CICLe: Conformal In-Context Learning for Largescale Multi-Class Food Risk Classification

Korbinian Randl,[1] **John Pavlopoulos,**[1,2,3] **Aron Henriksson,**[1] **Tony Lindgren**[1]

[1] Stockholm University, Borgarfjordsgatan 12, 164 07 Kista, Sweden
{korbinian.randl,ioannis,aronhen,tony}@dsv.su.se
[2]Athens University of Economics and Business, Patission 76, Athens 104 34, Greece
[3]Archimedes/Athena RC

Abstract

Contaminated or adulterated food poses a substantial risk to human health. Given sets of labeled web texts for training, Machine Learning and Natural Language Processing can be applied to automatically detect such risks. We publish a dataset of 7,546 short texts describing public food recall announcements. Each text is manually labeled, on two granularity levels (coarse and fine), for food products and hazards that the recall corresponds to. We describe the dataset and benchmark naive, traditional, and Transformer models. Based on our analysis, Logistic Regression with a tf-idf representation outperforms RoBERTa and XLM-R on classes with low support. Finally, we discuss different prompting strategies and present an LLM-in-the-loop framework, based on Conformal Prediction, which boosts the performance of the base classifier while reducing energy consumption compared to normal prompting.

1 Introduction

Food-bourne illnesses and contaminated food pose a serious threat to human health and lead to thousands of deaths (Majowicz et al., 2014; de Noordhout et al., 2014). Natural Language Processing (NLP) solutions based on Machine and Deep Learning (ML, DL) or Large Language Models (LLMs) enable fast responses to new threats by generating warnings from publicly available texts on the internet (Maharana et al., 2019; Tao et al., 2023). These texts, however, as we show, can be noisy and are characterized by thousands of classes. Categorization of incidents into these classes is needed to properly address food risks, as different food risks require different countermeasures with different urgency. For example, while a fraudulent import of foodstuffs usually imposes a very low risk to consumer health, Listeria or Salmonella contamination is a serious risk to consumer health and can even lead to death. Automatic classification of the hazard involved in food recalls can help

estimate the severity of food risks and lead to faster and more targeted measures. While we are aware of existing text datasets on the topic of food-bourne illnesses (Hu et al., 2022), they focus only on the detection of such illnesses and not on their categorization. Publicly available official food recall announcements, and specifically their shorter titles, contain detailed information on food risks and the specific products involved and may be used as a basis for data-driven ML approaches. Acknowledging this research gap, this paper is specifically addressing the following research questions:

R1: Are titles of food recall articles sufficient for classification of the food hazards and food products involved?

- **R1.1:** Do the titles of official announcements of food hazards comprise enough information for their classification?
- **R1.2:** What is the performance of ML and DL classification models trained on the presented dataset?

As LLMs like GPT-4, and Llama (Touvron et al., 2023) represent the current state-of-the-art in NLP, but are also known for their high resource consumption, we further explore the following question:

R2: Can LLMs be used to address or improve automatic food hazard classification?

- **R2.1:** What is the performance of LLMs for the classification tasks?
- **R2.2:** How can ML and DL be used to increase resource efficiency without harming the accuracy of LLMs for the task?

In this work, we address the above task of classifying food hazards and the involved products from food recall texts and provide the following main contributions:

1. We present the first dataset for text classification of food products and food hazards on two levels of granularity.[1]

---

[1] DOI:10.5281/zenodo.10820657

2. We present a benchmark on the introduced dataset using naive, traditional ML, and Transformer classifiers, showing that a Support Vector Machine outperforms the rest due to better performance on low-support classes.

3. We discuss different approaches to few-shot prompting with GPT-3.5 on our data and propose reducing the number of relevant classes for few-shot prompting using Conformal Prediction (CP). We show that this enables resource-friendly prompting at comparably high F_1 by boosting a base classifier's performance in a LLM-in-the-loop framework.

2 Background and Related Work

Research on ML with food data has traditionally focused on tabular and image data (Zhou et al., 2019; Jin et al., 2020; Wang et al., 2022), where food risk classification is usually presented as a detection task: rather than identifying specific hazards, the task is formulated as the binary classification of the presence or absence of an incident (Maharana et al., 2019; Tao et al., 2023; Wang et al., 2023). Focusing on the text modality, we found literature exploring such detection tasks based on Amazon reviews matched with FDA food recall announcements (Maharana et al., 2019), as well as Twitter data, labeled using instances from the US National Outbreak Reporting System (Tao et al., 2023). In both of these papers, the data is highly unbalanced towards the negative class (i.e. no incident). Furthermore, the description of the method for labeling the Twitter texts in Tao et al. (2023) lacks important details, as it is unclear on what grounds texts and events are matched. We only found one study on the actual classification of hazards: Xiong et al. (2023) which uses a hierarchical Transformer model to classify Chinese news texts into four categories of hazards. The authors do not share further details on their data, which makes an assessment of the paper's validity hard. Overall, the field of food risk classification based on text lacks reproducible work on fine-grained prediction of hazards based on openly accessible data.

ML models based on the attention mechanism by Vaswani et al. (2017) and the self-supervised masked language modeling pre-training paradigm generally outperform traditional methods both on monolingual (Devlin et al., 2019; Liu et al., 2019) and multilingual (Conneau et al., 2020) data. Although such Transformers were origi-

nally described as an encoder-decoder architecture (Vaswani et al., 2017), mapping an input-text to an output-text, models intended for classification tasks usually only employ an encoder (Devlin et al., 2019; Liu et al., 2019; Conneau et al., 2020), followed by several task-specific layers.

Recently, LLMs such as Llama 2 (Touvron et al., 2023), PaLM (Chowdhery et al., 2022), and GPT-family models (Brown et al., 2020) have been shown to exceed the capabilities of smaller Transformers even without further fine-tuning. That is, by simply providing a context of a few labeled samples per class, LLMs can predict the classes of unseen samples within this context. As LLMs are usually text-to-text Transformers, the context is provided directly in each prompt. This paradigm is commonly referred to as *few-shot prompting* or *in-context learning* (Brown et al., 2020). Since the discovery of the few-shot capabilities of LLMs, newer LLMs are often designed for high few-shot performance (Gao et al., 2021; Chowdhery et al., 2022).

Current work on few-shot prompting focuses mostly on creating/finding the optimal few-shot samples from the training data. Ahmed et al. (2023) proposed a workflow for automatically finding similar samples from the pool of labeled "training" samples for code summarization. Shi et al. (2023) focused on automatic generation of Chain of Thought (CoT) labels for samples in reasoning tasks. CoT is another prompting paradigm that asks the LLM to provide a chain of reasoning before delivering the prediction and has been shown to drastically improve reasoning performance (Wei et al., 2022). Nevertheless, to the best of our knowledge, there is no previous work on how to leverage few-shot prompting in LLMs for multi-class prediction problems with a high number of classes. While prompting with long contexts may "confuse" the LLM (Liu et al., 2023), the context window may even be too small to support a view of all classes in such cases. Furthermore, the previous approaches focus on selecting specific samples rather than relevant classes for the prompt.

The CP framework (Vovk et al., 2005) is a method for associating predictions of a classification algorithm with confidence guarantees. Essentially, CP selects sets of classes from the predicted labels of a classifier or regressor that statistically contain the true label with a predefined probability. We note, however, that the standard CP setting pro-

vides guarantees for the individual class labels; i.e., errors for one of the labels can, hence, exceed the set guarantees. A formulation, called Mondrian of CP, can make the guarantees hold also for the individual labels. Note that CP does a very good job of keeping the output (set of labels) small, while the prediction guarantees hold. It can be applied to any classification (or regression) model, as long as a calibration set is set aside together with a measure of the model's prediction error with regard to the true labels. This measure is referred to as the "*non-conformity function*". In our context, the stronger the model, and the better the non-conformity function, the fewer labels (hence, shots) will be produced with the same guarantee. We point to the work of Vovk et al. (2005); Johansson et al. (2014); Boström et al. (2017) for more information on CP.

3 The Food Recall Dataset

In this section, we first describe the dataset used and shared in this paper, then discuss the labeling process and quality, as well as class distribution.

3.1 Data Description

The dataset consists of 7,546 short texts (length in characters: min=5, avg=84, max=360), which are the titles of food-recall announcements (therefore referred to as title), crawled from 24 food-recall domains (governmental & NGO, see Table 3, Appendix A) by Agroknow[2] (Papakonstantinou et al., 2022). The texts are written in 6 languages, with English ($n = 6,644$) and German ($n = 888$) being the most common, followed by French ($n = 8$), Greek ($n = 4$), Italian ($n = 1$) and Danish ($n = 1$). As shown in Figure 1, most of the texts have been authored after 2010. The texts describe recalls of specific food products due to specific reasons. Experts manually classified each text to four groups of classes describing hazards and products on two levels of granularity:

hazard: A fine-grained description of the hazards mentioned in the texts comprising 261 classes.
hazard-category: A categorized version of the hazard classification task comprising 10 classes.
product: A fine-grained description of the products mentioned in the texts comprising 1,256 classes. **product-category**: A categorized version of the product classification task comprising 22 classes. The columns hazard-title and product-title comprise character spans, gener-

---

ated based on feature importance of a Logistic Regression (LR) classifier (see §3.1.1). These signify parts of the title that are important for the hazard and product classification. As the fine-grained tasks, hazard and product, have very low support for many classes, they may require further pre-processing (e.g. clustering/filtering of labels) depending on the application. The dataset, publicly released under a Creative Commons BY-NC-SA 4.0 license, comprises also metadata, such as the release date of the text (columns year, month, and day), the language of the text (column language), and the country of issue (column country). Appendix A contains more detailed statistics, as well as sample texts.
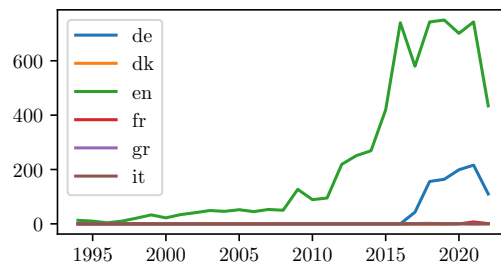

Figure 1: Languages in the dataset per year.

3.1.1 Ground Truth

It is important to note that samples were labeled not only based on the title, but also the content of the food recall article. This means that some of the samples may not contain evidence for all the classes assigned to them. As samples that are missing specific information are common in real-world data, we decided not to filter out such samples; instead, we provide an estimate of such noise in the data in the hazard-title and product-title classification tasks. To that end, we use the coefficients of the tf-idf Logistic Regression (TF-IDF-LR) classifiers for the hazard-category and product-category classes to extract important terms per class.

For each text-label pair T_i, Y_i, we split T_i in tokens $\{t_{i,1}, t_{i,2}, \ldots, t_{i,K}\}$, using the process described in §4.2.1. We then calculate a score by adding the positive model coefficients associated with $t_{i,k}$ if $y_{i,j} = 1$, and subtracting the positive model coefficients associated with $t_{i,k}$ if $y_{i,j} = 0$. Although the quality of these terms depends on class support, they can still help us frame the noise in the data by focusing on informative tokens, i.e., tokens with a positive coefficient for a specific class.
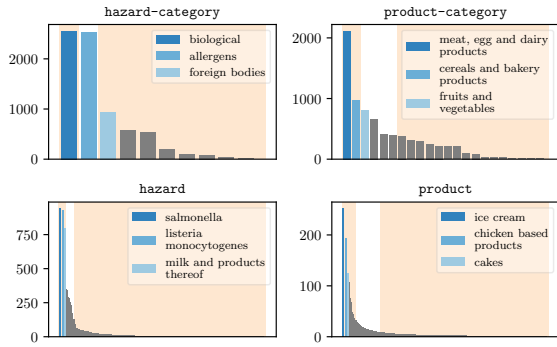
Figure 2: Class support with orange background for high- (on the left) and low-support classes (right) for performance analysis. For reasons of better readability, we only name the three most supported classes per task.

We find that each such token corresponds to 1.16 classes on average for the `hazard-category` and 1.30 classes for the `product-category`.[3] Also, we see that 13.1% of the samples do not have spans in `hazard-title` (17.1% for `product-title`), indicating that evidence for the class is missing.

Fine-grained labels were assigned per webdomain by a domain expert of Agroknow and then grouped to the final labels using internal ontologies.[4] In order to measure inter-annotator agreement (IAA), we asked a senior curator involved in the labeling process and one independent expert to label a sample of 30 texts (stratified over both `*-category` labels) independently of each other. Comparing the annotator's assessment to the labels in the dataset shows moderate agreement (Cohen's κ: 0.76 on `hazard-category`, 0.63 on `product-category`). The expert's assessment is intuitively slightly less aligned (Cohen's κ: 0.63 on `hazard-category`, 0.60 on `product-category`) but still acceptable. Expert and Annotator show a κ of 0.59 on `hazard-category`, 0.77 on `product-category`. The raw data of our IAA study are available along with the code.

3.2 Class Imbalance

One of the most prominent features of the data is the heavy class imbalance. Figure 2 shows the sample counts per class and classification task. All the classification tasks in the data show a long-tail distribution of classes, with just a small number of classes having most of the samples. Therefore, we extract sets of high-support classes $\mathcal{C}_{high}$ and

low-support classes $\mathcal{C}_{low}$ comprised of around onethird of the total number of samples in the data for each classification task. The classes included in these sets are highlighted by an orange background in Figure 2. For the `hazard-category` classification task, the $\mathcal{C}_{high}$ is comprised of only one class with $2,557$ samples, and the $\mathcal{C}_{low}$ consists of 8 classes with $2,462$ samples in total, and for `hazard` $|\mathcal{C}_{high}| = 2,668$ samples in 3 classes and $|\mathcal{C}_{low}| = 2,414$ samples in 248 classes. For `product-category` we have $|\mathcal{C}_{high}| = 3,089$ samples in 2 classes and $|\mathcal{C}_{low}| = 2,185$ samples in 16 classes, and for `product` $|\mathcal{C}_{high}| = 2,508$ samples in 39 classes and $|\mathcal{C}_{low}| = 2,483$ samples in $1,058$ classes.

