UniBuc at SemEval-2025 Task 9: Similarity Approaches to Classification

Marius Micluța-Câmpeanu
Interdisciplinary School of Doctoral Studies, University of Bucharest, Romania
marius.micluta-campeanu@unibuc.ro

Abstract

In this paper, we present a similarity-based method for explainable classification in the context of the SemEval 2025 Task 9: The Food Hazard Detection Challenge. Our proposed system is essentially unsupervised, leveraging the semantic properties of the labels. This approach brings some key advantages over typical classification systems. First, similarity metrics offer a more intuitive interpretation. Next, this technique allows for inference on novel labels. Finally, there is a non-negligible amount of ambiguous labels, so learning a direct mapping does not lead to meaningful representations.

Our team ranks 14th for the second sub-task. Our method is generic and can be applied to any classification task.

1 Introduction

As people become more aware of the health risks associated with the global food industry, there is a growing interest to ensure the safety and quality of these products. The complexity of modern food supply chains, which involve multiple stages of production, processing, and distribution, has made it increasingly difficult to monitor and control food safety hazards. In this context, natural language processing (NLP) tools lend themselves invaluable to identifying patterns and extracting information from heterogeneous data sources.

The main goal of the Food Hazard Detection Challenge is to explore the explainability of classification systems on texts associated with food safety risks. The dataset consists of food-incident reports collected from various sources written in English, representing a subset of a larger dataset created by the organizers that also includes texts in German and a few instances in four other languages (Randl et al., 2024). The first task requires systems to predict coarse-grained categories for hazards and products, while the second task is focused on determining the exact hazard and product (Randl et al., 2025). We participated only in the second sub-task to explore the viability of similarity methods.

The dataset reports contain a title, the full report text and other details such as date and country. Since the title does not always contain adequate information about products or hazards, we also add the full text as input to our system, ignoring the other features. Next, we use a large language model (LLM) to clean up this text. For classification, we employ cosine similarity between each clean text and every label (Schütze et al., 2008, p. 121) without any training step. In the case of hazards, we also apply lemmatization on each word and reorder the labels by importance and specificity (detailed in Section 3).

Beside data cleaning, our biggest challenge was to decide how to handle ambiguous labels. Even though this dataset design choice appears to be intentional (Randl et al., 2024), we believe that the addition of almost identical classes could have been avoided, at least to some extent. We settled for a similarity-based approach because this should aid the explainability of the system and it also eliminates the challenge of heavily imbalanced data.

We note that many of our wrong predictions were affected by label overlap. Our team ranked 14th out of 26 teams. Our system is unsupervised, employing small models and needing modest resources, while also allowing for easy interpretation. Our code is publicly available[1].

2 Related work

Document similarity Information retrieval relies extensively on cosine similarity for document classification (Chen et al., 2009). More recently, Schopf et al. (2023) propose baselines for unsupervised text classification and show that similarity-based approaches using sentence Transformers are

[1] https://github.com/mcmarius/SemEval-2025-Task-9

Proceedings of the The 19th International Workshop on Semantic Evaluation (SemEval-2025), pages 280–287

suitable for text classification of unseen classes and outperform zero-shot methods.

Label semantics Another line of work intended to overcome issues related to noisy data is label refinement, a process that aims to reduce ambiguity and uncertainty of labels. Song et al. (2023) provide an overview of the challenges associated with learning from noisy labels, underlining that robust learning methods used to overcome label noise are not designed to deal with extreme cases of data imbalance.

Chu et al. (2021) use k-means clustering of labels for dataless classification, showing improvements on unbalanced datasets and robustness in terms of label description choice. In their experiments, the datasets used have at most 20 classes. Huang et al. (2024) re-rank the predictions of hard samples based on similarity, while Dong et al. (2024) perform re-ranking with a separate model that receives refined labels.

To the best of our knowledge, our approach is different from existing techniques in the literature, as these previous efforts typically rely on a training step. They also either do not require fixed labels or do not perform label reordering.

3 System overview

Our system consists of two independent pipelines for hazard and product classification. Each pipeline has a data cleaning step, a pre-processing step and a classification step, optionally followed by a post-processing step, as shown in Figures 1 and 2. There is no training or fine-tuning involved.

