CDHF at SemEval-2025 Task 9: A Multi-Task Learning Approach for Food Hazard Classification

Chu Duong Huy Phuoc
University of Information Technology
Vietnam National University - Ho Chi Minh City, Vietnam
23521229@gm.uit.edu.vn

Abstract

We present our system in SemEval-2025 Task 9: Food Hazard Detection. Our approach focuses on multi-label classification of food recall titles into predefined hazard and product categories. We fine-tune pre-trained transformer models, comparing BERT and BART. Our results show that BART significantly outperforms BERT, achieving an F1-score of 0.8033 during development. However, in the final evaluation phase, our system obtained an F1-score of 0.7676, ranking 14th in Subtask 1. While our performance is not among the top, our findings highlight the importance of model choice in food hazard classification. Future work can explore additional improvements, such as ensemble methods and domain adaptation.

1 Introduction

SemEval 2025 Task 9: The Food Hazard Detection Challenge (Randl et al., 2025) focuses on classifying food incident reports to identify hazards and affected products. The task includes two sub-tasks: (ST1) food hazard prediction and (ST2) hazard and product vector detection.

We present a multi-task learning approach using the facebook/bart-large-mnli (Lewis et al., 2020) model to predict hazard and product categories. Our system fine-tunes a transformer-based model with a custom neural network featuring two classification heads for multi-label prediction. Experimental results highlight the effectiveness of this approach in food hazard detection. [1]

2 Task and Background

The Food Hazard Detection task focuses on developing explainable classification models to analyze the titles of food-incident reports collected from web sources. These models aim to assist automated systems, such as web crawlers, in identifying and extracting food safety issues from online platforms, including social media. Given the potential economic and public health impact of food hazards, ensuring transparency in these classification systems is essential.

2.1 Task Description

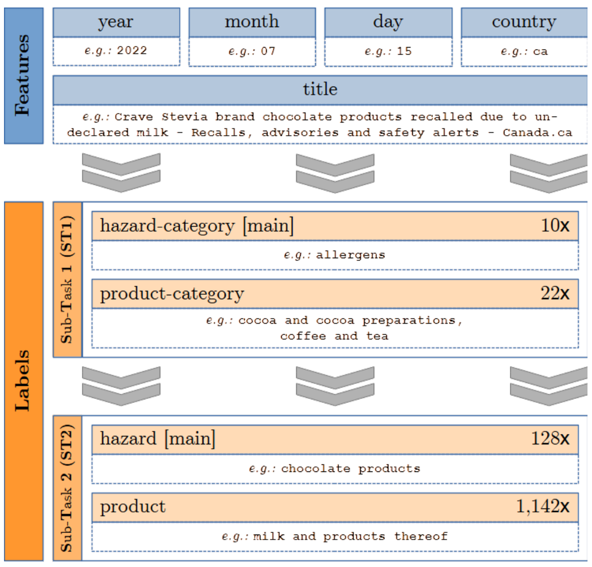

Figure 1: The blue boxes are model inputs; the orange boxes are ground truth labels per sub-task. The number on the right indicates unique values per label.

SemEval-2025 task 9: The Food Hazard Detection with two main subtask: **Subtask 1: category classification:** In this subtask, the goal is to classify a food-incident report into predefined categories. Specifically, models must predict both the product category (e.g., "meat, egg, and dairy products") and the hazard category (e.g., "biological contamination"). Since the dataset is highly imbalanced, handling rare categories effectively is a key challenge.**Subtask 2: vector classification:** This subtask requires a more detailed prediction by identifying the exact product (e.g., "ice cream") and hazard (e.g., "salmonella") mentioned in the report. Instead of selecting from broad categories, models

[1] https://github.com/fuocchu/CDHF_SemEval-2025-Task-9

must extract precise labels, making this task more complex. Performance is evaluated based on macro F1, with a focus on hazard detection.

2.2 Related Work

Explainability in food hazard detection remains underexplored despite its importance for trust and decision-making. Existing research mainly focuses on two types of explainability methods: model-specific and model-agnostic approaches. Model-specific methods are designed for particular models, offering tailored explanations. For example, (Assael* et al., 2022) and (Pavlopoulos et al., 2022) explored techniques that integrate explainability within neural architectures. These methods enhance transparency but are limited to the models they are built for. Model-agnostic approaches, such as LIME (Ribeiro et al., 2016), provide explanations that work across different models. They approximate model behavior by generating local explanations, making them more flexible but sometimes less precise.