4 Methodology

This section describes the problem setting and the methods utilized in our empirical evaluation.

4.1 The Task

The basic problem we address in this paper is extreme multi-class classification on heavily imbalanced data. More formally, given a number of training texts T_i, $i \in \{1, 2, \ldots, N\}$ and their corresponding classes y_i, we aim to train a classifier $f(T_i, \cdot) = \hat{y}_i$ that minimizes the error $|y_i - \hat{y}_i|$. For standard ML classifiers, the function $f(T_i, \cdot)$ is usually a two-step process with the first step mapping the text to a machine-readable embedding vector $X_i = e(T_i)$, and the second one involving the learning process on X_i: $f(T_i, \cdot) = f'(X_i, \cdot)$. Both the class labels y_i and predictions $\hat{y}_i$ are integer scalars in $\{1, 2, \ldots, M\}$. In all our classification tasks, we have at least $M \geq 10$.

4.2 Classifiers

As a naive baseline, we report the performance of two classifiers. The random classifier (RANDOM) yields a random integer in $\{1, 2, \ldots, M\}$ for each $\hat{y}_i$. The majority baseline (MAJORITY) chooses each $\hat{y}_i$ to be the class with the highest number of samples in the training data.

4.2.1 Traditional ML Classifiers

We use Bag-of-Words (BOW) and Term Frequency - Inverse Document Frequency (Spärck Jones, 1972, TF-IDF) encodings (see Appendix C), combined with k-Nearest Neighbours (KNN), LR, or Support Vector Machine (SVM) classifiers.[5] For KNN,

---

[3]Estimation for the fine-grained classification tasks is difficult because of low per-class support.

[4]Random checks of the labels were performed by more experienced curators for quality assurance.

[5]For the classifiers, we use the implementation from the Python library `scikit-learn` (Pedregosa et al., 2011).

we optimize over $k \in \{2, 4, 8\}$ neighbours based on cosine similarity of the embedding vectors. For LR, we use a 'liblinear' solver and optimize $C \in \{0.5, 1.0, 2.0\}$ for both L1 and L2 regularization. For the SVMs, we use a linear kernel and optimize the parameter $C \in \{0.5, 1.0, 2.0\}$ for L2 regularization on the validation splits (defined in §5.1). For LR and SVM, we use a one-vs-all approach to classification, which means that we train one binary classifier for each class in the predicted classification task and choose the class with the highest probability.

4.2.2 Encoder-only Transformers

As a more recent counterpart to the previously described traditional ML classifiers, we fine-tune two models from huggingface's Transformers[6] library: RoBERTa$_{base}$ (Liu et al., 2019) and XLM-RoBERTa$_{base}$ (Conneau et al., 2020, XLM-R) in their base-sizes (RoBERTa: *125M params*; XLM-R: *270M params*). Both models use the structure introduced by BERT$_{base}$ (Devlin et al., 2019, *L=12, H=768, A=12*), which improves comparability of their results. The different parameter counts result mainly from the different vocabulary sizes used in their Byte-Pair-based encoders (Sennrich et al., 2016): RoBERTa uses a vocabulary of size 50k, while XLM-R uses 250k tokens. For the purposes of this paper, the most important difference between the models is that while RoBERTa is only pre-trained on English texts, XLM-R is pre-trained on 100 different languages.

To fine-tune these two models, we use the standard sequence classification heads provided by the Transformers library. We optimize training using AdamW (Loshchilov and Hutter, 2019) in combination with a learning rate that stays constant at a value of $5 \cdot 10^{-5}$ for the first two epochs and then declines linearly towards 1% of its starting value after 20 epochs. Furthermore, we employ early stopping with a patience of five epochs on the minimum cross-entropy loss computed on the validation set. Due to hardware limitations, we use a batch size of 16 observations.

4.3 Prompting

Prompt engineering can lead to accurate classification results with limited training data. We investigate this path for our task by employing few-shot in-context learning with

---

[6]https://huggingface.co/docs/transformers/index

gpt-3.5-turbo-instruct.[7] For all our prompting examples, we use a temperature value of 0. The naive approach is to provide a context describing the classification task followed by the two most similar text-label pairs (based on cosine similarity of the TF-IDF-embeddings) per class from the training data. These examples are ordered from the most similar to least similar in the prompt (dubbed **GPT-ALL**).
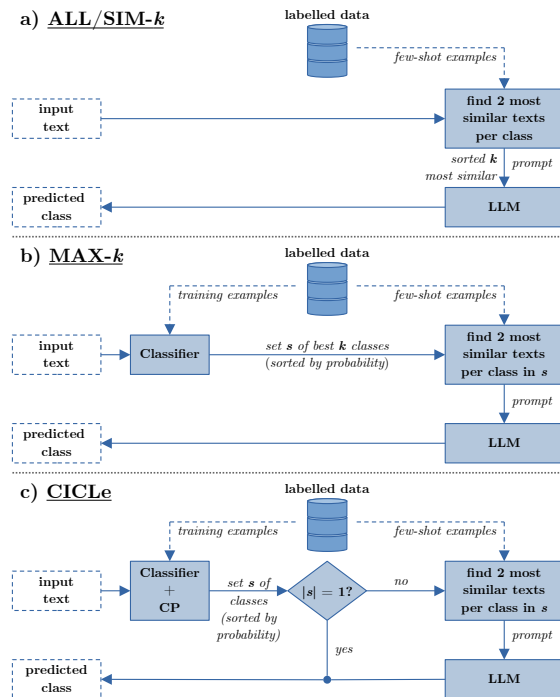
Figure 3: Prompting strategies used in this study. Although the strategies are generally independent of the choice of classifier and LLM, we only show results for TF-IDF-LR and OpenAI's gpt-3.5-turbo-instruct.

GPT-ALL is applicable only to coarse tasks due to the size of the context window: for the hazard task, we would have at most $\frac{4{,}096 \text{ tokens}}{261 \text{ classes}} = 15.7$ tokens per class which is insufficient for providing two example texts. Furthermore, reduced prompt sizes can lead to increased energy efficiency, as the size of the attention weights computed in the Transformer is *input size × output size* (Vaswani et al., 2017). Lastly, meaningfully reducing the classes in the prompt can increase predictive performance, as the probability of choosing the right class increases (Appendix D). Therefore, we use three different kinds of prompts (see Figure 3) with the third kind (GPT-CICLe) being novel. **GPT-SIM-k** is simply the GPT-ALL prompt shortened by only keeping the k most similar few-shot samples. **GPT-MAX-k**

---

[7]Accessed on 02-2024 through OpenAI's Python API.

selects the two most similar few-shot samples from the k most probable class labels as predicted by a base classifier. The few-shot samples are ordered from most to least probable, which yields $2 \cdot k$ few-shot samples per prompt.

4.3.1 Our proposed GPT-CICLe

We consider that the better the few shots the LLM is provided with, the better its performance will be. Therefore, in this work, we propose using Conformal Prediction (CP) (Vovk et al., 2005) to adapt the context length to a base classifier's certainty on the predicted classes. CP uses the concept of a non-conformity measure $\delta(y_i, \hat{y}_i)$ to create sets of predicted classes, which contain the true class with a probability of $p \approx 1 - \alpha$.

In our case, this non-conformity measure is simply the base classifier's uncertainty on the true class:

$$\delta(y_i, \hat{y}_i) = 1 - P(\hat{y}_i = y_i).$$

Here, $\hat{y}_i$ are the predictions and y_i are the corresponding true labels. If one defines

$$q = \frac{\lceil (N+1)(1-\alpha) \rceil}{N},$$

and $\hat{q}$ as the qth empirical quantile of $\{\delta(Y_i, \hat{Y}_i) | 1 \le i \le N\}$, it can be shown that for any prediction $\hat{Y}$ of the classifier on an unknown sample, a set of classes $\{j | \hat{y}_j \ge 1 - \hat{q}\}$ contains the true class with probability $p \ge 1 - \alpha$ (Vovk et al., 2005). This means that we accept an upper bound of $1 - \alpha$ on prompting accuracy, which we trade for adaptive context length based on sample difficulty.

In the extreme case of a single class in the prediction set, we bypass the LLM completely and predict this class. As it is possible for CP to predict empty sets, which is not useful in classification, we return a set containing only the most probable class in such cases. This means that for $\alpha > 1 - base$ *classifier accuracy*, the performance of this prompt becomes equal to that of the base classifier. As in GPT-MAX-k, we finally select the two most similar samples per class in the conformal set from the training data to compose the prompt. Samples are again ordered from most to least probable as determined by the classifier. We refer to this LLM-in-the-loop framework as *Conformal In-Context Learning* (CICLe). An example of each of these prompts is shown in Appendix B.

In this paper, we focus on the pure few-shot performance of gpt-3.5-turbo-instruct. Therefore, we decided not to employ additional prompt-engineering techniques, such as CoT. Our code is publicly available (under a GPL V3.0 license) at https://github.com/k-randl/conformal_prompting.

5 Empirical Analysis

In this section, we present the experimental setup and the experimental results.

5.1 Experimental Setup

For training and evaluation of our ML models, we apply 5-fold cross-validation (CV) to create 5 train-test splits. From each of these 5 training sets, we create a validation set using 10% holdout which we use for hyperparameter tuning as described in Appendix C. To reduce the computational cost, we only test our prompting methods other than GPT-ALL on the first CV split, which corresponds to a 20% holdout. We use stratification on the *-category classification tasks as those provide a sufficient number of samples for splitting in each class. We use the same splits for hazard-category and hazard, as well as product-category and product to keep the results comparable. Our classifiers provide baseline performance on the dataset. In order to demonstrate the effect of class imbalance on performance, we do not employ balancing methods like oversampling or class weights during training.

5.2 Experimental Results

In this section, we present the predictive performance of the classifiers described in §4. For hazard and product we present performance on classes that have a minimum support of four in the train set and at least one instance in the test set as for these ML is feasible. Performance on all classes in these tasks is reported in Appendix D.

5.2.1 Traditional Baselines

The F_1-scores on classification (overall, as well as $\mathcal{C}_{high}$ and $\mathcal{C}_{low}$ introduced in Figure 2) are presented in Table 1. All the classifiers outperform both naive baselines, except for the $\mathcal{C}_{high}$ of hazard-category. In this task, $\mathcal{C}_{high}$ only consists of the majority class. This means that classification on this task is not useful as there is only one label present in the test set. Every class not predicted to this label (even if correctly predicted in general) will therefore diminish the macro F_1.

Model	F_1 (*all classes*) mean	max.	F_1 ($\mathcal{C}_{high}$) mean	max.	F_1 ($\mathcal{C}_{low}$) mean	max.
hazard-category						10 *classes*
RANDOM	0.08	0.08	0.33	0.34	0.00	0.00
MAJORITY	0.05	0.05	1.00	1.00	0.00	0.00
TF-IDF-KNN	0.47	0.50	0.45	0.46	0.41	0.46
TF-IDF-LR	0.55	0.57	0.48	0.48	0.50	0.53
TF-IDF-SVM	**0.58**	**0.60**	0.47	0.47	**0.56**	**0.59**
RoBERTa	0.51	0.55	0.47	0.48	0.45	0.51
XLM-R	0.49	0.52	0.47	0.48	0.42	0.47
GPT-ALL	0.56	0.62	0.45	0.46	**0.56**	**0.67**
product-category						22 *classes*
RANDOM	0.01	0.01	0.20	0.22	0.01	0.01
MAJORITY	0.01	0.01	0.24	0.24	0.00	0.00
TF-IDF-KNN	0.42	0.47	0.78	0.79	0.43	0.49
TF-IDF-LR	0.52	0.57	**0.87**	**0.88**	0.49	0.56
TF-IDF-SVM	0.59	0.60	0.86	0.88	0.59	0.61
RoBERTa	0.30	0.53	0.68	0.90	0.26	0.48
XLM-R	0.35	0.52	0.78	0.89	0.30	0.48
GPT-ALL	**0.69**	**0.72**	0.84	0.85	**0.73**	**0.75**
hazard						79 / 261 *classes*
RANDOM	0.00	0.00	0.00	0.00	0.00	0.00
MAJORITY	0.00	0.00	0.00	0.00	0.00	0.00
TF-IDF-KNN	0.28	0.30	0.71	0.75	0.28	0.30
TF-IDF-LR	**0.41**	**0.44**	0.83	0.85	**0.39**	**0.43**
TF-IDF-SVM	0.23	0.24	0.84	0.86	0.17	0.18
RoBERTa	0.36	0.41	**0.85**	**0.86**	0.32	0.37
XLM-R	0.33	0.38	0.83	0.86	0.29	0.35
product						184 / 1256 *classes*
RANDOM	0.00	0.00	0.00	0.00	0.00	0.00
MAJORITY	0.00	0.00	0.00	0.00	0.00	0.00
TF-IDF-KNN	0.32	0.35	0.46	0.51	0.24	0.35
TF-IDF-LR	**0.51**	**0.53**	**0.68**	**0.72**	**0.43**	**0.52**
TF-IDF-SVM	0.09	0.10	0.50	0.52	0.00	0.00
RoBERTa	0.43	0.57	0.56	0.72	0.28	0.52
XLM-R	0.28	0.54	0.44	0.76	0.14	0.41

Table 1: Macro-averaged F_1-score (*mean & maximum*) over 5 CV splits. Blue cells highlight the best mean score per column and classification task. For hazard and product scores are only evaluated on well-supported classes. The grayed scores are added for completeness, but not comparable, as $\mathcal{C}_{high}$ of hazard-category consists of only one class.

Since the BOW-based classifiers in general perform worse than TF-IDF, we only report them in the Appendix (see Table 8).

The best-performing non-prompting classifier for the hazard-category and product-category tasks is TF-IDF-SVM as it excels on $\mathcal{C}_{low}$. For the fine-grained hazard and product tasks the SVM's performance on the less represented classes drops, and TF-IDF-LR takes the position as the best traditional classifier. In parallel, TF-IDF-SVM's relative F_1-score increases on high support classes, which we interpret as an inability to learn from classes with very few samples, paired with good performance on training sample counts > 100. TF-

IDF-KNN has the overall lowest performance of the traditional classifiers.

