Final Project: Hierarchical Multi-Label Text Classification

DATA304: Big Data Analysis

Due Date: December 20, 2025

General Instructions

This project is designed to foster your ability to apply the concepts learned in class to real-world data. The dataset and additional resources are available at the following link: **Link**.

- **How to Submit:** You must submit **three** components in total.

  1. **Prediction Results:** Submit your prediction results on Kaggle.
  2. **Code:** You must submit your code via GitHub. The submitted repository should include all necessary components to reproduce your final results (i.e., the best score achieved on Kaggle).
  3. **Report:** Use the following template: **Link**. The report should be written in English, up to 8 pages.

  Make sure to submit both on LMS and Kaggle before the deadline. All submissions are due by **23:59 KST on the deadline day**.

- **How the Project is Graded:**

  – **Performance (50%):** Your performance on Kaggle will be evaluated. A private leaderboard will be used internally for grading and will not be publicly disclosed.

  – **Report Quality (50%):** Evaluation based on the completeness, clarity, and organization of your report.

  – **Extra Credit (10%):** As requested by the College of Informatics, this component evaluates your contribution to this class: at least 10 GitHub commits (5%) and at least 90% utilization of the allocated AWS resources (5%). Information about remaining AWS usage time will be provided periodically via LMS. Full credit will be given as long as the above criteria are met.

  Please use the **Q&A board on LMS** for any assignment-related questions.

LLM Usage Policy

- **For code-related usage:** You are allowed up to **1,000 API calls** in total (approximately $1). You may use the **GPT-4o mini** API or any other freely available LLM (e.g., Llama-3.3-70B-Free). All input prompts and generated outputs associated with LLM calls must be submitted as separate files for verification purposes.
- **For report-related usage:** You may use LLMs for grammar checking, draft writing, or improving clarity and structure of your report. However, you must carefully review and take full responsibility for all content produced with LLM assistance.

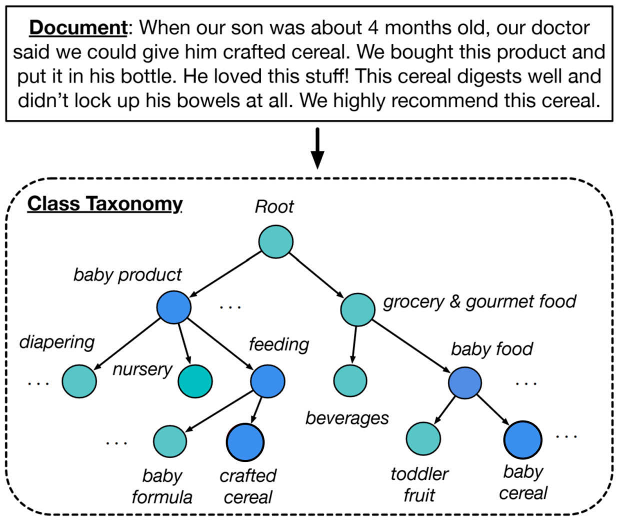
Honor Code

- All submitted code and report must be your own work.
- The use of any data other than the provided dataset is strictly prohibited. If you use publicly available pretrained models, make sure they are not fine-tuned on Amazon product data. It is your responsibility to verify the models you use.
- Any direct or indirect attempt to obtain additional data or information beyond the provided files — including collecting data from the web or making more than 1,000 API calls — will be regarded as academic misconduct and will result in a score of zero if detected.

Our group has worked with this dataset for a long time and is very familiar with the impact of each component; any irregular behavior can be easily identified. If you are uncertain or have any concerns, please consult the TA.

Task Description

Your task is to perform product review classification without using any labeled data. This problem is called a **Hierarchical Multi-Label Text Classification**, where each review can be associated with one or more product categories that are organized in a hierarchical taxonomy. You are encouraged to use this paper as a key reference for this project.



Document: When our son was about 4 months old, our doctor said we could give him crafted cereal. We bought this product and put it in his bottle. He loved this stuff! This cereal digests well and didn't lock up his bowels at all. We highly recommend this cereal.
Class Taxonomy
Root
baby product
grocery & gourmet food
diapering
nursery
feeding
baby food
beverages
baby formula
crafted cereal
toddler fruit
baby cereal

An example of hierarchical product taxonomy used in the task.