In this task, explainability is emphasized by requiring participants to submit vector labels (ST2) as justifications for their category classifications (ST1). This ensures that predictions are interpretable and supports better validation of food safety risks.

[2].

3 System Overview

Our system for the Food Hazard Detection task focuses exclusively on Subtask 1 (ST1), which involves classifying food-related hazard and product categories based on textual data. We employ a transformer-based multi-task learning approach to handle both classification tasks simultaneously. This section details the key components of our system, including data preprocessing, model architecture, training strategy, and evaluation.

3.1 Dataset

We use the dataset provided by the organizers, which consists of 6,644 short texts related to food recall incidents. Each text is a title extracted from official food agency websites (e.g., FDA) and has been manually labeled by two food science or food technology experts. The text length ranges from 5 to 277 characters, with an average of 88 characters. All texts are in English and are annotated with a hazard category and a product category. Upon task completion, the full dataset will be released under the Creative Commons BY-NC-SA 4.0 license on (Randl et al., 2024).

[2]https://zenodo.org/records/10891602

"Randsland brand Super Salad Kit recalled due to Listeria monocytogenes"	
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"	
hazard:	milk and products thereof
hazard-category:	allergens
product:	meat preparations
product-category:	meat, egg and dairy products

"Nestlé Prepared Foods Recalls Lean Cuisine Baked Chicken Meal Products Due to Possible Foreign Matter Contamination"	
hazard:	plastic fragment
hazard-category:	foreign bodies
product:	cooked chicken
product-category:	prepared dishes and snacks

Figure 3: A sample of the dataset.

3.2 Model Architecture

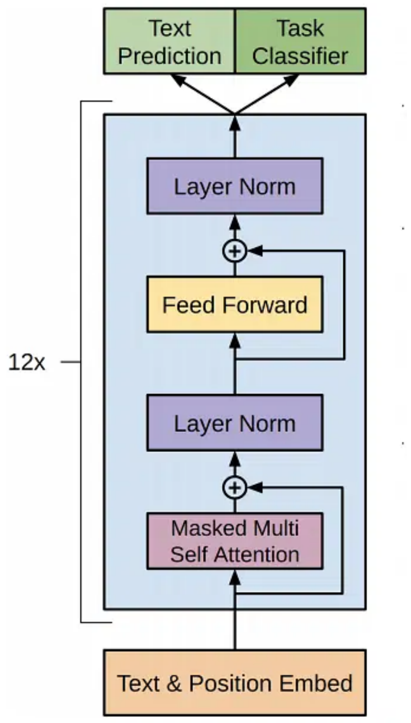

Figure 4: Simplified architecture based on BART.

We use BART (facebook/bart-large-mnli) (Lewis et al., 2020) as the backbone for food hazard classi-

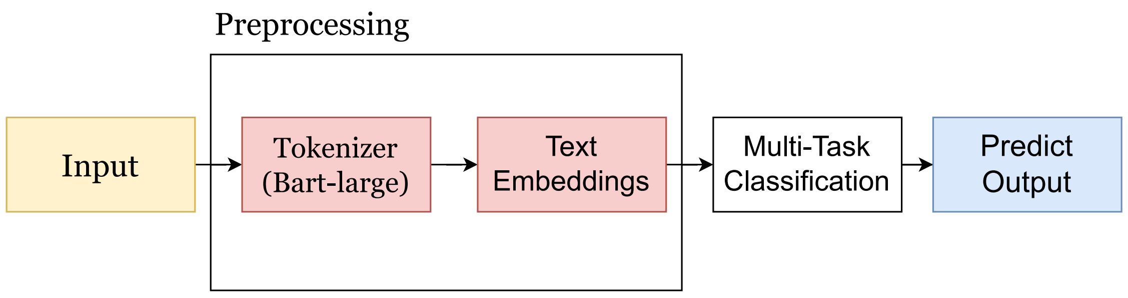

Figure 2: Our System Pipeline.

fication. BART is a transformer-based autoencoder pretrained for sequence-to-sequence tasks, making it well-suited for text classification. In our model, BART serves as an encoder to extract contextual representations from input text. The output from the [CLS] token is passed through two separate linear classifiers to predict the hazard category (10 classes) and product category (22 classes). BART enables the model to capture contextual information effectively, improving classification accuracy despite the imbalanced label distribution in the dataset.