5.2.2 Transformers

TF-IDF-LR performs overall comparably or better than the best encoder-only Transformer RoBERTa. We assume that the Transformer architectures would need higher numbers of training samples per class to perform on par or better compared to traditional methods.

Interestingly, RoBERTa and XLM-R both achieve lower and less stable F_1-score for product and product-category. This is not seen in other classifiers. By inspecting the data, we see that in both of these tasks brand names (e.g. "SnoBalls", "Sriracha") or hard-to-define word creations (e.g. "pecan caramel stars", "Vanilla Cups") make up the relevant part for classification. These are hard to learn during fine-tuning, as they are both very specific and most probably underrepresented in the data used for pre-training. Analysis of the spans in product-title shows that TF-IDF-LR often infers the product from the supplier and/or the hazard in such cases.

Surprisingly, the multilingual XLM-R only outperforms RoBERTa in the $\mathcal{C}_{high}$ of the hazard-category and product-category tasks even though the texts come in multiple languages. We assume the low number of per-class samples is not sufficient for the very large embedding layer of XLM-R. Analysis of the performance on the two most common languages, English and German, is shown in Appendix D, Table 7. The results suggest that while XLM-R performs closer to RoBERTa on non-English texts, its performance in terms of F_1 is generally worse than that of RoBERTa. XLM-R performs especially worse than RoBERTa on German texts in tasks with low per-class support (namely hazard and product), which supports our claim that XLM-R needs more training examples than RoBERTa to deliver good performance.

5.2.3 Large Language Models

A simple prompting approach (GPT-ALL) performs comparably to TF-IDF-LR and RoBERTa on the hazard-category and is the best classifier for product-category. In both cases, it excels on the $\mathcal{C}_{low}$ as it only needs two labeled samples per class. Since when prompting, the predicted class is delivered in free text, the LLM may produce output that is not within the set of class labels. While these outputs could be interpreted as belonging to one of the class labels, we only count exact matches.

In both coarse tasks, GPT-ALL failed to predict any class for 2% of the samples (Table 2). In a qualitative analysis, shown in Appendix D, we see that while some of the prompting failures can be improved by better prompt design (e.g. more samples per class, see also Table 2), others can even be seen as a positive feature of prompting.

Model	F_1 (macro)	LLM usage	Prompt Size (chars)	Classes per Prompt	Samples per Class	Failure Rate
hazard-category					10 *classes*; $\alpha = 0.05$	
TF-IDF-LR	0.55	-	-	-	-	-
GPT-ALL	0.53	100%	2301.7	10.0	2.0	2%
GPT-SIM-10	0.52	100%	1206.3	5.8	1.7	2%
GPT-MAX-5	0.55	100%	1214.9	5.0	2.0	2%
GPT-SIM-20	0.53	100%	2301.7	10.0	2.0	2%
GPT-MAX-10	0.49	100%	2301.7	10.0	2.0	3%
GPT-CICLe	**0.56**	60%	**707.5**	2.6	2.0	2%
product-category					22 *classes*; $\alpha = 0.05$	
TF-IDF-LR	0.52	-	-	-	-	-
GPT-ALL	0.72	100%	5621.1	22.0	2.0	2%
GPT-SIM-10	0.66	100%	**1378.6**	6.6	1.5	4%
GPT-MAX-5	0.64	100%	1395.6	5.0	2.0	4%
GPT-SIM-20	0.69	100%	2614.0	11.6	1.7	2%
GPT-MAX-10	0.68	100%	2631.8	10.0	2.0	6%
GPT-CICLe	**0.69**	98%	1620.1	5.9	2.0	5%
hazard					102 / 261 *classes*; $\alpha = 0.20$	
TF-IDF-LR	0.37	-	-	-	-	-
GPT-SIM-10	**0.44**	100%	1308.0	7.5	1.3	15%
GPT-MAX-5	0.42	100%	1335.4	5.0	2.0	14%
GPT-SIM-20	**0.44**	100%	2457.0	14.2	1.4	11%
GPT-MAX-10	0.42	100%	2521.1	10.0	2.0	13%
GPT-CICLe	0.42	87%	**1251.6**	4.7	2.0	14%
product					257 / 1256 *classes*; $\alpha = 0.40$	
TF-IDF-LR	0.54	-	-	-	-	-
GPT-SIM-10	0.54	100%	**1238.7**	8.5	1.2	34%
GPT-MAX-5	0.56	100%	1270.0	5.0	2.0	20%
GPT-SIM-20	0.53	100%	2324.3	16.7	1.2	36%
GPT-MAX-10	0.57	100%	2385.2	10.0	2.0	23%
GPT-CICLe	**0.58**	96%	1645.8	6.7	2.0	24%

Table 2: Comparison of reduced prompting strategies. Blue cells highlight the best performance of reduced prompts per column and classification task; green cells the most resource-friendly value. For hazard and product F_1 is only evaluated on well-supported classes.

Reduced prompts Naive few-shot prompting (e.g., with two shots per class) may work for few but is not optimal for a high number of classes. In order to reduce the number of classes for which we include shots in the prompt, we compare three approaches that meaningfully reduce the context length. A condensed version of the results for reduced prompting is shown in Table 2 (the full version is in Table 9 of the Appendix D). Simply reducing the number of few-shot samples by moving to the GPT-SIM-k prompts also reduces the overall F_1-scores[8] as performance on $\mathcal{C}_{low}$ drops (see Table 9). This is largely owed to the circumstance that the $\mathcal{C}_{low}$ makes up the bulk of the classes and we

---

[8]GPT-ALL and GPT-SIM-20 are identical in hazard-category

are more likely to remove samples of a low-support class than the less prominent high-support classes. The latter actually profit from the reduction, as they now compete with fewer other classes. An analysis of the effect of reducing classes in the prompts on maximum and minimum performance is shown in Appendix D. Nevertheless, this very simple setup is already very powerful and outperforms the more sophisticated prompts on product.

A noteworthy detail of the hazard-category task is that it contains exactly ten classes. This means that GPT-ALL, GPT-SIM-20, and GPT-MAX-10 are identical apart from sample order: while GPT-ALL and GPT-SIM-20 are ordered by cosine similarity, GPT-MAX-10 is ordered by class probability as determined by the base classifier. While the evidence is anecdotal, the F_1-scores suggest that sample order is important in prompting and that ordering by similarity may be better.

An estimate of the capabilities of *-MAX-k vs. *-CICLe is shown in Figure 4: CP leads to more concise prediction sets (i.e. smaller sets at the same set accuracy) than GPT-MAX-k. In contrast to max-k, CP produces shorter sets if the classifier has a high certainty on the true class. Nevertheless, classifiers with lower overall performance will improve on max-k's accuracy only at larger sets. This is also reflected in Table 2: GPT-CICLe performs better or at least equal to GPT-MAX-k in all tasks. **Focusing on GPT-CICLe**, Table 2 shows that it outperforms its base classifier TF-IDF-LR on all tasks as well as all reduced prompting baselines for three out of four tasks. For higher numbers of classes, the performance of GPT-CICLe drops in $\mathcal{C}_{low}$ compared to the other prompts (see Table 9). We attribute this to the higher α in these cases: here we sacrifice CP guarantees for faster prediction. GPT-CICLe also produces the lowest prompt lengths at the same F_1 (Table 2). Here, we argue, that the base classifier used by GPT-MAX-k and GPT-CICLe is able to produce more concise contexts for the LLM than simple cosine similarity. While this is already a step towards energy-efficient prompting, CICLe can avoid accessing the LLM completely in some cases: the extent depends on the task and the α, but ranges between accessing the LLM in only 60% to 98% of cases in our tasks. The selection of alpha can in theory be done through an extensive hyperparameter search. As this becomes very computationally expensive for LLMs, we select α based on Figure 4 in our experiments (see Appendix C). We leave computationally
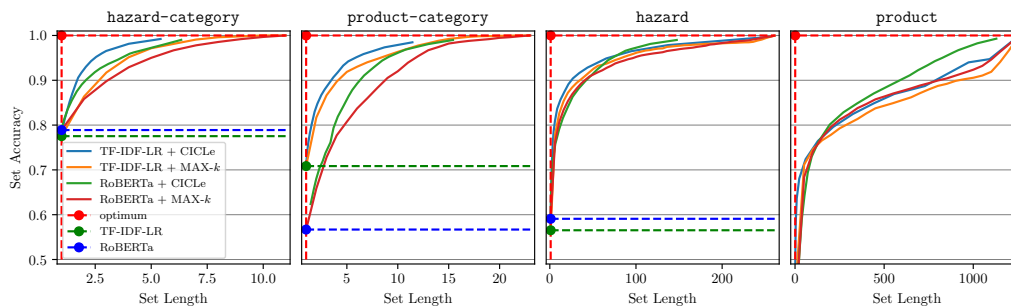
Figure 4: CP performance on the best traditional classifier (green dashed line) and the best encoder-only Transformer (blue dashed line). The optimal prediction set (red dashed line) would have one element, which is the true class with a probability of 1.

efficient optimization of α for future research.

6 Discussion

Our results suggest that a general reduction of few-shot examples, taken from a population that is very likely to contain the true class, can outperform traditional ML classifiers and Transformers in terms of F_1 even without using other prompting techniques, e.g., example matching (Ahmed et al., 2023) or CoT. This makes simple selection of few shots based on cosine similarity (GPT-SIM-k) a powerful approach, especially for tasks with a high number of classes or extraction tasks. In this setting, we leverage the well-documented strong in-context reasoning capabilities of LLMs (Chowdhery et al., 2022; Wei et al., 2022; Shi et al., 2023), which do not necessarily need a full view of all possible classes. However, in cases where we are able to train a decently performing base classifier, CICLe is a better choice: it boosts the performance of the base classifier by detecting insecure samples using CP and redirecting them to an LLM. Further analysis (see Appendix D.3) shows that CICLe's strength is mainly based on better performance on low-support classes compared to the base classifier. This is most probably owed to the low demand for "training data" in few-shot prompting. In general, the influence of the base classifier and LLM on CICLe's performance is controlled by the α value: low values favor the LLM, while high values favor the base classifier. Lastly, CICLe is subject to a trade-off between predictive and temporal performance: it allows sacrificing accuracy (e.g. due to hardware and time constraints) while keeping the context for the LLM as concise as possible. This makes it a resource-friendly alternative to prompting with all classes. Our results further suggest, in line with Liu et al. (2023), that sample order and context length are highly influential on prompting performance. We therefore see a need for future

work on optimal ordering of few-shot samples and tuning of the α parameter within CICLe.

7 Conclusions

We present the first dataset for large-scale multi-class classification of short texts describing food recalls, which we release publicly under a CC BY-NC-SA 4.0 license. The dataset contains labels annotated by experts on food products and hazards on two levels of granularity, coarse (tens) and fine (hundreds).

Answering our research question **R1**, we find that the titles of food recall announcements mostly hold sufficient information for classification, although not all of them include information on both product and hazards (**R1.1**). A strong indication for this statement is that we are able to reach above random performance with multiple classifiers: we carry out benchmarking, analysing the performance of naive, traditional, and deep-learning classifiers for all four classification tasks in the proposed dataset, showing the potential of traditional classifiers on this dataset. Our results further indicate that products are harder to classify than hazards, which we attribute to the often creative choice of product names (**R1.2**).

Moving on to **R2**, we also experimented with prompting using one few-shot example per class. Here we see that gpt-3.5 performs comparably or better than the best non-LLM classifiers (**R2.1**). We also show that resource-efficient prompting at comparable predictive performance is possible by proposing a novel LLM-in-the-loop framework that leverages Conformal Prediction (**R2.2**).

8 Limitations

Nevertheless, the dataset and approach discussed in this paper are subject to limitations. Regarding the dataset, we identified the following:

- The labels in our dataset are subject to noise. Specifically, some samples are missing evidence for one or more of the assigned classes, while tokens important for classification may indicate more than one class. This may lead to classifiers trained on the data seeing contradicting examples and therefore limit their predictive performance.

- The spans in `hazard-title` and `product-title` are machine generated and not manually curated. This means that while they give an estimation of word importance, they are no gold standard for explainability tasks.

We aim to improve these limitations in future iterations of the dataset. For our approach leveraging CP for few-shot prompting, we found the following limitations:

- As visualized in Figure 4, CP represents a trade-off between high prediction set accuracy and low set length relative to the total number of classes. This means that with a rising total number of classes, we will have to sacrifice predictive performance in order to keep the prompt size feasible, which ultimately might render the approach useless.

- In this paper we used normal conformal prediction, which only guarantees a certain probability of a single true class being in the prediction set. In order to achieve true multilabel guarantees we would need to switch to mondrian CP.

- We only verify our approach on a single LLM. This means that our approach might not be generalizable to other LLMs and perform differently or not at all for few-shot prompting with such models.

While we have to accept the first of these points as inherent to the approach, we are planning to address the remaining points in future work. As all the data used in our dataset was already publicly available before the publication of this work, we do not violate anybody's privacy by republishing it.

Acknowledgements

References

Toufique Ahmed, Kunal Suresh Pai, Premkumar Devanbu, and Earl T. Barr. 2023. Improving few-shot prompts with relevant static analysis products.

S. Bird, E. Loper, and Klein E. 2009. *Natural Language Processing with Python*. O'Reilly Media Inc.

Henrik Boström, Lars Asker, Ram Gurung, Isak Karlsson, Tony Lindgren, and Panagiotis Papapetrou. 2017. Conformal prediction using random survival forests. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 812–817.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira,

Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale.

Charline Maertens de Noordhout, Brecht Devleesschauwer, Frederick J Angulo, Geert Verbeke, Juanita Haagsma, Martyn Kirk, Arie Havelaar, and Niko Speybroeck. 2014. The global burden of listeriosis: a systematic review and meta-analysis. *The Lancet Infectious Diseases*, 14(11):1073–1082.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners.