3.1 Data cleaning

Since the title does not always specify the hazard or the product, we need to look up this information in the full article. This raw text may contain duplicate lines, HTML markup and a lot of instructions for consumers that are irrelevant for us. We want to remove as much noise as possible to provide downstream steps with useful data and to reduce the context size of LLM prompts.

First, we remove duplicate lines. For hazards, we apply custom-made regex rules, with a catch-all match that only keeps longer lines. Then, we remove HTML tags if present. Finally, we keep only the first 3000 characters. The resulting text preceded by the title is fed to a LLM that is tasked to extract the hazard or the product, including a short description of that entity. See Appendix A for the prompts used.

3.2 Pre-processing

In this stage, we remove task-specific stop-words by analyzing the classification mistakes, such as "product", "category", and "food". These words are not specific to a particular class, while others are artifacts of LLM extraction.

In the case of hazards, we also perform lemmatization on texts and labels, a common approach in information retrieval. Perhaps surprisingly, this did not lead to improved results for products. One possible explanation is due to the nature of product names (e.g. brand names) that might be less amenable to lemmatization, though this needs further investigation, since we also include product descriptions specifically to mitigate such issues.

3.3 Similarity classification

We model the classification task as a similarity search problem, assigning the label with the highest cosine similarity between the embedded text and the embedded label. The reason for also asking the LLM in Section 3.1 to include a description is to aid this search by including more common terms along commercial or highly specific names.

3.4 Post-processing

There are several instances where an incident report contains multiple distinct problems. We need to decide which hazard should be prioritized, since the task is modeled as single-label classification. The following example (training set, ID 120) can be characterized by both "allergens" ("walnut") and "fraud" ("mislabelled" – gold label) categories:

> Has been mislabelled and may contain walnuts which may pose a health risk to people allergic to walnuts.

In such a case, we first prioritize the category that poses a greater risk (allergens), since this is the most critical facet. Next, if there are multiple issues within the same category, we pick the most specific hazard ("walnut" instead of "allergens"), thus prioritizing risk over specificity. To determine this priority, we pre-compute these lists by sorting by importance and specificity using a LLM. This process is detailed in Appendix B.

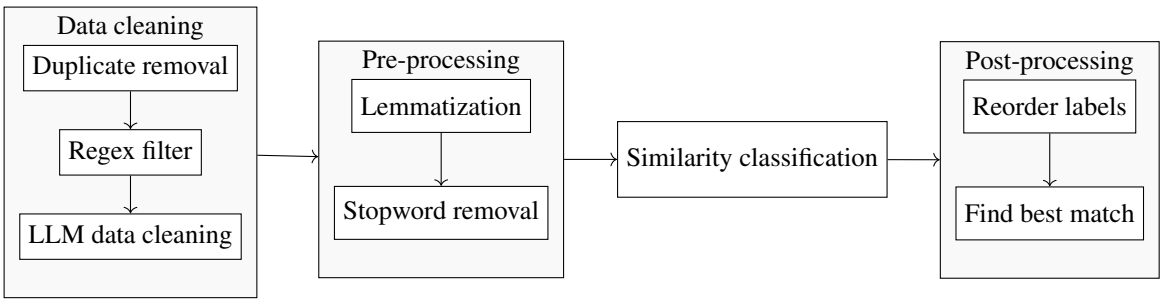


Figure 1: Hazard classification pipeline. Data cleaning and label sorting is performed using the same LLM. For similarity, texts and labels are encoded with the same model, different from the LLM.

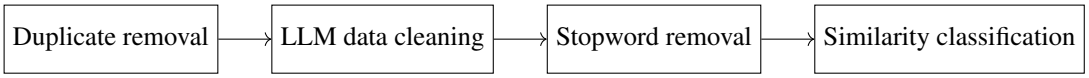


Figure 2: Product classification pipeline. Some components from the hazard pipeline are omitted here as they do not improve the predictions.

4 Experimental setup

Data cleaning and label sorting is achieved with a Llama 3.2-3B-Instruct model (Grattafiori et al., 2024), with 4-bit quantization (Q4_K_M) using ollama, restricting output to 50 tokens. The data cleaning process is the most time-consuming, requiring at least 4 hours for the training set. Ideally, this should be a one time effort. This is necessary regardless of how we decide to implement the classification, so we have to verify that the hazards and products are extracted appropriately.

Similarities are computed by embedding the texts and the labels with the help of Sentence Transformers models (Reimers and Gurevych, 2019; Li et al., 2023). We choose different embeddings for hazards[2] and products[3], observing that larger models do not always lead to improved scores. For lemmatization we use simplemma[4].