3.3 Multi-Task Classification

Our model performs two parallel classification tasks:The hazard-category classifier predicts one of 10 possible hazard categories.The product-category classifier predicts one of 22 product categories. Both classifiers share the same base model but have separate fully connected (linear) layers for final classification. The CLS token representation is used as input to these classification layers. Formally, given an input text x, we obtain:

$$\hat{y}_{\text{hazard}} = \text{softmax}(W_h h + b_h)$$

$$\hat{y}_{\text{product}} = \text{softmax}(W_p h + b_p)$$

where W_h, W_p and b_h, b_p are learnable weight and bias parameters. The dataset is highly imbalanced, with certain classes appearing much more frequently than others. To address this, we use label encoding to convert categorical labels into numerical indices. Apply shuffling during training to improve generalization. Use early stopping to prevent overfitting.

4 Experimental Setup

We use the dataset provided by the organizers, which consists of 6,644 short texts. The data is split into training, validation, and test sets. The training set is used for model training, the validation set for hyperparameter tuning and early stopping, and the test set for final evaluation. Each text is tokenized using the BART-large-MNLI (Lewis et al., 2020) tokenizer with a maximum sequence length of 128. Labels are encoded using Scikit-learn's `LabelEncoder`, and missing labels in the validation and test sets are assigned default values. We fine-tune the BART-large-MNLI model for multi-label classification with a batch size of 16 and the AdamW optimizer (learning rate 2×10^{-5}). A learning rate scheduler (`ReduceLROnPlateau`) adjusts the learning rate based on validation loss. Early stopping is applied with a patience of 3 epochs, and training runs for up to 20 epochs. Our implementation uses Transformers (Wolf et al., 2020), Torch (Paszke et al., 2019), Scikit-learn (Pedregosa et al., 2011), and Pandas.

4.1 Evaluation Mectric

Task organizers compute the performance for ST1 and ST2 by calculating the macro-F1-score on the participants' predicted labels (`hazards_pred` & `products_pred`) using the annotated labels (`hazards_true` & `products_true`) as ground truth. The F1-score is calculated as follows:

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

where Precision and Recall are defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

where TP, FP, and FN represent true positives, false positives, and false negatives, respectively.

Phase	Team	Subtask 1 (F1)
Evaluation	Anastasia	0.8223
	MyMy	0.8112
	SRCB	0.8039
	PATeam	0.8017
	HU	0.7882
	CDHF (ours Top 14)	0.7646
Post evaluation	Anastasia	0.8223
	UIT-NaiveNotNice	0.8153
	MyMy	0.8112
	dml	0.8049
	SRCB	0.8039

Table 1: Top 5 results of Subtask 1.

5 Results

Our system did not achieve a high ranking in the competition, placing 14th in Subtask 1. However, we observed notable improvements in performance through model selection. Initially, using BERT resulted in a macro-F1 score of 0.71. By switching to BART, performance increased significantly to 0.8033 on the validation set. Finally, in the evaluation phase, our system achieved a macro-F1 score of 0.7676. To further analyze the effectiveness of our approach, we compared different model architectures. The results confirm that leveraging a more powerful pretrained model such as BART provides a substantial boost over BERT, likely due to its stronger contextual representation and sequence-to-sequence training paradigm.

For error analysis, we examined some misclassified examples and found that many errors involved ambiguous or rare hazard categories, suggesting that our model struggles with underrepresented classes. A confusion matrix could provide more insight into these misclassifications, highlighting specific categories where performance can be improved. Future improvements could focus on handling class imbalance, exploring data augmentation techniques, or incorporating external knowledge sources to enhance classification accuracy.

6 Conclusion

In this paper, we presented our approach for Subtask 1 of the Food Hazard Detection shared task. Our system leveraged pre-trained transformer models fine-tuned for multi-label classification, focusing on identifying food hazard categories and affected product types. Through our experiments, we observed a significant performance improvement when switching from BERT to BART, with the F1-score increasing from 0.71 to 0.8033 on the development set. This result highlights the advantage of using more advanced transformer architectures for text classification tasks. However, in the final evaluation phase, our system's performance slightly declined to an F1-score of 0.7676, ranking 14th overall. While our model did not achieve top-tier rankings, our findings underscore the potential of transformer-based approaches for food hazard classification. The results suggest that further optimizations, such as better handling of class imbalances, domain adaptation, or ensemble methods, could further enhance performance in future iterations of this task.