Ruofan Hu, Dongyu Zhang, Dandan Tao, Thomas Hartvigsen, Hao Feng, and Elke Rundensteiner. 2022. Tweet-fid: An annotated dataset for multiple foodborne illness detection tasks.

Cangyu Jin, Yamine Bouzembrak, Jiehong Zhou, Qiao Liang, Leonieke M. van den Bulk, Anand Gavai, Ningjing Liu, Lukas J. van den Heuvel, Wouter Hoenderdaal, and Hans J.P. Marvin. 2020. Big data in food safety - a review. *Current Opinion in Food Science*, 36:24–32. Food Safety.

Ulf Johansson, Henrik Boström, Tuve Löfström, and Henrik Linusson. 2014. Regression conformal prediction with random forests. *Machine learning*, 97:155–176.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the middle: How language models use long contexts.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach.

I. Loshchilov and F. Hutter. 2019. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019*, University of Freiburg.

Adyasha Maharana, Kunlin Cai, Joseph Hellerstein, Yulin Hswen, Michael Munsell, Valentina Staneva, Miki Verma, Cynthia Vint, Derry Wijaya, and Elaine O Nsoesie. 2019. Detecting reports of unsafe foods in consumer product reviews. *JAMIA Open*, 2(3):330–338.

Shannon E. Majowicz, Elaine Scallan, Andria Jones-Bitton, Jan M. Sargeant, Jackie Stapleton, Frederick J. Angulo, Derrick H. Yeung, and Martyn D. Kirk. 2014. Global incidence of human shiga toxin–producing escherichia coli infections and deaths: A systematic review and knowledge synthesis. *Foodborne Pathogens and Disease*, 11(6):447–455. PMID: 24750096.

Mihalis Papakonstantinou, Manos Karvounis, Giannis Stoitsis, and Nikos Manouselis. 2022. *Deploying a Scalable Big Data Platform to Enable a Food Safety Data Space*, pages 227–248. Springer International Publishing, Cham.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

R. Sennrich, B. Haddow, and A. Birch. 2016. Neural machine translation of rare words with subword units. In *54th Annual Meeting of the Association for Computational Linguistics, ACL 2016 - Long Papers*, volume 3, pages 1715–1725, School of Informatics, University of Edinburgh.

Fobo Shi, Peijun Qing, Dong Yang, Nan Wang, Youbo Lei, Haonan Lu, and Xiaodong Lin. 2023. Prompt space optimizing few-shot reasoning success with large language models.

K. Spärck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28:11–21.

Dandan Tao, Ruofan Hu, Dongyu Zhang, Jasmine Laber, Anne Lapsley, Timothy Kwan, Liam Rathke, Elke Rundensteiner, and Hao Feng. 2023. A novel foodborne illness detection and web application tool based on social media. *Foods*, 12(14).

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu,

Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. 2005. *Algorithmic Learning in a Random World*. Springer International Publishing.

Xinxin Wang, Yamine Bouzembrak, AGJM Oude Lansink, and H. J. van der Fels-Klerx. 2022. Application of machine learning to the monitoring and prediction of food safety: A review. *Comprehensive Reviews in Food Science and Food Safety*, 21(1):416–434.

Xinxin Wang, Yamine Bouzembrak, A. G. J. M. Oude Lansink, and H. J. van der Fels-Klerx. 2023. Weighted bayesian network for the classification of unbalanced food safety data: Case study of risk-based monitoring of heavy metals. *Risk Analysis*, n/a(n/a).

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Shufeng Xiong, Wenjie Tian, Vishwash Batra, Xiaobo Fan, Lei Xi, Hebing Liu, and Liangliang Liu. 2023. Food safety news events classification via a hierarchical transformer model. *Heliyon*, 9(7):e17806.

Lei Zhou, Chu Zhang, Fei Liu, Zhengjun Qiu, and Yong He. 2019. Application of deep learning in food: A review. *Comprehensive Reviews in Food Science and Food Safety*, 18(6):1793–1811.

Appendix

A Data sample and statistics:

In this appendix, we provide additional insights into the dataset shared with this publication. Table 3 lists the original websites our texts were collected from, Table 4 provides a statistical overview over the class- and language-distribution in the data, and Figure 5 is a heatmap highlighting co-occurrence of classes in the hazard-category and product-category tasks. Finally, we share a few data samples along with their labels in Figure 6.

Domain	Samples
www.fda.gov:	1740
www.fsis.usda.gov:	1112
www.productsafety.gov.au:	925
www.food.gov.uk:	902
www.lebensmittelwarnung.de:	886
www.inspection.gc.ca:	864
www.fsai.ie:	358
www.foodstandards.gov.au:	281
inspection.canada.ca:	124
www.cfs.gov.hk:	123
recalls-rappels.canada.ca:	96
tna.europarchive.org:	52
wayback.archive-it.org:	23
healthycanadians.gc.ca:	18
www.sfa.gov.sg:	11
www.collectionscanada.gc.ca:	10
securite-alimentaire.public.lu:	8
portal.efet.gr:	4
www.foodstandards.gov.scot:	3
www.ages.at:	2
www.accessdata.fda.gov:	1
webarchive.nationalarchives.gov.uk:	1
www.salute.gov.it:	1
www.foedevarestyrelsen.dk:	1

Table 3: Data sources, ordered by support number
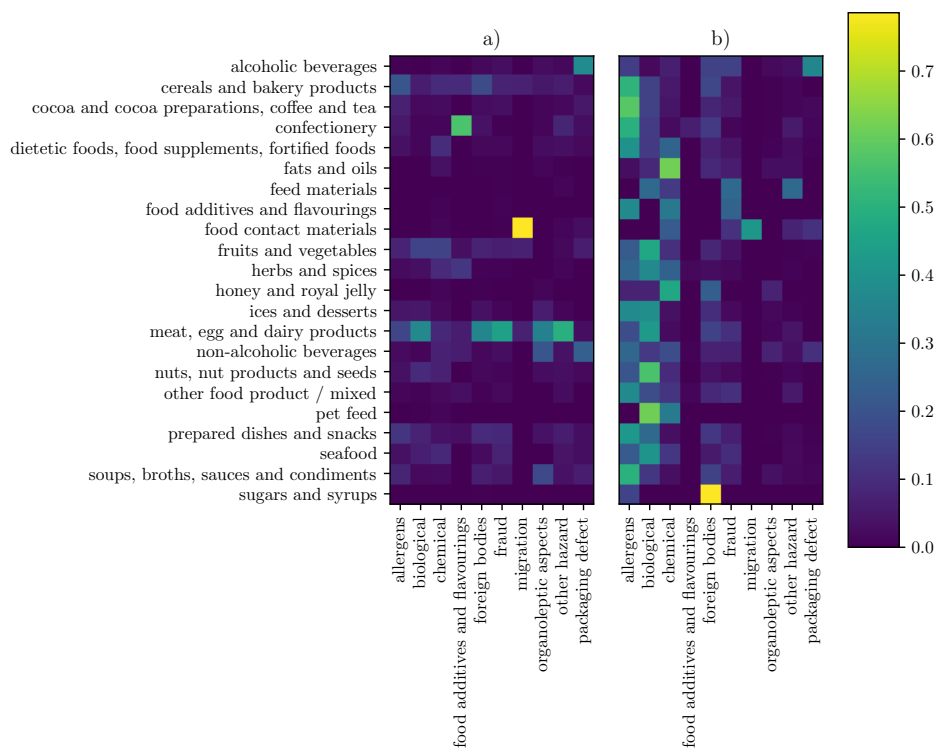
Figure 5: Label co-occurence for the `hazard-category` and `product-category` tasks normalized by **a)** `hazard-category`, and **b)** `product-category`. While this is not true for all the `hazard-category`-`product-category` pairs, some show strong linkage.

" Randsland brand Super Salad Kit recalled due to Listeria monocytogenes"			
Labels	hazard: listeria monocytogenes	hazard-category:	biological
	product: salads	product-category:	fruits and vegetables
"Create Common Good Recalls Jambalaya Products Due To Misbranding and Undeclared Allergens "			
Labels	hazard: milk and products thereof	hazard-category:	allergens
	product: meat preparations	product-category:	meat, egg and dairy products
"Revised Guan's Mushroom Co Recalls Enoki Because of Possible Health Risk "			
Labels	hazard: listeria monocytogenes	hazard-category:	biological
	product: mushrooms	product-category:	fruits and vegetables
"Nestlé Prepared Foods Recalls Lean Cuisine Baked Chicken Meal Products Due to Possible Foreign Matter Contamination"			
Labels	hazard: plastic fragment	hazard-category:	foreign bodies
	product: cooked chicken	product-category:	prepared dishes and snacks
"Undeclared allergen (sulphur dioxide and sorbic acid) found in prepackaged date sample"			
Labels	hazard: too high content of e 200 - sorbic acid	hazard-category:	food additives and flavourings
	product: dates	product-category:	fruits and vegetables

Figure 6: Some labeled sample texts. Colored spans signify the spans in `hazard-title` and `product-title`

	Per Class		Support Based			Total	
	most freq.	least freq.	$\mathcal{C}_{high}$	$\mathcal{C}_{medium}$	$\mathcal{C}_{low}$		
hazard-category	biological	migration					
By Year 1994 - 1998	34 (33.6)		34 (33.6)	4 (71.0)	20 (34.6)	58	(36.5)
1999 - 2002	50 (50.8)		50 (50.8)	21 (52.9)	59 (46.0)	130	(49.0)
2003 - 2006	53 (52.8)		53 (52.8)	61 (60.5)	78 (58.1)	192	(57.4)
2007 - 2010	158 (95.3)		158 (95.3)	49 (81.0)	112 (72.5)	319	(85.1)
2011 - 2014	308 (91.6)		308 (91.6)	293 (81.2)	233 (71.9)	834	(82.4)
2015 - 2018	911 (91.5)		911 (91.5)	1048 (86.8)	726 (82.2)	2685	(87.1)
2019 - 2022	1043 (85.7)	14 (44.1)	1043 (85.7)	1051 (89.0)	1234 (83.7)	3328	(86.0)
By Language DE	278 (58.8)	10 (29.0)	278 (58.8)	96 (52.7)	514 (58.4)	888	(57.9)
DK				1 (42.0)		1	(42.0)
EN	2278 (90.5)	4 (81.8)	2278 (90.5)	2426 (87.4)	1940 (85.2)	6644	(87.8)
FR				1 (73.0)	7 (71.7)	8	(71.9)
GR				3 (74.0)	1 (47.0)	4	(67.2)
IT	1 (51.0)		1 (51.0)			1	(51.0)
Total	2557 (87.0)	14 (44.1)	2557 (87.0)	2527 (86.0)	2462 (79.5)	7546	(84.2)
product-category	meat, egg and dairy products	sugars and syrups					
By Year 1994 - 1998	43 (32.6)		48 (34.9)	5 (46.2)	5 (41.8)	58	(36.5)
1999 - 2002	37 (48.6)		58 (47.6)	29 (47.5)	43 (51.8)	130	(49.0)
2003 - 2006	50 (55.8)		89 (56.0)	51 (59.0)	52 (58.1)	192	(57.4)
2007 - 2010	97 (76.6)		129 (75.2)	70 (67.7)	120 (105.8)	319	(85.1)
2011 - 2014	312 (76.8)	1 (58.0)	398 (78.0)	222 (83.3)	214 (89.8)	834	(82.4)
2015 - 2018	825 (87.1)	2 (141.0)	1178 (87.7)	793 (88.6)	714 (84.5)	2685	(87.1)
2019 - 2022	751 (83.0)	3 (132.7)	1189 (84.6)	1102 (87.1)	1037 (86.5)	3328	(86.0)
By Language DE	250 (63.7)		353 (63.1)	260 (56.9)	275 (52.3)	888	(57.9)
DK			1 (42.0)			1	(42.0)
EN	1862 (83.5)	6 (123.0)	2729 (84.8)	2008 (89.2)	1907 (90.6)	6644	(87.8)
FR	3 (76.3)		3 (76.3)	4 (65.2)	1 (85.0)	8	(71.9)
GR			3 (74.0)		1 (47.0)	4	(67.2)
IT					1 (51.0)	1	(51.0)
Total	2115 (81.1)	6 (123.0)	3089 (82.3)	2272 (85.4)	2185 (85.8)	7546	(84.2)
hazard	salmonella	rhodamine b					
By Year 1994 - 1998	3 (25.0)		16 (32.4)	23 (35.4)	19 (41.3)	58	(36.5)
1999 - 2002	6 (58.7)		33 (48.3)	46 (49.1)	51 (49.2)	130	(49.0)
2003 - 2006	13 (56.8)		47 (57.9)	82 (59.0)	63 (54.9)	192	(57.4)
2007 - 2010	96 (105.2)		138 (97.9)	83 (72.3)	98 (77.9)	319	(85.1)
2011 - 2014	143 (104.6)		315 (92.5)	310 (76.8)	209 (75.7)	834	(82.4)
2015 - 2018	288 (87.1)	1 (95.0)	1054 (92.5)	914 (85.0)	717 (82.1)	2685	(87.1)
2019 - 2022	393 (84.2)		1065 (87.9)	1006 (85.6)	1257 (84.8)	3328	(86.0)
By Language DE	130 (52.7)		215 (57.0)	264 (60.6)	409 (56.7)	888	(57.9)
DK			1 (42.0)			1	(42.0)
EN	812 (95.5)	1 (95.0)	2450 (92.3)	2193 (84.3)	2001 (86.1)	6644	(87.8)
FR				6 (76.2)	2 (59.0)	8	(71.9)
GR			2 (80.0)	1 (62.0)	1 (47.0)	4	(67.2)
IT					1 (51.0)	1	(51.0)
Total	942 (89.6)	1 (95.0)	2668 (89.4)	2464 (81.8)	2414 (81.1)	7546	(84.2)
product	ice cream	canned meat stew					
By Year 1994 - 1998	3 (48.3)		34 (37.0)	18 (36.9)	6 (32.3)	58	(36.5)
1999 - 2002	8 (60.0)		62 (49.5)	36 (49.3)	32 (47.4)	130	(49.0)
2003 - 2006	6 (61.7)		63 (59.3)	71 (57.1)	58 (55.7)	192	(57.4)
2007 - 2010	69 (120.7)		163 (97.4)	81 (70.1)	75 (74.6)	319	(85.1)
2011 - 2014	45 (120.8)		331 (92.2)	261 (75.0)	242 (77.1)	834	(82.4)
2015 - 2018	64 (100.0)		930 (88.3)	861 (87.4)	894 (85.7)	2685	(87.1)
2019 - 2022	57 (97.4)	1 (90.0)	925 (90.6)	1227 (84.4)	1176 (84.1)	3328	(86.0)
By Language DE	6 (93.5)		196 (63.9)	320 (55.4)	372 (57.0)	888	(57.9)
DK			1 (42.0)			1	(42.0)
EN	246 (106.3)	1 (90.0)	2309 (89.9)	2229 (86.4)	2106 (87.0)	6644	(87.8)
FR			2 (87.0)	1 (73.0)	5 (65.6)	8	(71.9)
GR			1 (62.0)	3 (69.0)		4	(67.2)
IT				1 (51.0)		1	(51.0)
Total	252 (106.0)	1 (90.0)	2508 (87.9)	2555 (82.4)	2483 (82.4)	7546	(84.2)

Table 4: Statistical overview of the data by prediction task. Each cell contains the number of samples in this category as well as the average text length of these samples in characters (in brackets). Empty cells indicate no samples in this category.