The classification step is efficient, taking less than a minute for the entire training set of 5082 examples, which enabled us to conduct several experiments. Comparatively, a single call to a LLM takes a few seconds.

5 Results

The task organizers propose a custom scoring function based on F1-score that favors predicting hazards, taking into account product predictions only for examples with correct hazard detection. Our team[5] obtained a score of 0.3453 for the second sub-task, having 0.4257 F1-score for 128 hazards and 0.2528 F1-score for 1142 products. With these results, we are the 14th team out of 26 contestants.

Component	Train	Validation	Test
Data cleaning	42.68	38.55	41.22
+ 1) lemma	41.23	**43.32**	41.90
+ 2) stop words	43.22	37.79	40.88
+ 3) sort w/ cat	41.82	33.86	40.39
+ 4) sort w/o cat	42.08	34.17	40.52
+ 5) predict cat	42.68	38.55	24.08
+ 1), 2) *	41.81	<u>42.95</u>	**42.57**
+ 1), 2), 3)	<u>45.17</u>	41.27	41.93
+ 1), 2), 4)	44.72	41.27	<u>42.07</u>
+ 2), 3)	**46.33**	36.69	40.32

Table 1: Hazard classification F1-scores. Best result in bold, second-best result underlined. The asterisk indicates the final submission.

5.1 Hazard classification

We concentrate most of our efforts on hazard classification since this is the main goal of the task. Our system is designed to aid with the interpretability of results. We show the impact of each component, summarized in Table 1.

Lemmatization This pre-processing step aims to reduce word variation, which should increase similarities with related words. This improves predictions, although it usually has to be combined with other techniques.

[2] sentence-transformers/paraphrase-MiniLM-L6-v2
[3] thenlper/gte-large
[4] https://github.com/adbar/simplemma
[5] CodaLab username: marius.micluta-campeanu

Stop word removal The motivation for removing stop words is that we are only interested in comparing content words for relatedness. This is also a feature that is helpful most of the time.

Label sorting This post-processing step gives us mixed results. We use this step in two settings, by sorting with or without taking category information into account. Despite obtaining slightly lower scores when we include the category, we believe this to be better suited for real-world applications, allowing experts to focus on the highest risks.

Category prediction We attempt to lower the number of candidate classes by first determining the appropriate category, also through similarities. Since the category names are semantically more distant from concrete instances of hazards, we risk to exclude the true category. This hypothesis is confirmed empirically as results either stay the same or worsen.

While category prediction could be delegated to a trainable classifier (for example, a conformal base classifier as suggested by the task organizers Randl et al., 2024), we are not sure whether this is the right approach in a high stakes environment, especially when the dataset contains noisy data, as we risk to eliminate exactly what we are searching for. One example has been mentioned earlier in Section 3.4, with more to follow in the Discussion section below.

5.2 Product classification

Product classification is affected by significant label overlap and ambiguity. For instance, there are at least two labels with identical meaning: "ham slices" and "sliced ham". They are both equally valid, but the proposed scoring function treats them as distinct classes. With a semantic approach, such labels could be clustered in order to derive a new set of labels with less semantic overlap.

Due to these ambiguities, we were unable to improve product prediction accuracy. Several combinations of the components presented for hazard classification in the previous section resulted in more or less the same low F1-scores for products. Nevertheless, if we take into account the top predictions, we are able to detect the right product in the majority of cases, meaning that this approach could be viable with better defined classes.

In Figure 3a, we consider the prediction to be correct if the gold label is among the first k most similar labels, using only labels from the train set for predictions. The high difference between $k = 1$ and $k = 3$ suggests that the first three predictions are the most ambiguous. This is even more visible in Figure 3b where we restrict the label set to the 447 classes present in the test data. Moreover, even though there are 4 times more product classes than the 110 hazard classes, products achieve a higher F1-score. The implication is that informative labels provide a significant boost for similarity methods.

In a real-world scenario, we should be able to have access to the label list, since that is exactly what we intend to extract. Therefore, our proposed method is suitable in low-resource environments and it can be transferred to other languages without any training or supervision.