7 Future Work

While our multi-task design with two classification heads achieved reasonable results, we plan to explore more advanced architectures, such as shared bottlenecks or task-specific modules, to better capture task differences. The model's difficulty with rare and similar classes suggests applying class-balanced losses, targeted augmentation, and domain adaptation to improve generalization.

8 Limitations

Despite the improvements gained from using BART, our system still faced several challenges. First, our performance remained significantly lower than the top-ranked teams, suggesting that further enhancements are needed, such as better preprocessing, data augmentation, or more sophisticated model architectures. Second, our approach did not fully leverage domain-specific knowledge, which might have contributed to misclassifications. Finally, we did not explore ensemble methods, which could have further improved robustness and generalization. Future work could focus on addressing these limitations to develop a more effective food hazard detection system.

References

Yannis Assael*, Thea Sommerschield*, Brendan Shillingford, Mahyar Bordbar, John Pavlopoulos, Marita Chatzipanagiotou, Ion Androutsopoulos, Jonathan Prag, and Nando de Freitas. 2022. Restoring and attributing ancient texts with deep neural networks. *Nature*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Proceedings of ACL*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems (NeurIPS)*.

John Pavlopoulos, Leo Laugier, Alexandros Xenos, Jeffrey Sorensen, and Ion Androutsopoulos. 2022. From the detection of toxic spans in online discussions to the analysis of toxic-to-civil transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3721–3734, Dublin, Ireland. Association for Computational Linguistics.

Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Edouard Perrot, and Eric Duchesnay. 2011. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830.

Korbinian Randl, Manos Karvounis, George Marinos, John Pavlopoulos, Tony Lindgren, and Aron Henriksson. 2024. Food recall incidents.

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.

Marco Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Demonstrations*, pages 97–101, San Diego, California. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP): System Demonstrations*.

A Bart Model Architecture

BART (Bidirectional and Auto-Regressive Transformer) is a denoising autoencoder that combines a bidirectional encoder (like BERT) and an autoregressive decoder (like GPT). The encoder processes the full input context, while the decoder generates text sequentially from left to right. During training, BART applies text corruption techniques such as token masking, deletion, and sentence permutation, then learns to reconstruct the original text. It is widely used for text classification, summarization, and generation tasks. During training, BART applies text corruption techniques such as token masking, deletion, and sentence permutation, then learns to reconstruct the original text. It is widely used for text classification, summarization, and generation tasks.

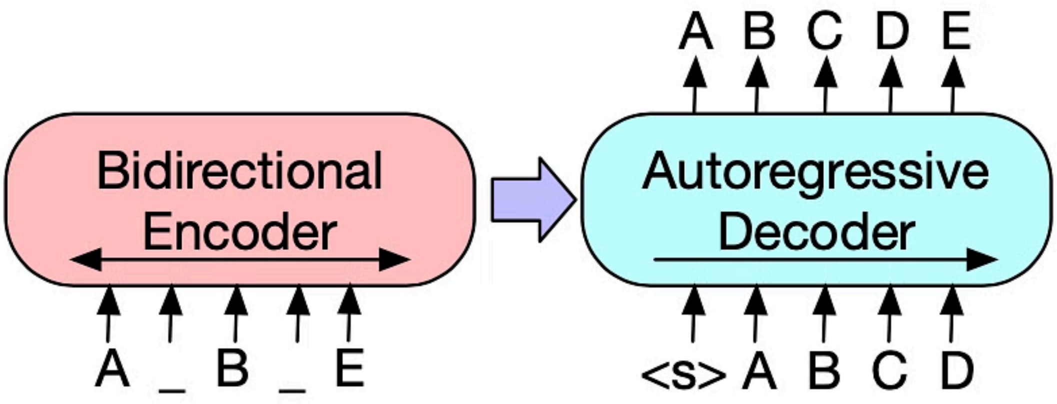

Figure 5: Bart model explained