B Sample Prompts:

GPT-ALL:
We are looking for food hazards in texts. Please predict the correct class for the following sample: "CSM Bakery Solutions Voluntary Recalls 8" => allergens "CSM Bakery Solutions' Voluntary Recall of Cinnabon Stix® Due to Undeclared Peanut Allergen" => allergens "H-E-B Issues Voluntary Recall On Bakery Products" => biological "2009 - christie cookie recalls certain lots of peanut butter cookies due to expanded recall by peanut corporation of america" => biological "Gourmet Culinary Solutions Recalls Turkey Sausage Products Due To Possible Foreign Matter Contamination" => foreign bodies "Golden Natural Products Inc. Issues Allergy Alert on Undeclared Sulfites in Two Dried Apricot Products" => fraud "Texas Best Protein DBA Farm to Market Foods Issues Allergy Alert on Undeclared Peanut in Green Bean Casserole" => fraud "SPAR recalls its SPAR Chicken Tikka Chunks because it may contain small pieces of glass" => foreign bodies "Lactalis Nestlé Chilled Dairy UK recalls Nescafé Shakissimo Espresso Latte because of contamination with cleaning solution" => chemical "oriental packing co. inc. issues alert on lead in curry powder" => chemical "Garden Fortune Cookies" => migration "Green Field Farms Dairy Issues Voluntary Recall of Whole Chocolate Milk" => other hazard "Bakery Product - Labelling - Undeclared food additives (Preservative (211) and Colour (129) - 29 November 2010" => food additives and flavourings "FSIS Issues Public Health Alert for Ready-To-Eat Ham Product Due to Possible Processing Deviation" => other hazard "Pauls—Breaka chocolate UHT milk" => organoleptic aspects "Nestle—Billabong chocolate ice cream 8 pack" => organoleptic aspects "Undeclared allergen (sulphur dioxide and sorbic acid) found in prepackaged date sample" => food additives and flavourings "Heinz—Baked Beans in Tomato Sauce" => packaging defect "John West—Scottish sardines In vegetable oil" => packaging defect "Notos Schafskäse / Bulgarischer Schafskäse in Salzlake gereift 200g" => migration **"CSM Bakery Solutions Issues Allergy Alert on Undeclared Peanut in Chick-Fil-A Chocolate Chunk Cookies" =>**

GPT-SIM-k:
We are looking for food hazards in texts. Please predict the correct class for the following sample: $k = 1 \rightarrow$ "CSM Bakery Solutions Voluntary Recalls 8" => allergens $k = 2 \rightarrow$ "CSM Bakery Solutions' Voluntary Recall of Cinnabon Stix® Due to Undeclared Peanut Allergen" => allergens $k = 3 \rightarrow$ "H-E-B Issues Voluntary Recall On Bakery Products" => biological $k = 4 \rightarrow$ "2009 - christie cookie recalls certain lots of peanut butter cookies due to expanded recall by peanut corporation of america" => biological $k = 5 \rightarrow$ "Gourmet Culinary Solutions Recalls Turkey Sausage Products Due To Possible Foreign Matter Contamination" => foreign bodies $k = 6 \rightarrow$ "Golden Natural Products Inc. Issues Allergy Alert on Undeclared Sulfites in Two Dried Apricot Products" => fraud $k = 7 \rightarrow$ "Texas Best Protein DBA Farm to Market Foods Issues Allergy Alert on Undeclared Peanut in Green Bean Casserole" => fraud $k = 8 \rightarrow$ "SPAR recalls its SPAR Chicken Tikka Chunks because it may contain small pieces of glass" => foreign bodies $k = 9 \rightarrow$ "Lactalis Nestlé Chilled Dairy UK recalls Nescafé Shakissimo Espresso Latte because of contamination with cleaning solution" => chemical $k = 10 \rightarrow$ "oriental packing co. inc. issues alert on lead in curry powder" => chemical **"CSM Bakery Solutions Issues Allergy Alert on Undeclared Peanut in Chick-Fil-A Chocolate Chunk Cookies" =>**

GPT-MAX-k:
We are looking for food hazards in texts. Please predict the correct class for the following sample: "CSM Bakery Solutions Voluntary Recalls 8" => allergens $k = 1 \rightarrow$ "CSM Bakery Solutions' Voluntary Recall of Cinnabon Stix® Due to Undeclared Peanut Allergen" => allergens "Golden Natural Products Inc. Issues Allergy Alert on Undeclared Sulfites in Two Dried Apricot Products" => fraud $k = 2 \rightarrow$ "Texas Best Protein DBA Farm to Market Foods Issues Allergy Alert on Undeclared Peanut in Green Bean Casserole" => fraud "Heinz—Baked Beans in Tomato Sauce" => packaging defect $k = 3 \rightarrow$ "John West—Scottish sardines In vegetable oil" => packaging defect "Pauls—Breaka chocolate UHT milk" => organoleptic aspects $k = 4 \rightarrow$ "Nestle—Billabong chocolate ice cream 8 pack" => organoleptic aspects "Green Field Farms Dairy Issues Voluntary Recall of Whole Chocolate Milk" => other hazard $k = 5 \rightarrow$ "FSIS Issues Public Health Alert for Ready-To-Eat Ham Product Due to Possible Processing Deviation" => other hazard **"CSM Bakery Solutions Issues Allergy Alert on Undeclared Peanut in Chick-Fil-A Chocolate Chunk Cookies" =>**

GPT-CICLe:
We are looking for food hazards in texts. Please predict the correct class for the following sample: "CSM Bakery Solutions Voluntary Recalls 8" => allergens "CSM Bakery Solutions' Voluntary Recall of Cinnabon Stix® Due to Undeclared Peanut Allergen" => allergens **"CSM Bakery Solutions Issues Allergy Alert on Undeclared Peanut in Chick-Fil-A Chocolate Chunk Cookies" =>**

Table 5: Sample prompts taken from the hazard-category classification task. Example classes are color coded for better visibility. The order of the examples in GPT-ALL and GPT-SIM-k is identical, while in GPT-CICLe and GPT-MAX-k they are sorted by model certainty (most probable one first).

C   Model Specifics

C.1   Embeddings

As discussed in §4, we use two different types of embeddings for our classical models: *bag-of-words* (BOW) and *tf-idf* (TF-IDF). For the BOW representation,[9] we create a vector:

$$X_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,V}], \ \forall x_{i,v} \in [0, 1]$$

for each sample i, where V is the vocabulary size, and $x_{i,v}$ is the normalized count of occurrences of token v in sample i. For the TF-IDF-embedding, we calcluate

$$x_{i,v} = \text{tf}_{i,v} \cdot \text{idf}_v,$$

where $\text{tf}_{i,v}$ is the count of the samples' v-th token in the i-th training sample, and $\text{idf}_v = \ln \frac{N}{n_v}$ with n_v being the number of documents that contain token v. In order to make the above embeddings independent of the sample length we normalize X_i with the L2-norm in both cases. Text pre-processing comprises the application of a `TreebankWordTokenizer`, followed by `PorterStemmer` from the NLTK (Bird et al., 2009) Python package.

C.2   Traditional Classifiers

Model	Task	Parameters
BOW-KNN	hazard-category	$[k=2]\,(3\times) + [k=4]\,(2\times)$
	product-category	$[k=2]\,(4\times) + [k=4]\,(1\times)$
	hazard	$[k=2]\,(3\times) + [k=4]\,(2\times)$
	product	$[k=2]\,(3\times) + [k=8]\,(2\times)$
TFIDF-KNN	hazard-category	$[k=4]\,(1\times) + [k=8]\,(4\times)$
	product-category	$[k=2]\,(2\times) + [k=4]\,(2\times) + [k=8]\,(1\times)$
	hazard	$[k=2]\,(1\times) + [k=4]\,(1\times) + [k=8]\,(3\times)$
	product	$[k=2]\,(1\times) + [k=4]\,(1\times) + [k=8]\,(3\times)$
BOW-LR	hazard-category	$[\text{L1}, C=2.0]\,(5\times)$
	product-category	$[\text{L1}, C=2.0]\,(5\times)$
	hazard	$[\text{L1}, C=2.0]\,(5\times)$
	product	$[\text{L1}, C=2.0]\,(5\times)$
TFIDF-LR	hazard-category	$[\text{L1}, C=2.0]\,(5\times)$
	product-category	$[\text{L1}, C=2.0]\,(5\times)$
	hazard	$[\text{L1}, C=2.0]\,(5\times)$
	product	$[\text{L1}, C=2.0]\,(5\times)$
BOW-SVM	hazard-category	$[C=2.0]\,(5\times)$
	product-category	$[C=2.0]\,(5\times)$
	hazard	$[C=2.0]\,(5\times)$
	product	$[C=2.0]\,(5\times)$
TFIDF-SVM	hazard-category	$[C=1.0]\,(1\times) + [C=2.0]\,(4\times)$
	product-category	$[C=2.0]\,(5\times)$
	hazard	$[C=2.0]\,(5\times)$
	product	$[C=2.0]\,(5\times)$

Table 6: Parameters of Traditional Classifiers

As already discussed in §4, we optimze the models trained on each of our 5 CV splits on

---

[9]We use or own implementation for the representations.

a calibration set created using 10% holdout on the respective training data. We optimize...

... $k \in \{2, 4, 8\}$ neighbours for KNN

... $C \in \{0.5, 1.0, 2.0\}$ for both L1 and L2 for LR

... $C \in \{0.5, 1.0, 2.0\}$ for SVM

The best values found for each of the splits are reported in Table 6. Apart from KNN, we find stable sets of parameters for the best performance during grid search.

C.3   Encoder-only-Transformers
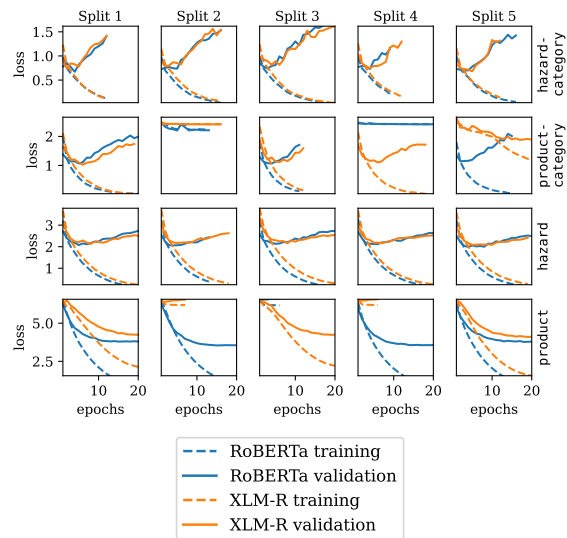
Figure 7: Learning curves of fine-tuning of RoBERTa and XLM-R averaged over 5 CV-splits.

For fine-tuning RoBERTa and XLM-R we use a learning rate of $5 \cdot 10^{-5}$ as well as a batch size of 16. Instead of grid searching for the best parameter we employ early stopping at the minimum validation loss with a patience of 5 epochs. As the average learning curves in Figure 7 show, `product-category` and `product` are harder to learn, as for some of the splits the curves show no progression. For the remaining models we reach the mininmum validation loss at an epoch of 5 on average.

C.4   CICLe

As we are aiming for a low set size, we select the α based on Figure 4. For `hazard-category` and `product-category`, we choose $\alpha = 0.05$ (meaning $p = 0.95$), as even for this high accuracy we get set sizes smaller than 10. For the remaining classification tasks we are limited by set length and choose $\alpha = 0.20$ for `hazard` and $\alpha = 0.40$ for `product`. This means, that for `product` we have

to accept a chance of at most 40% that the prediction set does not contain the ground truth in order to keep the set size low. This results in on average 2.6 classes per prompt in `hazard-category`, 5.3 classes per prompt in `product-category`, 4.7 classes per prompt in `hazard`, and 6.7 classes per prompt in `product`.

D Extended Results

In this appendix, we present additional results not included in the main paper. Tables 8 and 9 show additional metrics (*micro-*F_1, *macro*-precision, and *macro*-recall), as well as the bag-of-word (BOW) based classifiers. Table 7 compares *macro-*F_1 of RoBERTa and XLM-R on the two most represented languages in our data: English and German. We refer to §5 for a discussion of the aforementioned tables.