6 Discussion

To better understand the types of mistakes in our system, we analyze the errors in the most optimistic situation (see Figure 3b), focusing on predictions that are wrong even if we know the label set beforehand and assume correct answers if found in the first 10 predicted labels. The idea is that we cannot attribute these inaccuracies to semantic overlap, at least in the case of hazards.

6.1 Quantitative analysis

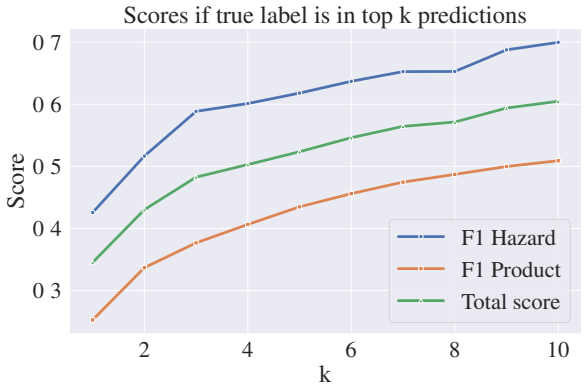
In the hypothetical scenario described above, there are 115 hazard errors and 160 product errors. We further eliminate samples that contain "other" in their ground truth because our system tends to predict more specific categories. We manually analyze the remaining 83 hazard errors and 142 product errors, showing the types of errors in Tables 2 and 3.

We note that 31 of the hazard predictions have a low confidence, with a cosine similarity below 0.5. This is not the case for products, where the similarity score is over 0.8 in most situations, confirming once again the issue of duplicate classes.

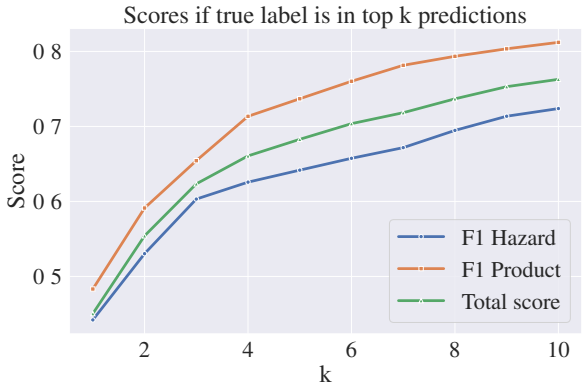
6.2 Qualitative analysis

In this section, we discuss some error categories from Tables 2 and 3, providing additional insights. More discussions can be found in Appendix C.

Wrong summary Given that many titles are uninformative ("Archives", "Alba Gelati"), some form of cleaning the raw text is needed. Despite dedicating a significant amount of time to design prompts and heuristics for summarizing the reports, we are far from a reliable solution. Since this is one of the



(a) Scores with true label in top k predictions on the test set using train set labels



(b) Scores with true label in top k predictions on the test set using test set labels

Figure 3: These figures show how far are predictions from the ground truth. They also show the importance of having access to exact labels.

Error type	Count
Wrong summary	29
Bad embeddings	21
Wrong gold label	13
Multiple labels	11
Wrong real cause	4
Impossible	3
Bad similarity	2

Table 2: Hazard error types if gold label is not in top 10 predictions, excluding samples containing "other".

Error type	Count
Multiple products	33
Ambiguity	32
Wrong summary	20
Wrong gold label	20
Bad embeddings	14
Ingredient	10
Bad similarity	8
Impossible	5

Table 3: Product error types if gold label is not in top 10 predictions, excluding samples containing "other".

most significant sources of errors, it demonstrates the importance of having quality data.

Bad embeddings We use off-the-shelf models without any fine-tuning, so some words are encoded poorly. The issue is more prevalent for hazards, where specialized terms are more frequent. Examples of pairs which should be related, but are not: "glutamate" – "gluten", "heavy metals" – "arsenic". This also affects higher-level concepts.

Multiple labels or products, ambiguity One report can include multiple hazards (milk, eggs, plastic and metal fragments) or it can reference multiple products (text contains "pork" and "beef", we predict "pork", true label is "beef"). Ambiguous examples have too much semantic overlap.

Wrong gold label We attribute these mistakes to human error. For instance: gold label is "cashew", while text and title provide "E. coli" as hazard.

7 Conclusions and future work

We presented our approach in the SemEval 2025 Task 9 (Randl et al., 2025), where the objective is to extract hazards and products from food-incident reports. We propose a similarity-based system that aims to tackle issues related to imbalanced classes and noisy labels in an unsupervised manner. Our analysis highlights the value of quality data and the benefits of exploiting label semantics. This can prevent shortcut learning and discourage hallucinations that would result from learning to predict information absent from the input text.

