Coresets In NLP

Ayush Srivastava(2021457) [1] Abhishek Sushil(2021441) [1] Manas Narang(2021473) [1]

[1]Indraprastha Institute of Information Technology Delhi

Introduction

NLP faces challenges with large unlabeled datasets due to limited supervised task utility. This work enhances coreset selection with embedding techniques, dynamic balancing, and optimized preprocessing to improve performance and ensure class balance.

Flowchart

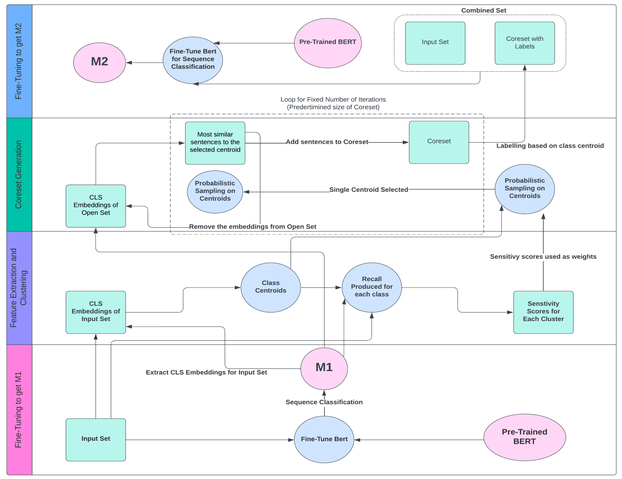
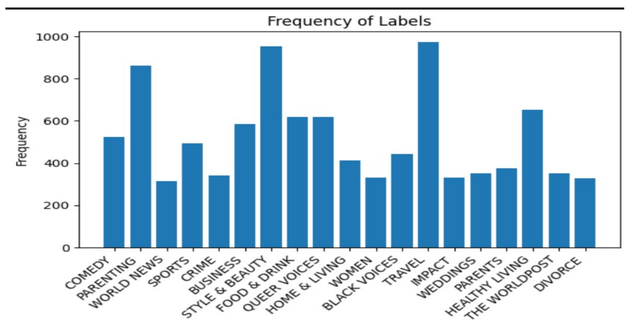


Figure 1. Flowchart of work

Dataset Description

- **Emotion Classification Data:** Used the English component of the MELD-FR dataset (sentences and emotions from F.R.I.E.N.D.S.) as the input set and created a 7000-sample open set of unlabeled data from similar sitcoms.
- **News Category Classification Data:** Used a Kaggle dataset for news categories, splitting 10% (10,000 samples) as the input set and the rest as the open set, with preprocessing yielding 19 final categories.



Methodology

1. Initial Training & Baseline Setup

1. **Dataset Preparation:** Split labeled data into training and test sets with equal class representation in the test set for fair evaluation.
2. **Baseline Model:** Fine-tuned a pre-trained BERT model (BERT Model-1) on the training set and evaluated it on the test set to establish a performance baseline.

2. Embedding Extraction & Clustering

1. **Embedding Generation:** Used BERT Model-1 to extract [CLS] token embeddings from the dataset by concatenating headlines and descriptions for meaningful sentence representations.
2. **Clustering:** Applied class-based clustering and weighted k-means to determine centroids for similarity calculations and sentence selection.

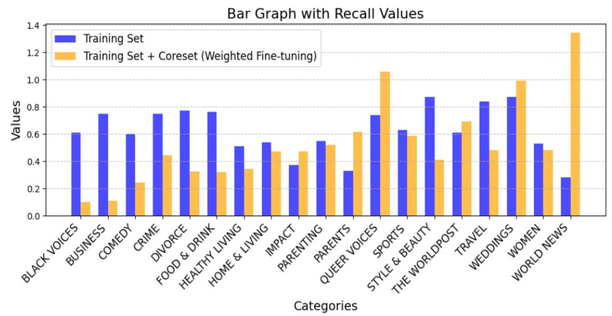


Figure 3. Methodology

3. Coreset Extraction & Enhanced Training

1. **Coreset Extraction:** Utilized weighted k-means, probabilistic sampling, and approximation tools (e.g., ANNoy, FAISS) to extract diverse coresets ensuring balanced class representation.
2. **Final Model:** Combined coresets with the initial training data to fine-tune BERT Model-2, whose performance was compared to the baseline and alternative approaches.

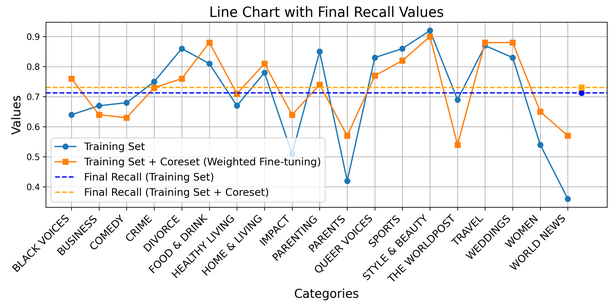


Figure 4. Methodology

Class Sampling Weights Heuristic

The Class Sampling Weights Heuristic determines the number of samples to extract from the open set for each class to ensure approximately equal true positives across all classes.

Definitions

Let:

- Z: Number of classes,
- n_i: Number of samples in class i in the original dataset,
- r_i: Recall for class i,
- N_i: Number of samples predicted to belong to class i in the open set,
- R: Average recall across all classes.

Approximate True Positives

The approximate number of true positives for class i is:

$$A[\text{TP}_i] = n_i + r_i A[N_i].$$

Equalizing Approximate True Positives

To ensure equal approximate true positives across all classes ($A[\text{TP}_i] = A[\text{TP}_j]$ for all i, j), the approximate true positives for any class are given by:

$$A[\text{TP}_i] \approx \frac{R \cdot \sum_{j=1}^{Z} A[N_j] + \sum_{j=1}^{Z} n_j}{Z}.$$

Solving for N_i

Rearranging for N_i, the number of samples to extract for class i is:

$$N_i \approx \frac{\frac{R \cdot \sum_{j=1}^{Z} A[N_j] + \sum_{j=1}^{Z} n_j}{Z} - n_i}{r_i}.$$

This provides the required number of samples from the open set for each class to balance the approximate true positives.

Results and Analysis

Method	Recall	Precision	F1-Score
Input Set	0.61	0.62	0.61
Exact Number Method	0.58	0.55	0.55
Probabilistic Sampling using Cosine Similarity	0.69	0.68	0.68
Probabilistic Sampling using ANNoy, FAISS	0.71	0.72	0.71

Table 1. A table caption.

Currently the best results are obtained using Probabilistic Sampling using ANNoy, FAISS. However, cosine similarity produces almost similar results.

References

[1] Practical coreset construction in machine learning, 2017. Available at: https://arxiv.org/abs/1703.06476.

[2] Efficient coreset selection with cluster-based methods. *ACM*, 2023. doi: 10.1145/3580305.3599326. Available at: https://doi.org/10.1145/3580305.3599326.

[3] Coreset sampling from open-set for fine-grained self-supervised learning, 2023. Available at: https://arxiv.org/abs/2303.11101.