D.1 Theoretical boundaries on prompting performance

In Table 10 we present upper and lower bounds on the performance of our prompting strategies. In order to estimate the upper bounds (*-MAX), we assume that the LLM predicts the true class every time it is included in the prompt, and otherwise, the class of a random sample in the prompt. As is to be expected, the highest upper bound is always taken by the *-ALL prompts, as they include all classes in the training set. Note that for `hazard` and `product` the maximum scores are < 1, indicating that not all classes are present in the training data. This is a consequence of the low class support for some of the classes, which can not be divided into the five CV-splits. On our data, the reduction mostly affects the low-support classes, as they make up the bulk of the data but have a low number of tetxs per class. This means there is a higher possibility of completely removing them from the prompt compared to the other segments. Maximum performance on high support classes remains almost untouched by the reduction as they are very unlikely to be completely removed from the prompts.

For the lower bounds (MIN-*), we simply use the class of a random few-shot-sample in each prompt as the prediction. Here we assume that an LLM never predicts worse than random. The best minimum performance is delivered by CICLe most of the time, indicating that it produces short prompts with a high probability of containing the true class.

D.2 Analysis of prompting failures

We also provide a qualitative assessment of why prompting fails to produce a valid class label in some cases. Based on our experiments, we identified three reasons for failures:

1. Insufficient information in the sample for determining a class. In such cases, we found that the output of GPT explicitly states the problem. For example, consider the following texts and the corresponding LLM outputs for prompting on the `hazard` task:

text:	"Li Li Handmade Chicken and Pork Dumplings"
output:	"none (no food hazards mentioned)"
text:	"The Double Cola Company Recalls Select Cases of Its Cherry Ski Product"
output:	"none"
text:	"The Original Smoke and Spice Co Sea Salt Gourmet Seasoning"
output:	"none (no food hazards mentioned)"

None of the above examples contain a food hazard. GPT therefore states the lack of evidence which does not lead to a valid class label. We argue that in a real-world scenario, this is a strength rather than a weakness, as it can help avoid misclassification.

2. The text contains more than one possible class. In such cases, we found that the LLM produces a mix of labels:

text:	"Bravo Packing, Inc. Recalls All Performance Dog and Ground Beef Raw Pet Food Because of Possible Salmonella and Listeria Monocytogenes Health Risk to Humans and Animals"
class:	"listeria monocytogenes"
output:	"salmonella and listeria monocytogenes"
text:	"Exceptional Health Products Issues Allergy Alert On Undeclared Soy And Milk Allergens In Angel Wings 99-Daily Multi 120 Capsules"
class:	"milk and products thereof"
output:	"soybeans and products thereof, milk and products thereof"
text:	"Great Value brand Spaghetti Marinara recalled due to undeclared milk and sulphites"
class:	"milk and products thereof"
output:	"milk and products thereof, sulphur dioxide and sulphites"

The above examples are again for prompting on the `hazard` task. The texts contain multiple food hazards, which are combined by the LLM. Again, this can be seen as a strength of prompting rather than a weakness.

3. Too few few-shot-examples per class in the prompt. In Table 2 we see a high failure rate for the GPT-SIM-k prompts in product. A closer look at GPT's failed responses to these prompts shows the following:

text:	"New American Food Products LLC Issues Allergy Alert on Undeclared Milk in Packaged Goods Received From Our Supplier GKI Foods."
class:	"dark chocolate coated almonds"
output:	"dairy products"
text:	"Keats London Vegan Irish Cream Truffles 140g"
class:	"chocolate truffles"
output:	"vegan truffles"
text:	"New York Firm Recalls Boneless Veal Products Due To Possible E. Coli Contamination"
class:	"veal"
output:	"veal products"

In these cases, the LLM is not able to reproduce the exact label although it often finds the right product. We argue that because of the low number of samples per class, the LLM is not able to capture the context.

The above samples show, that prompting failures are not necessarily a sign of bad performance (see cases 1. and 2.). In cases where the model is unable to capture the context (e.g. case 3.), better prompt design can help reduce the failure rate.

D.3 Confusion Matrices:

In order to further analyze how prompting performs compared to other classifiers, we plotted confusion matrices of the best non-prompting-classifier (TF-IDF-LR) and GPT-CICLe in Figure 8. To achieve a better representation, we grouped classes based on the support zones introduced in the paper.

The figure shows, that while GPT-CICLe is generally worse than TF-IDF-LR in correctly predicting classes with high support, it performs much better on classes with low support. The latter are usually a problem for ML classifiers, as the impact of the few samples that are seen during training is usually overruled by the more supported classes.

TF-IDF-LR is generally more likely to wrongly predict samples to classes with higher support. This is natural, as we do not apply rebalancing techniques during training. For GPT-CICLe, however, this behavior is less evident. We surmise, that the LLM's lower need for training samples is able to balance out this shortcoming of TF-IDF-LR which is used as a basis of GPT-CICLe.
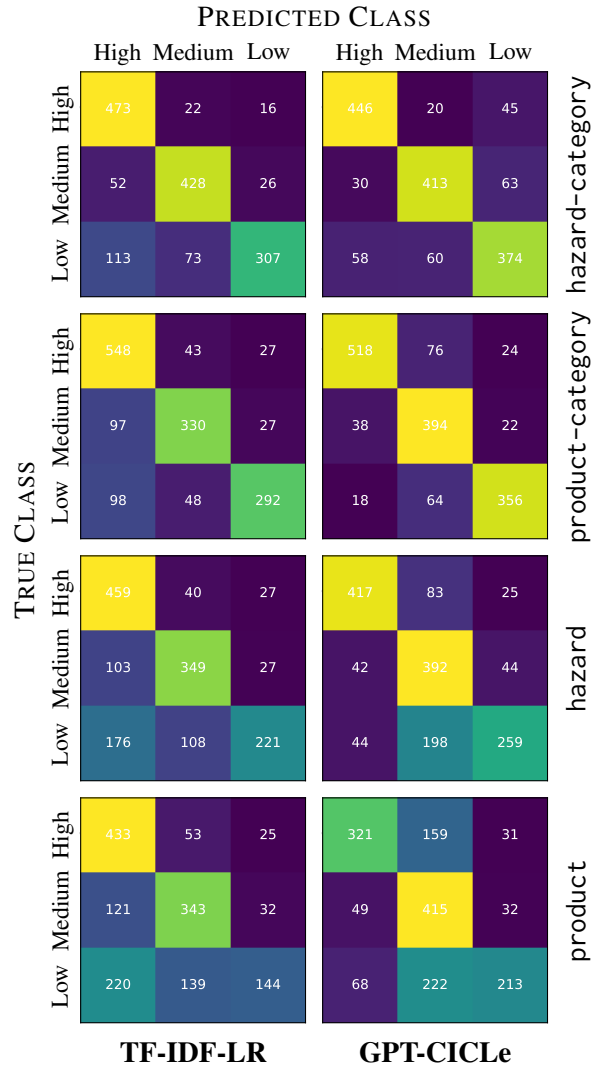
Figure 8: Confusion Matrices for all four prediction tasks. Classes were grouped by class support as shown in Figure 2.

Model	English		German	
	F_1 *(micro)*	F_1 *(micro)*	F_1 *(micro)*	F_1 *(macro)*
hazard-category				10 *classes*
RoBERTa	**0.83** ±0.01	**0.57** ±0.03	**0.39** ±0.03	0.13 ±0.02
XLM-R	0.82 ±0.01	0.53 ±0.02	0.37 ±0.06	**0.16** ±0.04
product-category				22 *classes*
RoBERTa	0.56 ±0.22	0.31 ±0.23	0.41 ±0.10	0.15 ±0.10
XLM-R	**0.61** ±0.17	**0.37** ±0.21	**0.47** ±0.13	**0.20** ±0.11
hazard				79 / 261 *classes*
RoBERTa	**0.65** ±0.02	**0.38** ±0.03	0.26 ±0.03	**0.06** ±0.00
XLM-R	0.64 ±0.03	0.36 ±0.04	**0.26** ±0.01	0.05 ±0.00
product				184 / 1256 *classes*
RoBERTa	**0.53** ±0.19	**0.43** ±0.17	**0.36** ±0.14	**0.10** ±0.04
XLM-R	0.39 ±0.28	0.28 ±0.22	0.25 ±0.18	0.06 ±0.05

Table 7: Average model F_1 and mean deviation over 5 CV-splits on texts in the two most common languages. Blue cells highlight the best score per column and language. For hazard and product we report performance on well-supported classes only.