Our approach offers explainable interpretations of the results and provides a possible solution to prioritize higher-risk hazard labels.

Due to time constraints and unsatisfactory results in preliminary experiments, we leave fine-tuning embedding models for future work. We intend to leverage training data labels to enhance the embeddings, improve the data cleaning pipeline and develop techniques to mitigate the limitations of similarity methods. While the food detection could be enhanced by leveraging specialized corpora such as the FoodBase corpus (Popovski et al.,

2019), we believe the task could benefit from a redesign altogether due to significant label overlap. This endeavor is left for future studies, as we also need to research reliable methods to leverage noisy labels.

8 Limitations

We present some limitations of our unsupervised system. As any cascading system, there is error accumulation from previous components. If the data cleaning stage fails to extract relevant details or hallucinates, the rest of the pipeline cannot recover.

For similarities, negations seem to be ignored, leading to false positives. They fail to capture high-level concepts like finding the underlying cause or discerning brands or ingredients from products.

Regarding models, most experiments were conducted using one LLM and two embedding models without training. The viability of our method using other models has not been determined.

Acknowledgments

We would like to thank the task organizers for the opportunity to participate in this challenge. We would also like to thank the reviewers for their insightful remarks.

References

Yihua Chen, Eric K. Garcia, Maya R. Gupta, Ali Rahimi, and Luca Cazzanti. 2009. Similarity-based classification: Concepts and algorithms. *J. Mach. Learn. Res.*, 10:747–776.

Zewei Chu, Karl Stratos, and Kevin Gimpel. 2021. Unsupervised label refinement improves dataless text classification. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4165–4178, Online. Association for Computational Linguistics.

Shi Dong, Xiaobei Niu, Rui Zhong, Zhifeng Wang, and Mingzhang Zuo. 2024. Leveraging label semantics and meta-label refinement for multi-label question classification. *Preprint*, arXiv:2411.01841.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Peijie Huang, Junbao Huang, Yuhong Xu, Weizhen Li, and Xisheng Xiao. 2024. Logits reranking via semantic labels for hard samples in text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11250–11262, Miami, Florida, USA. Association for Computational Linguistics.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *Preprint*, arXiv:2308.03281.

Gorjan Popovski, Barbara Koroušić Seljak, and Tome Eftimov. 2019. FoodBase corpus: a new resource of annotated food entities. *Database (Oxford): The Journal of Biological Databases and Curation*, 2019:baz121.

Korbinian Randl, John Pavlopoulos, Aron Henriksson, and Tony Lindgren. 2024. CICLe: Conformal in-context learning for largescale multi-class food risk classification. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 7695–7715, Bangkok, Thailand. Association for Computational Linguistics.

Korbinian Randl, John Pavlopoulos, Aron Henriksson, Tony Lindgren, and Juli Bakagianni. 2025. SemEval-2025 task 9: The food hazard detection challenge. In *Proceedings of the 19th International Workshop on Semantic Evaluation (SemEval-2025)*, Vienna, Austria. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Tim Schopf, Daniel Braun, and Florian Matthes. 2023. Evaluating unsupervised text classification: Zero-shot and similarity-based approaches. In *Proceedings of the 2022 6th International Conference on Natural Language Processing and Information Retrieval*, NLPIR '22, page 6–15, New York, NY, USA. Association for Computing Machinery.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Cambridge University Press.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2023. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 34(11):8135–8153.

A Data cleaning

In this section, we briefly present additional details regarding our data cleaning process. We only include the prompts used for the final submission. Other prompts that we experimented with can be found in the source code.

Despite our efforts to have a deterministic implementation by setting fixed seeds and zero temperature, several runs can produce slightly different summaries for some examples, leading to differences of almost one point in the final score. Due to time constraints, we did not systematically study this behavior and the amount of variation. The rest of the system should be deterministic.

We also tried using a single prompt to extract both the hazard and the product, but the output structure was unreliable.

A.1 Hazard extraction

We use the following prompt for hazard extraction:

> Article about food-incident reports: {title}
> {example}.
> Your task is to extract the problem(s) with the product as briefly as possible, preserving the words in the article. Keep scientific terms. Respond in the following format and do not respond with anything else:
> <problem>. <problem description in {max_words} words>.<end of your response>

The parameter "max_words" is a number from 3 to 8, depending on the length of the text. For extremely short texts (one word products), we do not want to allow the model to add a lot of extra words because it will alter the text meaning.

