Model Summary

A1. Team Background

Competition Name:	Coleridge Initiative - Show US the Data
Team Name:	Chun Ming Lee
Private Leaderboard Score:	0.575
Private Leaderboard Place:	2nd

Team Member (1/1)

Name:	Chun Ming Lee
Location:	Singapore
Email:	leecming@gmail.com

A2. Individual Background

Team Member (1/1): Chun Ming Lee

Professional Background:	- NLP Data Scientist at various Singaporean startups - Management Consultant at McKinsey & Co. - Trading Software Developer at Bermudan Hedge Fund
Academic Background:	- MBA, London Business School - Master of Information Systems Management, Carnegie Mellon University - BSc. Computer Science, Carnegie Mellon University
Misc.:	- Joined competition to keep occupied during COVID-19 lockdown - 1st place winner of 2018 and 2020 Kaggle Jigsaw Toxic Comments NLP competitions

A3. Summary

For business users:

The key insight was that datasets are frequently referenced in papers as mixed cap words followed by an acronym in parentheses e.g., “Baltimore Longitudinal Study of Aging (BLSA)”. My solution searches documents for such phrases and applies an AI model to distinguish between datasets (e.g., “Baltimore Longitudinal Study of Aging (BLSA)) and other types of references (e.g., “Organisation for Economic Co-operation and Development (OECD)).

For technical users:

1. Search across documents for strings in the format “LONG-NAME (ACRONYM)” using the Schwartz-Hearst non-learning string search algorithm (think of it as an advanced regex search)
2. Classify candidate strings as datasets (or not) using a HuggingFace Transformer binary classifier fine-tuned on hand-annotated labels derived from the training dataset
3. Threshold and propagate - search for LONG-NAME (i.e., without acronym) if LONG-NAME (ACRONYM) occurs frequently across documents
4. Collate predictions for each document - accept candidates if they meet key requirements (i.e., either minimum doc frequency or regex match strings that are very likely to be datasets e.g., mixed cap words ending with Study/Survey)

Refer to the Kaggle forum post - “2nd place solution overview” which provides an informal discussion of my approach.

A4. Feature Engineering

- Minimal text pre-processing was applied except the application of the clean_text function provided by the competition organisers which filters raw text for alphanumeric characters and single spaces.

A5. Training Method(s)

- The Schwartz-Hearst algorithm used to search for LONG-NAME (ACRONYM) is a non-learning algorithm and did not requiring training (or any other configuration)
- The training code for the HuggingFace Transformer binary classifier can be found in label_classifier.py (alongside the 5K custom labelled corpus, roberta-annotate-abbr.csv) included in this zip.
    - Use the included Dockerfile to create the Docker environment with the required Python libraries
    - Training settings: I trained for 5 epochs on shufflings of the 5K training samples with learning rate fixed at 1e-5 and batch-size 32. Given the relatively short texts involved, I set a maximum sequence length of 128 (truncating/padding as

necessary). I did not set a fixed seed for shuffling so you might see different results if you train yourself.
- The final fine-tuned model I used can be found here.

A6. Interesting Findings

- One reason for my outperformance was I focused on building a model that could reject non-dataset strings. I observed that many teams were scraping data.gov and other websites for positive examples of datasets. However, building a model leveraging this data was likely to induce a bias to making more predictions since the models were trained with many positive examples and relatively fewer negative examples. This is important given the competition metric which punished False Positives more than False Negatives.
- Any attempt to inject semantic context into my models seemed to worsen performance. I wonder if the annotation methodology could be improved for future competitions - instead of providing just the dataset string as the training label; perhaps have the annotators include the full sentence + the dataset string as labels for a future competition?

A7. Simple Features and Methods

- I did not perform extensive tests but a simple regex match of "(A-Z)[a-z]+ Study/Survey$" and "(Study|Survey) of" in lieu of using a Transformer classifier would likely capture a significant chunk of datasets.
- Replacing the Transformer with other NLP architectures (e.g., word2vec + RNN) might be another way of improving run-time without losing too much accuracy.

A8. Model Execution Time

- **Training time**: It takes <10 minutes to fine-tune the HuggingFace Transformer binary classifier using roberta-base on local hardware (AMD Threadripper 3960X, 64GB RAM, Nvidia A6000 GPU)
- **Inference time**: It takes ~10 minutes to perform inference on the test set.

A9. References

- Inferencing notebook used to generate 0.575 private LB predictions
- Kaggle dataset for final fine-tuned Transformer binary classifier
- "A Simple Algorithm for Identifying Abbreviation Definitions in Biomedical Text"
- Github for Python implementation of Schwartz-Hearst