Model	Scores (all classes)				Scores ($\mathcal{C}_{high}$)				Scores ($\mathcal{C}_{low}$)			
	F_1 (micro)	F_1	Precision	Recall	F_1 (micro)	F_1	Precision	Recall	F_1 (micro)	F_1	Precision	Recall
hazard-category												10 *classes*
RANDOM	0.34 ±0.01	0.08 ±0.00	0.07 ±0.00	0.10 ±0.00	0.50 ±0.01	0.33 ±0.00	0.50 ±0.00	0.25 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
MAJORITY	0.34 ±0.00	0.05 ±0.00	0.03 ±0.00	0.10 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	1.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
BOW-KNN	0.73 ±0.01	0.46 ±0.01	0.64 ±0.03	0.41 ±0.01	0.86 ±0.01	0.46 ±0.00	0.50 ±0.00	0.43 ±0.01	0.57 ±0.02	0.39 ±0.06	0.70 ±0.04	0.29 ±0.01
BOW-LR	0.76 ±0.01	0.52 ±0.02	0.67 ±0.01	0.47 ±0.02	0.91 ±0.01	0.48 ±0.00	0.50 ±0.00	0.45 ±0.00	0.64 ±0.01	0.47 ±0.03	0.71 ±0.02	0.36 ±0.03
BOW-SVM	0.79 ±0.01	0.57 ±0.03	0.66 ±0.02	0.53 ±0.03	0.90 ±0.02	0.47 ±0.01	0.50 ±0.00	0.45 ±0.01	0.69 ±0.01	0.53 ±0.03	0.71 ±0.04	0.44 ±0.03
TF-IDF-KNN	0.70 ±0.01	0.47 ±0.02	0.64 ±0.02	0.41 ±0.02	0.82 ±0.01	0.45 ±0.00	0.50 ±0.00	0.41 ±0.01	0.56 ±0.03	0.41 ±0.03	0.69 ±0.03	0.30 ±0.03
TF-IDF-LR	0.78 ±0.01	0.55 ±0.01	0.67 ±0.02	0.49 ±0.02	0.91 ±0.01	0.48 ±0.00	0.50 ±0.00	0.46 ±0.00	0.65 ±0.01	0.50 ±0.02	0.72 ±0.03	0.39 ±0.02
TF-IDF-SVM	0.79 ±0.01	0.58 ±0.01	0.64 ±0.01	0.55 ±0.01	0.88 ±0.01	0.47 ±0.00	0.50 ±0.00	0.44 ±0.01	0.70 ±0.01	0.56 ±0.01	0.71 ±0.04	0.47 ±0.01
RoBERTa	0.78 ±0.01	0.51 ±0.03	0.64 ±0.02	0.47 ±0.03	0.89 ±0.03	0.47 ±0.01	0.50 ±0.00	0.44 ±0.02	0.65 ±0.04	0.45 ±0.04	0.66 ±0.04	0.36 ±0.05
XLM-R	0.77 ±0.01	0.49 ±0.02	0.61 ±0.03	0.45 ±0.03	0.90 ±0.02	0.47 ±0.01	0.50 ±0.00	0.45 ±0.01	0.64 ±0.03	0.42 ±0.04	0.62 ±0.04	0.34 ±0.04
GPT-ALL	0.76 ±0.01	0.56 ±0.03	0.62 ±0.03	0.56 ±0.04	0.82 ±0.02	0.45 ±0.01	0.41 ±0.01	0.50 ±0.00	0.68 ±0.02	0.57 ±0.05	0.57 ±0.04	0.62 ±0.06
product-category												22 *classes*
RANDOM	0.08 ±0.00	0.01 ±0.00	0.01 ±0.00	0.05 ±0.00	0.22 ±0.01	0.20 ±0.01	0.17 ±0.01	0.26 ±0.01	0.03 ±0.00	0.01 ±0.00	0.00 ±0.00	0.04 ±0.00
MAJORITY	0.13 ±0.00	0.01 ±0.00	0.01 ±0.00	0.05 ±0.00	0.32 ±0.00	0.24 ±0.00	0.16 ±0.00	0.50 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
BOW-KNN	0.50 ±0.01	0.35 ±0.03	0.40 ±0.03	0.35 ±0.02	0.76 ±0.01	0.73 ±0.02	0.83 ±0.02	0.65 ±0.02	0.52 ±0.03	0.38 ±0.03	0.54 ±0.02	0.31 ±0.03
BOW-LR	0.69 ±0.01	0.49 ±0.03	0.59 ±0.03	0.45 ±0.02	0.89 ±0.01	0.86 ±0.02	0.94 ±0.01	0.80 ±0.02	0.69 ±0.02	0.46 ±0.04	0.65 ±0.05	0.37 ±0.03
BOW-SVM	0.70 ±0.01	0.54 ±0.03	0.61 ±0.04	0.50 ±0.03	0.89 ±0.01	0.86 ±0.01	0.94 ±0.01	0.79 ±0.01	0.72 ±0.02	0.52 ±0.05	0.70 ±0.05	0.44 ±0.05
TF-IDF-KNN	0.57 ±0.02	0.42 ±0.03	0.49 ±0.05	0.39 ±0.02	0.81 ±0.01	0.78 ±0.01	0.87 ±0.01	0.71 ±0.01	0.60 ±0.02	0.43 ±0.04	0.62 ±0.06	0.35 ±0.03
TF-IDF-LR	0.71 ±0.01	0.52 ±0.02	0.61 ±0.03	0.48 ±0.02	0.90 ±0.00	0.87 ±0.01	0.94 ±0.01	0.81 ±0.01	0.72 ±0.02	0.49 ±0.04	0.68 ±0.05	0.40 ±0.04
TF-IDF-SVM	0.73 ±0.01	0.59 ±0.01	0.64 ±0.02	0.56 ±0.02	0.89 ±0.01	0.86 ±0.01	0.95 ±0.01	0.79 ±0.01	0.77 ±0.01	0.59 ±0.01	0.76 ±0.02	0.52 ±0.02
RoBERTa	0.54 ±0.21	0.30 ±0.22	0.33 ±0.25	0.30 ±0.20	0.81 ±0.14	0.71 ±0.30	0.68 ±0.14		0.42 ±0.34	0.26 ±0.21	0.34 ±0.27	0.23 ±0.18
XLM-R	0.59 ±0.16	0.35 ±0.20	0.37 ±0.22	0.35 ±0.18	0.85 ±0.07	0.78 ±0.15	0.83 ±0.20	0.74 ±0.11	0.49 ±0.25	0.30 ±0.20	0.39 ±0.25	0.27 ±0.17
GPT-ALL	0.75 ±0.01	0.69 ±0.02	0.70 ±0.02	0.74 ±0.02	0.86 ±0.00	0.84 ±0.01	0.74 ±0.01	0.99 ±0.00	0.80 ±0.01	0.73 ±0.02	0.68 ±0.02	0.83 ±0.02
hazard												261 *classes*
RANDOM	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
MAJORITY	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
BOW-KNN	0.45 ±0.01	0.10 ±0.01	0.12 ±0.01	0.11 ±0.01	0.76 ±0.02	0.75 ±0.02	0.88 ±0.01	0.67 ±0.03	0.29 ±0.02	0.09 ±0.01	0.12 ±0.02	0.09 ±0.01
BOW-LR	0.56 ±0.02	0.13 ±0.01	0.15 ±0.01	0.12 ±0.01	0.84 ±0.02	0.84 ±0.02	0.89 ±0.01	0.79 ±0.02	0.41 ±0.02	0.11 ±0.01	0.14 ±0.01	0.10 ±0.01
BOW-SVM	0.54 ±0.01	0.07 ±0.00	0.08 ±0.00	0.07 ±0.00	0.85 ±0.01	0.85 ±0.01	0.90 ±0.01	0.81 ±0.01	0.31 ±0.01	0.04 ±0.01	0.06 ±0.00	0.04 ±0.00
TF-IDF-KNN	0.43 ±0.02	0.10 ±0.01	0.12 ±0.01	0.11 ±0.01	0.71 ±0.03	0.71 ±0.03	0.83 ±0.02	0.63 ±0.04	0.31 ±0.00	0.09 ±0.01	0.13 ±0.00	0.09 ±0.01
TF-IDF-LR	0.57 ±0.02	0.15 ±0.01	0.17 ±0.01	0.15 ±0.01	0.83 ±0.01	0.83 ±0.01	0.89 ±0.01	0.79 ±0.02	0.44 ±0.02	0.13 ±0.01	0.17 ±0.01	0.12 ±0.01
TF-IDF-SVM	0.54 ±0.01	0.07 ±0.00	0.07 ±0.00	0.07 ±0.00	0.84 ±0.02	0.84 ±0.02	0.91 ±0.01	0.78 ±0.02	0.32 ±0.01	0.04 ±0.00	0.06 ±0.00	0.04 ±0.00
RoBERTa	0.57 ±0.02	0.11 ±0.01	0.12 ±0.01	0.12 ±0.01	0.84 ±0.01	0.85 ±0.01	0.91 ±0.01	0.79 ±0.02	0.40 ±0.03	0.09 ±0.01	0.10 ±0.01	0.09 ±0.01
XLM-R	0.56 ±0.02	0.10 ±0.01	0.11 ±0.01	0.11 ±0.01	0.83 ±0.01	0.83 ±0.01	0.90 ±0.01	0.78 ±0.02	0.38 ±0.04	0.08 ±0.01	0.10 ±0.01	
product												1256 *classes*
RANDOM	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
MAJORITY	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
BOW-KNN	0.23 ±0.00	0.05 ±0.01	0.06 ±0.01	0.06 ±0.01	0.47 ±0.03	0.40 ±0.03	0.59 ±0.04	0.34 ±0.04	0.12 ±0.03	0.03 ±0.01	0.03 ±0.01	0.03 ±0.01
BOW-LR	0.41 ±0.01	0.09 ±0.00	0.09 ±0.00	0.11 ±0.00	0.71 ±0.01	0.69 ±0.03	0.79 ±0.03	0.65 ±0.02	0.18 ±0.02	0.03 ±0.00	0.04 ±0.00	0.04 ±0.00
BOW-SVM	0.19 ±0.01	0.01 ±0.00	0.01 ±0.00	0.02 ±0.00	0.55 ±0.01	0.47 ±0.02	0.62 ±0.03	0.45 ±0.02	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
TF-IDF-KNN	0.28 ±0.02	0.07 ±0.00	0.08 ±0.00	0.08 ±0.00	0.51 ±0.05	0.46 ±0.05	0.64 ±0.03	0.40 ±0.05	0.16 ±0.01	0.04 ±0.00	0.04 ±0.00	0.04 ±0.00
TF-IDF-LR	0.42 ±0.01	0.10 ±0.00	0.10 ±0.00	0.12 ±0.00	0.71 ±0.01	0.68 ±0.02	0.79 ±0.02	0.63 ±0.02	0.22 ±0.02	0.05 ±0.00	0.05 ±0.00	0.05 ±0.00
TF-IDF-SVM	0.20 ±0.01	0.01 ±0.00	0.01 ±0.00	0.02 ±0.00	0.59 ±0.02	0.50 ±0.01	0.61 ±0.02	0.49 ±0.01	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
RoBERTa	0.36 ±0.13	0.09 ±0.03	0.08 ±0.03	0.10 ±0.04	0.61 ±0.21	0.56 ±0.22	0.63 ±0.25	0.52 ±0.20	0.17 ±0.07	0.04 ±0.01	0.04 ±0.02	0.04 ±0.02
XLM-R	0.25 ±0.18	0.04 ±0.04	0.04 ±0.03	0.06 ±0.04	0.48 ±0.33	0.44 ±0.35	0.48 ±0.38	0.42 ±0.32	0.06 ±0.05	0.01 ±0.01	0.01 ±0.01	0.01 ±0.01
hazard (*well-supported only*)												79 *classes*
RANDOM	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.01	0.00 ±0.00	0.00 ±0.00	0.01 ±0.01
MAJORITY	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.02 ±0.00
BOW-KNN	0.48 ±0.02	0.27 ±0.01	0.33 ±0.02	0.28 ±0.02	0.76 ±0.02	0.75 ±0.02	0.88 ±0.01	0.67 ±0.03	0.35 ±0.02	0.26 ±0.02	0.39 ±0.04	0.23 ±0.02
BOW-LR	0.59 ±0.02	0.38 ±0.02	0.47 ±0.02	0.36 ±0.02	0.84 ±0.02	0.84 ±0.02	0.89 ±0.01	0.79 ±0.02	0.48 ±0.02	0.36 ±0.03	0.51 ±0.03	0.31 ±0.03
BOW-SVM	0.58 ±0.01	0.24 ±0.00	0.29 ±0.01	0.22 ±0.00	0.85 ±0.01	0.85 ±0.01	0.90 ±0.01	0.81 ±0.01	0.38 ±0.01	0.17 ±0.01	0.25 ±0.02	0.14 ±0.01
TF-IDF-KNN	0.46 ±0.02	0.28 ±0.01	0.34 ±0.02	0.29 ±0.01	0.71 ±0.03	0.71 ±0.03	0.83 ±0.02	0.63 ±0.04	0.36 ±0.01	0.28 ±0.01	0.41 ±0.02	0.25 ±0.02
TF-IDF-LR	0.60 ±0.02	0.41 ±0.02	0.49 ±0.03	0.38 ±0.02	0.83 ±0.01	0.83 ±0.01	0.89 ±0.01	0.79 ±0.02	0.50 ±0.02	0.39 ±0.03	0.54 ±0.04	0.34 ±0.02
TF-IDF-SVM	0.58 ±0.01	0.23 ±0.00	0.26 ±0.01	0.23 ±0.00	0.84 ±0.02	0.84 ±0.02	0.91 ±0.01	0.78 ±0.02	0.39 ±0.01	0.17 ±0.01	0.23 ±0.01	0.15 ±0.01
RoBERTa	0.61 ±0.02	0.36 ±0.03	0.41 ±0.04	0.36 ±0.03	0.84 ±0.01	0.85 ±0.01	0.91 ±0.01	0.79 ±0.02	0.49 ±0.04	0.32 ±0.04	0.40 ±0.05	0.31 ±0.03
XLM-R	0.60 ±0.02	0.33 ±0.04	0.39 ±0.02	0.33 ±0.04	0.83 ±0.01	0.83 ±0.01	0.90 ±0.01	0.78 ±0.02	0.46 ±0.03	0.29 ±0.04	0.38 ±0.03	0.27 ±0.04
product (*well-supported only*)												184 *classes*
RANDOM	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
MAJORITY	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.01 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
BOW-KNN	0.34 ±0.01	0.23 ±0.01	0.29 ±0.01	0.24 ±0.01	0.47 ±0.03	0.40 ±0.03	0.59 ±0.04	0.34 ±0.04	0.28 ±0.09	0.20 ±0.08	0.22 ±0.08	0.19 ±0.09
BOW-LR	0.58 ±0.02	0.48 ±0.02	0.53 ±0.02	0.49 ±0.02	0.71 ±0.01	0.69 ±0.03	0.79 ±0.03	0.65 ±0.02	0.46 ±0.10	0.34 ±0.08	0.36 ±0.07	0.33 ±0.08
BOW-SVM	0.29 ±0.01	0.09 ±0.00	0.12 ±0.01	0.11 ±0.01	0.55 ±0.01	0.47 ±0.02	0.62 ±0.03	0.45 ±0.02	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
TF-IDF-KNN	0.40 ±0.03	0.32 ±0.02	0.38 ±0.01	0.33 ±0.03	0.51 ±0.05	0.46 ±0.05	0.64 ±0.03	0.40 ±0.05	0.33 ±0.07	0.24 ±0.04	0.27 ±0.04	0.23 ±0.04
TF-IDF-LR	0.59 ±0.01	0.51 ±0.01	0.56 ±0.01	0.52 ±0.02	0.71 ±0.01	0.68 ±0.02	0.79 ±0.02	0.63 ±0.02	0.57 ±0.05	0.43 ±0.07	0.45 ±0.07	0.42 ±0.06
TF-IDF-SVM	0.31 ±0.01	0.09 ±0.00	0.10 ±0.01	0.11 ±0.00	0.59 ±0.02	0.50 ±0.01	0.61 ±0.02	0.49 ±0.01	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00	0.00 ±0.00
RoBERTa	0.51 ±0.19	0.43 ±0.17	0.45 ±0.18	0.45 ±0.18	0.61 ±0.21	0.56 ±0.22	0.63 ±0.25	0.52 ±0.20	0.37 ±0.19	0.28 ±0.17	0.32 ±0.20	0.26 ±0.15
XLM-R	0.37 ±0.27	0.28 ±0.22	0.28 ±0.23	0.30 ±0.24	0.48 ±0.33	0.44 ±0.35	0.48 ±0.38	0.42 ±0.32	0.20 ±0.20	0.14 ±0.15	0.16 ±0.16	0.13 ±0.14

Table 8: Average model F_1(*micro & macro*), Precision (*macro*), and Recall (*macro*) as well as mean deviation over 5 CV-splits. Blue cells highlight the best score per column and classification task. Bold scores are the best per column, classification task, and type of classifier (*naive, traditional, encoder-transformer, and prompting*).