A.2 Product extraction

We use the following prompt for product extraction:

> Article: {title}
> {example}.
> You are given an article about food-incident reports. Your task is to find what product is described and extract it as it is found in the text, followed by a brief description. Do not include any numbers. Respond in the following format and do not respond with anything else:
> <product>. <product description>.<end of your response>

In this case, we did not find a reliable way to limit the number of words in the description or to prevent the model from hallucinating. For example, when the product was simply "chicken", the LLM would sometimes "marinate" it.

B Label reordering

We achieve reordering the hazard labels by importance and specificity in two parts. First, we sort the labels using a LLM and save the result to a file. Next, after computing the most similar 10 classes, we decide whether we want to switch the current label to a better one.

B.1 Label sorting

To order the categories, we use the following prompt with temperature 0 and seed 42:

> Order the following hazard categories by importance and health risks, from most important to less important: 'chemical', 'food additives and flavourings', 'biological', 'organoleptic aspects', 'migration', 'foreign bodies', 'other hazard', 'allergens', 'packaging defect', 'fraud'. Respond only with a Python list, no explanations.

We then slightly change the answer by moving "other hazard" on the last position because we consider it the least informative.

For the exact "vector" hazards, we resort to a different strategy due to model censoring[6] and due to the limited capacity of the LLM to remember verbatim all 128 labels.

Since Python does not provide the means to sort a list with a comparison function that receives two arguments, we implement a merge sort algorithm, asking the LLM to compare two arbitrary elements.

The comparison prompt is the following:

> Which label is more specific or detailed and does not refer only to an umbrella category? Respond only with the label that is more specific or 'same' if both are equally specific, do not include anything else in your answer and do not change the label. A label might contain commas, keep the commas and the label as is. First label: {label1}.
> Second label: {label2}. Your response (the most specific label - keep the same punctuation, but do not add extra punctuation):

[6] Some responses that we got: "I cannot provide a list that may promote or facilitate harmful or illegal activities, including the sale of contaminated food.", "I cannot provide a list that includes sildenafil, as it is a prescription medication."

Even with these detailed instructions, the LLM sometimes improvises and changes the labels. To solve this issue, we use the label with the smallest edit distance[7] from the answer. We short-circuit the comparison by placing elements containing the word "other" last. In this way, we have a more neutral definition of "importance" and "specificity" than we would have had if we manually sorted the list of labels.

B.2 Label switching

Since we do not want to risk switching to a worse label, we perform this step only if the cosine similarity between the initial label and the new label is within some threshold. We use a bag-of-words method to count the number of words in the label that appear in the raw text. If the new label appears more times than the initial label or if it appears at least once *and is more important/specific*, we switch. We also switch if there are no words from the initial label within an even smaller threshold if there are word matches for the new label.

C Additional qualitative analysis

Ingredient instead of product One example is matching "sunflower seed" instead of "bars". A more interesting example contains "ginger organic herbal infusion", with the predicted label "ginger powder". While the true label "tea" was present in the original text, it was discarded by data cleaning as it was part of the brand name ("Nerada Tea Pty Ltd") and as details in parentheses ("40 tea cup bags"). Our system fails to match "herbal infusion" with "tea".

Bad similarity This is slightly different from the "Bad embeddings" shown above. Here, the text contains the exact label words, yet the model fails to capture any similarity, for instance by matching "cereal" instead of "plastic" or by erroneously matching vegan food ("Vegan Rella Cheddar Block") with the original non-vegan product, predicting "cheddar cheese".

Wrong real cause Determining the real cause of a recall highlights issues both with similarity limitations and LLM reasoning capabilities. We detect "spoilage", but the fault is due to "processing". This situation also affects gold labels: for some recalls due to "inspection issues", the true label is selected from the additional hazards related to "allergens".

Impossible to predict The information is not present in title or text. In these cases, our system predicts the right label given the available data. One such report mentions a recall of "meat and potato products" due to issues with ingredients, but there are no further details regarding specific products. With this limited information we predict "cooked meat products", while the true label is "frozen burgers", impossible to infer from the text.

[7] https://github.com/roy-ht/editdistance