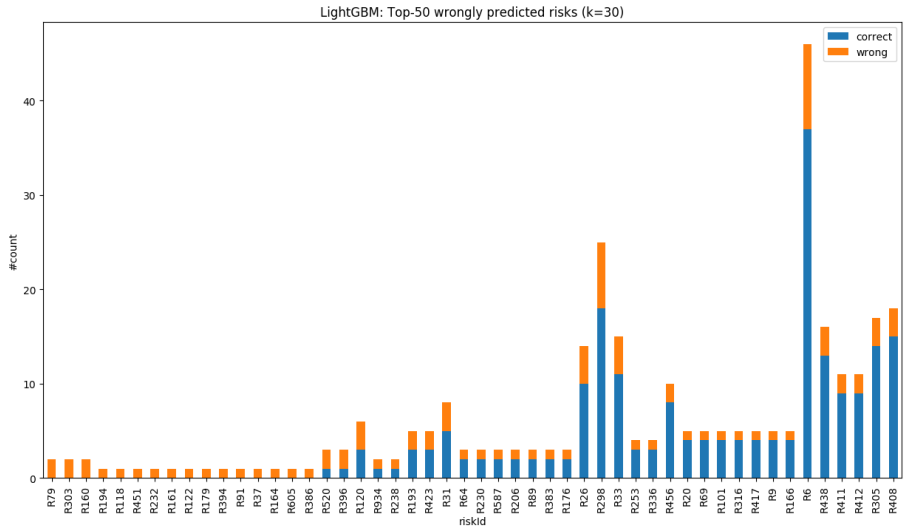

Figure 32: LightGBM: Top-50 failed risks using LightGBM on the test set

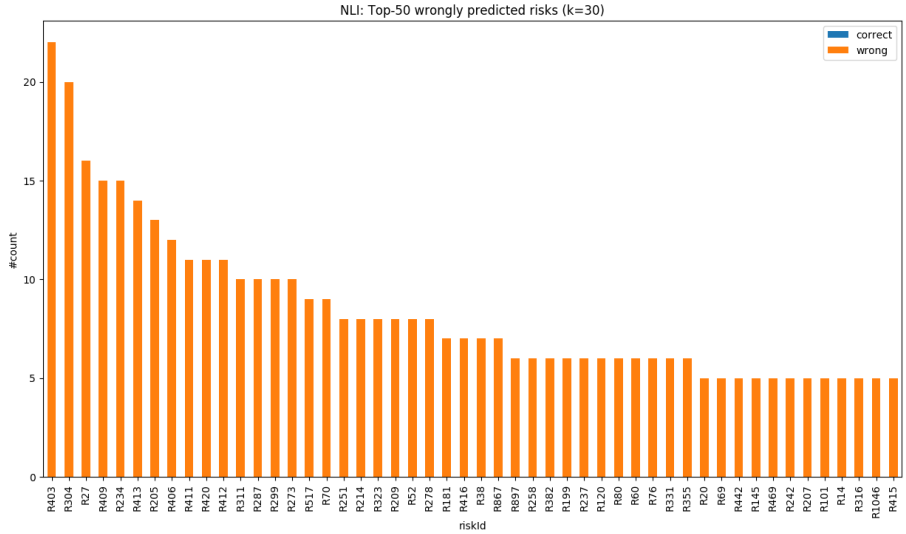

Figure 33: Natural Language Inference: Top-50 failed risks using natural language inference on the test set

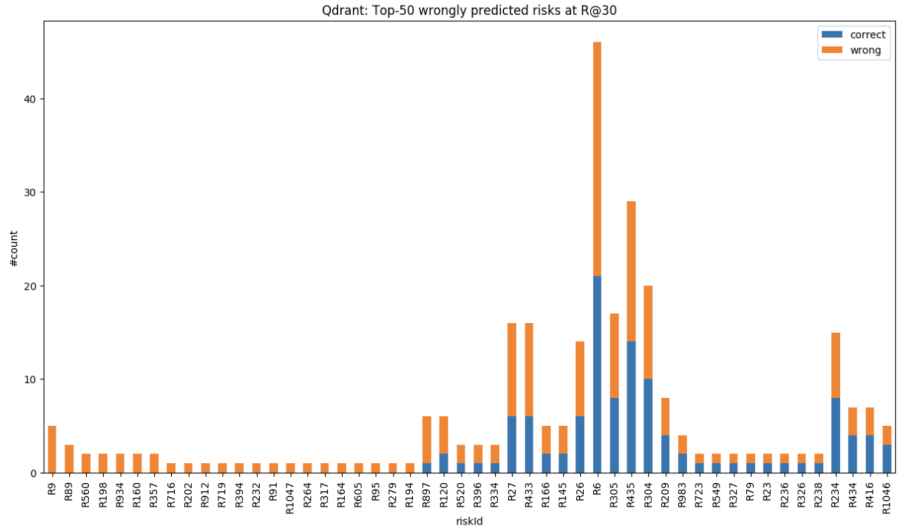

Figure 34: Qdrant Neural Search: Top-50 failed risks using qdrant neural search on the test set

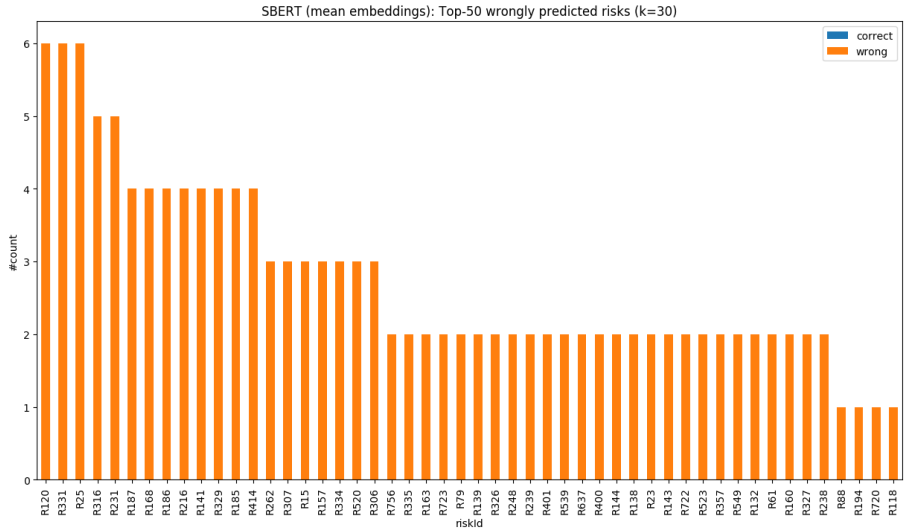

Figure 35: Mean Reference Embeddings: Top-50 failed risks using mean reference embeddings on the test set