Model	Scores (all classes)				Scores ($\mathcal{C}_{high}$)				Scores ($\mathcal{C}_{low}$)			
	F_1 (micro)	F_1 (macro)	Precision	Recall	F_1 (micro)	F_1 (macro)	Precision	Recall	F_1 (micro)	F_1 (macro)	Precision	Recall
hazard-category											10 classes; $\alpha = 0.05$	
RANDOM	0.33	0.08	0.07	0.10	0.51	0.34	0.50	0.25	0.00	0.00	0.00	0.00
MAJORITY	0.34	0.05	0.03	0.10	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
TF-IDF-LR	0.77	0.55	0.65	0.50	0.93	0.48	0.50	0.46	0.65	0.50	0.68	0.41
GPT-ALL	0.74	0.53	0.59	0.53	0.82	0.45	0.41	**0.50**	0.67	**0.54**	0.54	0.59
GPT-SIM-10	0.76	0.52	0.55	0.52	0.85	0.46	0.42	**0.50**	0.67	0.50	0.48	0.57
GPT-SIM-20	0.74	0.53	0.59	0.53	0.82	0.45	0.41	**0.50**	0.67	**0.54**	0.54	0.59
GPT-MAX-5	0.75	0.55	0.54	0.59	0.85	0.46	0.42	**0.50**	**0.69**	**0.54**	0.47	**0.65**
GPT-MAX-10	0.72	0.49	**0.63**	0.49	0.79	0.44	0.39	**0.50**	0.63	0.49	**0.59**	0.54
GPT-CICLe	**0.77**	**0.56**	0.53	**0.61**	**0.87**	**0.47**	**0.44**	**0.50**	0.68	0.53	0.46	**0.65**
product-category											22 classes; $\alpha = 0.05$	
RANDOM	0.08	0.01	0.01	0.05	0.23	0.21	0.17	0.28	0.03	0.00	0.00	0.03
MAJORITY	0.13	0.01	0.01	0.05	0.32	0.24	0.16	0.50	0.00	0.00	0.00	0.00
TF-IDF-LR	0.71	0.58	0.62	0.56	0.87	0.84	0.94	0.76	0.76	0.60	0.77	0.52
GPT-ALL	0.76	**0.72**	**0.72**	0.77	0.86	0.85	0.74	0.98	0.80	**0.75**	**0.71**	0.86
GPT-SIM-10	0.77	0.66	0.64	0.75	0.89	0.87	0.79	0.98	0.80	0.67	0.60	0.85
GPT-SIM-20	0.77	0.69	0.67	0.75	0.87	0.86	0.76	0.98	**0.81**	0.69	0.64	0.81
GPT-MAX-5	0.79	0.64	0.61	0.70	**0.90**	**0.89**	**0.81**	**0.99**	0.80	0.61	0.55	0.75
GPT-MAX-10	0.78	0.68	0.65	0.78	0.87	0.86	0.76	**0.99**	0.80	0.68	0.61	**0.88**
GPT-CICLe	**0.80**	0.69	0.65	**0.79**	**0.90**	0.88	0.80	0.98	**0.81**	0.68	0.61	**0.88**
hazard											261 classes; $\alpha = 0.20$	
RANDOM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00
MAJORITY	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.00
TF-IDF-LR	0.55	0.15	0.17	0.14	0.81	0.81	0.88	0.76	0.44	0.13	0.17	0.12
GPT-SIM-10	0.60	**0.18**	**0.18**	**0.20**	0.81	**0.81**	0.71	**0.93**	0.48	**0.18**	0.17	**0.21**
GPT-SIM-20	0.60	**0.18**	**0.18**	0.19	0.80	0.80	0.71	0.92	**0.50**	**0.18**	**0.18**	**0.21**
GPT-MAX-5	**0.61**	0.16	0.15	0.19	**0.82**	**0.81**	**0.73**	0.92	0.49	0.15	0.14	0.19
GPT-MAX-10	0.58	0.16	0.15	0.19	0.77	0.77	0.67	0.91	0.49	0.15	0.14	0.19
GPT-CICLe	0.59	0.16	0.15	0.18	0.79	0.79	0.70	0.91	0.49	0.15	0.14	0.19
product											1256 classes; $\alpha = 0.40$	
RANDOM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MAJORITY	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TF-IDF-LR	0.43	0.11	0.11	0.12	0.71	0.66	0.76	0.62	0.23	0.05	0.05	0.05
GPT-SIM-10	**0.52**	**0.18**	**0.18**	**0.20**	0.61	0.57	0.46	**0.85**	0.47	**0.13**	**0.12**	0.13
GPT-SIM-20	0.51	**0.18**	**0.18**	0.19	0.61	0.57	0.46	0.82	**0.48**	**0.13**	**0.12**	**0.14**
GPT-MAX-5	0.51	0.16	0.17	0.16	**0.68**	**0.65**	**0.57**	0.81	0.38	0.09	0.10	0.10
GPT-MAX-10	0.51	0.16	0.17	0.17	**0.68**	0.62	0.53	0.82	0.38	0.10	0.10	0.10
GPT-CICLe	**0.52**	0.16	0.17	0.17	0.66	0.61	0.52	0.81	0.39	0.10	0.10	0.10
hazard (well-supported only)											102 classes; $\alpha = 0.20$	
RANDOM	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.01
MAJORITY	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.01	0.00	0.00	0.01
TF-IDF-LR	0.57	0.37	0.45	0.35	0.81	0.81	0.88	0.76	0.48	0.36	0.48	0.31
GPT-SIM-10	**0.62**	**0.44**	0.42	**0.50**	0.81	**0.81**	0.71	**0.93**	0.52	**0.44**	0.40	**0.55**
GPT-SIM-20	**0.62**	**0.44**	**0.44**	0.49	0.80	0.80	0.71	0.92	**0.54**	**0.44**	**0.41**	0.54
GPT-MAX-5	**0.62**	0.42	0.39	0.49	**0.82**	**0.81**	**0.73**	0.92	0.53	0.40	0.35	0.53
GPT-MAX-10	0.59	0.42	0.39	0.49	0.77	0.77	0.66	0.91	0.53	0.41	0.36	0.53
GPT-CICLe	0.61	0.42	0.39	0.48	0.79	0.79	0.70	0.91	0.53	0.40	0.35	0.52
product (well-supported only)											257 classes; $\alpha = 0.40$	
RANDOM	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MAJORITY	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
TF-IDF-LR	0.57	0.47	0.50	0.49	0.71	0.66	0.76	0.62	0.58	0.41	0.44	0.40
GPT-SIM-10	0.59	0.54	0.52	0.62	0.61	0.57	0.46	**0.85**	0.71	0.58	0.56	0.61
GPT-SIM-20	0.59	0.53	0.51	0.62	0.61	0.57	0.46	0.82	0.72	0.59	0.57	0.64
GPT-MAX-5	0.62	0.56	0.57	0.62	**0.68**	**0.65**	**0.57**	0.81	0.70	0.56	0.55	0.60
GPT-MAX-10	**0.63**	0.57	0.57	0.63	**0.68**	0.62	0.53	0.82	0.72	0.58	0.57	0.62
GPT-CICLe	0.62	**0.58**	**0.58**	**0.65**	0.66	0.61	0.52	0.81	**0.73**	**0.61**	**0.60**	**0.65**

Table 9: Model F_1 (*micro & macro*), Precision (*macro*), and Recall (*macro*) on 20% holdout (*i.e. only calculated on the first CV-split to save resources*). Blue cells highlight the best score per column and category.

Model	Scores (*all classes*)				Scores ($\mathcal{C}_{high}$)				Scores ($\mathcal{C}_{low}$)			
	F_1 (*micro*)	F_1 (*macro*)	Precision	Recall	F_1 (*micro*)	F_1 (*macro*)	Precision	Recall	F_1 (*micro*)	F_1 (*macro*)	Precision	Recall
`hazard-category`											10 *classes;* $\alpha = 0.05$	
RANDOM	0.33	0.08	0.07	0.10	0.51	0.34	0.50	0.25	0.00	0.00	0.00	0.00
MAJORITY	0.34	0.05	0.03	0.10	1.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00
TF-IDF-LR	0.77	0.55	0.65	0.50	0.93	0.48	0.50	0.46	0.65	0.50	0.68	0.41
MIN-ALL	0.10	0.07	0.14	0.09	0.10	0.09	0.05	**0.50**	0.10	0.08	0.15	0.11
MIN-SIM-5	0.29	0.22	0.22	0.26	0.32	0.24	0.16	**0.50**	0.32	0.25	0.19	0.39
MIN-SIM-10	0.17	0.10	0.12	0.11	0.16	0.14	0.08	**0.50**	0.20	0.12	0.11	0.14
MIN-SIM-20	0.09	0.06	0.07	0.09	0.09	0.08	0.04	**0.50**	0.10	0.07	0.07	0.12
MIN-MAX-5	0.18	0.12	0.16	0.13	0.19	0.16	0.10	**0.50**	0.22	0.17	0.15	0.20
MIN-MAX-10	0.11	0.08	0.12	0.11	0.10	0.09	0.05	**0.50**	0.15	0.10	0.12	0.15
MIN-CICLe	**0.56**	**0.32**	**0.32**	**0.33**	**0.66**	**0.40**	**0.33**	0.50	**0.45**	**0.29**	**0.23**	**0.40**
MAX-ALL	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MAX-SIM-5	0.92	0.78	0.72	0.89	0.98	0.50	0.49	0.50	0.86	0.74	0.65	0.89
MAX-SIM-10	0.98	0.85	0.81	0.95	**1.00**	**1.00**	**1.00**	**1.00**	0.96	0.82	0.76	0.94
MAX-SIM-20	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MAX-MAX-5	0.98	0.79	0.74	0.88	**1.00**	0.50	0.50	0.50	0.95	0.74	0.67	0.86
MAX-MAX-10	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MAX-CICLe	0.96	0.76	0.71	0.87	0.99	0.50	0.50	0.50	0.92	0.71	0.64	0.85
`product-category`											22 *classes;* $\alpha = 0.05$	
MIN-ALL	0.05	0.03	0.04	0.05	0.08	0.09	0.05	0.49	0.06	0.04	0.04	0.07
MIN-SIM-5	0.22	0.16	0.17	0.16	0.37	**0.37**	0.25	**0.69**	0.26	0.22	0.16	0.38
MIN-SIM-10	0.14	0.09	0.09	0.08	0.25	0.25	0.16	0.58	0.18	0.10	0.08	0.19
MIN-SIM-20	0.09	0.06	0.06	0.06	0.16	0.17	0.10	0.56	0.10	0.07	0.05	0.11
MIN-MAX-5	0.19	**0.18**	**0.19**	**0.20**	0.28	0.27	0.18	0.56	0.26	0.22	0.18	0.40
MIN-MAX-10	0.10	0.06	0.06	0.07	0.21	0.20	0.12	0.61	0.10	0.06	0.05	0.11
MIN-CICLe	**0.23**	0.13	0.13	0.15	**0.40**	**0.37**	**0.27**	0.60	0.24	0.14	0.10	0.24
MAX-ALL	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
MAX-SIM-5	0.84	0.65	0.62	0.70	0.95	0.94	0.90	0.98	0.81	0.61	0.54	0.74
MAX-SIM-10	0.93	0.79	0.77	0.88	0.99	0.99	0.98	**1.00**	0.89	0.75	0.70	0.88
MAX-SIM-20	0.97	0.85	0.86	0.87	**1.00**	**1.00**	**1.00**	**1.00**	0.94	0.81	0.80	0.84
MAX-MAX-5	0.92	0.72	0.68	0.81	**1.00**	0.99	0.99	**1.00**	0.86	0.65	0.58	0.79
MAX-MAX-10	0.96	0.81	0.77	0.92	**1.00**	**1.00**	**1.00**	**1.00**	0.91	0.75	0.68	0.91
MAX-CICLe	0.95	0.80	0.76	0.91	**1.00**	**1.00**	**1.00**	**1.00**	0.91	0.75	0.68	0.91
`hazard`											261 *classes;* $\alpha = 0.20$	
MIN-ALL	0.01	0.00	0.01	0.01	0.01	0.01	0.00	0.28	0.01	0.01	0.01	0.01
MIN-SIM-5	0.19	0.06	**0.06**	0.07	0.36	0.36	0.27	0.54	0.17	**0.07**	**0.06**	0.09
MIN-SIM-10	0.11	0.03	0.04	0.04	0.23	0.22	0.15	0.45	0.08	0.03	0.03	0.04
MIN-SIM-20	0.06	0.02	0.02	0.03	0.13	0.13	0.08	0.43	0.04	0.03	0.02	0.03
MIN-MAX-5	0.15	0.05	0.04	0.07	0.29	0.29	0.20	0.51	0.19	0.05	0.04	0.09
MIN-MAX-10	0.08	0.03	0.03	0.05	0.16	0.16	0.10	0.48	0.10	0.03	0.02	0.07
MIN-CICLe	**0.33**	**0.07**	**0.06**	**0.09**	**0.60**	**0.60**	**0.50**	0.75	**0.25**	**0.07**	**0.06**	**0.11**
MAX-ALL	**0.98**	**0.46**	**0.47**	**0.45**	**1.00**	**1.00**	**1.00**	**1.00**	**0.95**	**0.45**	**0.46**	**0.44**
MAX-SIM-5	0.68	0.20	0.21	0.21	0.93	0.93	0.88	0.98	0.53	0.20	0.19	0.22
MAX-SIM-10	0.78	0.25	0.25	0.26	0.97	0.97	0.95	0.99	0.62	0.24	0.23	0.26
MAX-SIM-20	0.84	0.28	0.28	0.29	0.99	0.99	0.99	**1.00**	0.68	0.26	0.26	0.28
MAX-MAX-5	0.76	0.20	0.18	0.23	0.99	0.99	0.99	**1.00**	0.60	0.18	0.16	0.22
MAX-MAX-10	0.81	0.21	0.20	0.25	**1.00**	**1.00**	**1.00**	**1.00**	0.65	0.19	0.17	0.24
MAX-CICLe	0.78	0.20	0.19	0.24	0.99	0.99	0.99	**1.00**	0.62	0.18	0.16	0.23
`product`											1256 *classes;* $\alpha = 0.40$	
MIN-ALL	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
MIN-SIM-5	0.12	0.04	0.04	0.05	0.20	0.17	0.13	0.28	0.09	**0.02**	**0.02**	**0.03**
MIN-SIM-10	0.08	0.02	0.03	0.03	0.15	0.14	0.09	0.33	0.06	**0.02**	**0.02**	0.02
MIN-SIM-20	0.05	0.01	0.02	0.02	0.09	0.08	0.05	0.18	0.04	0.01	0.01	0.01
MIN-MAX-5	0.11	0.03	0.03	0.04	0.20	0.22	0.16	**0.45**	0.06	0.01	0.01	0.02
MIN-MAX-10	0.06	0.02	0.02	0.03	0.08	0.08	0.06	0.16	0.05	0.01	0.01	0.01
MIN-CICLe	**0.15**	**0.05**	**0.05**	**0.06**	**0.24**	**0.24**	**0.20**	0.39	**0.09**	**0.02**	**0.02**	0.02
MAX-ALL	**0.90**	**0.35**	**0.36**	**0.34**	**1.00**	**1.00**	**1.00**	**1.00**	**0.73**	**0.24**	**0.25**	**0.24**
MAX-SIM-5	0.49	0.16	0.17	0.16	0.72	0.66	0.58	0.81	0.38	0.10	0.10	0.11
MAX-SIM-10	0.58	0.18	0.20	0.19	0.81	0.76	0.69	0.88	0.41	0.12	0.12	0.12
MAX-SIM-20	0.65	0.21	0.23	0.22	0.87	0.82	0.77	0.91	0.48	0.14	0.14	0.14
MAX-MAX-5	0.59	0.17	0.17	0.17	0.85	0.82	0.78	0.91	0.33	0.08	0.08	0.08
MAX-MAX-10	0.64	0.18	0.19	0.19	0.90	0.86	0.83	0.92	0.37	0.09	0.09	0.09
MAX-CICLe	0.62	0.18	0.19	0.18	0.86	0.84	0.80	0.93	0.36	0.08	0.09	0.09

Table 10: Model F_1(*micro & macro*), Precision (*macro*), and Recall (*macro*) on 20% holdout (*i.e. only calculated on the first CV-split to save resources*). Blue cells highlight the best score per column and category. Rows named "MIN-[METHOD]" sigignify a lower bound on performance for prompting with [METHOD], while rows named "MAX-[METHOD]" signify an upper bound.