Task Definition. Formally, the task input includes an unlabeled text corpus $\mathcal{D}$ and a directed acyclic graph $\mathcal{T} = (\mathcal{C}, \mathcal{R})$ that defines the class taxonomy. Each $c_i \in \mathcal{C}$ represents a product class in the taxonomy. Each edge $\langle c_i, c_j \rangle \in \mathcal{R}$ indicates a parent-child relation, where class c_j is a sub-class of c_i. For example, one such edge may exist between $s_i =$ "baby food" and $s_j =$ "baby cereal".

The goal is to learn a multi-label text classifier $f(\cdot)$ that maps a document d into a binary label vector $\mathbf{y} = [y_1, \ldots, y_{|\mathcal{C}|}]$, where $y_i = 1$ if d belongs to class c_i, and $y_i = 0$ otherwise. In the given dataset, each document is associated with at least two and at most three labels.

Task Setup. You are provided with the following resources:

1. **Training corpus:** 29,487 product reviews without class labels.
2. **Classes:** 531 product categories.
3. **Class hierarchy:** A taxonomy file that defines parent–child relationships among classes (each line represents one relation).
4. **Class-related keywords:** A list of keywords associated with each product class.
5. **Test corpus:** 19,658 product reviews for evaluation.

You may use **all** of the above information and up to **1,000 LLM calls** to complete the task. That is, you are allowed to use the test corpus information during training.

Report Guideline

The organization of the report is flexible — you may organize it in any way you find most effective. However, your report must clearly describe the following elements:

1. **Silver label generation:** Explain the methodology used to generate silver labels Describe your design choices, heuristics, and the use of large language models (LLMs), if any. Provide reasoning for how your approach maintains label reliability.

2. **Training process:** Describe how you trained your model(s), including learning objectives (e.g., loss functions) and advanced strategies applied (e.g., self-training, regularization, data augmentation). Provide reasoning for how your approach improves learning effectiveness.

3. **Model and Prediction Method:** Clearly describe how you make predictions — which models were used (e.g., pretrained encoders, LLMs, classifiers) and how they were combined or integrated to produce final predictions, if applicable.

4. **Experimental results:** Summarize your experimental results in a table that includes both your best-performing result and other meaningful attempts or ablation variants you tried. You are encouraged to use techniques covered in class, but you are not limited to them. We expect your results table to include a comparison of various techniques introduced in class, including at least **self-training** and **GNN-based approaches**. For reference, see Table 2 in the reference paper.

5. **Case study and discussion:** Present at least one successful example and one failure case from your classification results. Discuss possible reasons for errors and insights about the model's limitations and behavior.

Code Guidelines

All previous programming assignments were intended to prepare you for this final project. You are now expected to handle new data and implement the entire pipeline independently, building upon the codes and techniques introduced throughout the course. Therefore, no additional baseline code will be provided. A dummy baseline (random prediction) is provided to illustrate the required Kaggle submission format.

To ensure reproducibility, you must minimize randomness in your implementation. You must explicitly fix random seeds for both Python and PyTorch, as shown below:

```
import random
import numpy as np
import torch

random.seed(42)
np.random.seed(42)
torch.manual_seed(42)
torch.cuda.manual_seed_all(42)
```

Submission Guidelines

Kaggle Submission

- Submit your prediction file in `.csv` format to Kaggle: Link
- Ensure that the filename of your submission follows the format: `studentID_final.csv`
- Set your **Kaggle team name** to your AWS account ID (e.g., `korea-dt-xx`). **This is mandatory in order to receive credit in our grading system. Please be careful!**
- There is **no limitation on the number of Kaggle submissions**.

LMS Submission

- Submit the following materials on the LMS:
  - Your report in **PDF** format.
  - The **GitHub repository link** containing your final implementation.

Code Submission

Your GitHub repository will be directly used to verify the reproducibility of your Kaggle results. Please follow the instructions below carefully to ensure that your code can be successfully executed and evaluated.

- The submitted code must be executable **without any modification**. We will clone your repository and run your code to verify reproducibility. Submissions that cannot be reproduced will not receive credit.
- Provide a clear and detailed description of how to reproduce your results in the repository's `README.md` file. Include instructions for running your codes.
- If your implementation involves multiple steps and cannot be executed with a single Python file, include a shell script (`.sh`) that runs the entire process to ensure reproducibility.
- If you use intermediate files or trained models that are too large to upload to GitHub, store them in an external file system (e.g., Google Drive) and clearly provide the download link in your `README.md`.
- **GitHub details:**
  - Repository name: `https://github.com/Your_GitHub_ID/20252R0136DATA30400`
  - Commits made after the deadline will not be considered.
  - It is recommended to keep your repository **private before the deadline** and make it **public after submission**.