CMSC848M: Multimodal Computer Vision Coding Assignment

Instructor: Ruohan Gao, TA: Kelin Yu

Spring 2025

1 Overview

This assignment explores different approaches to multi-modal learning by leveraging different sensory data—vision, touch, and audio—focused on the task of material classification for real-world objects. You will implement and compare several methods across four tasks, progressing from unimodal, multi-modal, to cross-modal learning, allowing you to gain hands-on experience and a deeper understanding of multi-modal learning techniques.

2 Setup Instructions

2.1 Environment Setup

1. Create and activate a new conda environment with Python 3.9:

```
    conda create -n cmsc848m python=3.9
    conda activate cmsc848m

```

2. Install required packages using pip:

```
    conda install pytorch torchvision torchaudio pytorch-cuda=11.8 -c pytorch
    -c nvidia
    pip install -r requirements.txt

```

2.2 Verification

To verify your setup: Ensure CUDA is available if using GPU:

```
    import torch
    print(torch.cuda.is_available())

```

3 Dataset Organization

The dataset consists of sensory measurements of about 100 real-world objects represented across three modalities.

3.1 Dataset

You can download the dataset from the google drive link below:
https://drive.google.com/file/d/1XxBkgVMf3Thbyk5EzYfzC5Kvc8DqS4es/view

3.2 Data Structure

- **Vision**: RGB images (`/vision/`)
- **Touch**: Tactile sensor readings (`/touch/`)
- **Audio**: Spectrogram representations (`/audio_examples/`)

3.3 Dataset Files

- **Split Files**:
  - Tasks 1-2: `split_0.json`
  - Tasks 3-4: `contrastive_split.json`
- **Metadata**:
  - `label.json`: Class labels for materials
  - `objects.csv`: Detailed object information
  - `material_dict.json`: Material category

4 Task 1: Unimodal Classification (25 points)

4.1 Objective

Implement single-modality classifiers for material recognition using network architectures in `models/unimodel.py`.

4.2 Implementation Requirements

1. Complete the `FENet`:
   - Implement feature enhancement network using depthwise separable convolutions
   - Add squeeze-and-excitation blocks for channel attention
   - Include skip connections for better gradient flow
   - Design an efficient architecture with 4 feature enhancement blocks
2. Implement three modality-specific classifiers:
   - `VisionClassifier`: For RGB image input
   - `TouchClassifier`: For tactile input
   - `AudioClassifier`: For audio spectrogram input

4.3 Training Instructions

- Use **train.py** with provided shell script **train_unimodal.sh** for training. Change the backbone input in the shell script accordingly.
- Support both ResNet and FENet backbones
- Results will be stored in `experiments/unimodal/`
- Monitor training progress through tensorboard logs

4.4 Deliverables

- Completed implementation code
- Training curves showing loss and accuracy
- Comparative analysis between ResNet and FENet performance

4.5 Open Question

Analyze and explain which modality provides the most discriminative features for material classification. Support your answer with experimental results and reasoning.

5 Task 2: Multi-Modal Fusion (25 points)

5.1 Objective

Implement and compare different fusion strategies for combining information from multiple modalities in `models/multimodal.py`.

5.2 Implementation Requirements

1. Complete the `LateFusion` class:
   - Design weighted fusion mechanism
   - Implement prediction combination strategy
   - Use the best-performing backbone from Task 1
2. Complete the `AttentionFusion` class:
   - Implement cross-modal attention mechanism
   - Design multi-head attention architecture
   - Create effective feature combination strategy

5.3 Training Instructions

- Use the **train.py** with the corresponding shell script **train_multimodal.sh** for training
- Results will be stored in `experiments/multimodal/`
- Compare performance between fusion strategies

5.4 Deliverables

- Completed implementation code
- Training curves for both fusion methods
- Analysis of fusion strategy effectiveness

5.5 Open Questions

Compared with unimodal material classification, is the multimodal fusion increasing the performance for material classificaiton? Why or why not? Analyze and explain which modality has better performance for this task and support your answer with experimental results.

6 Task 3: Contrastive Learning (25 points)

6.1 Objective

Implement contrastive learning for learning aligned cross-modal representations in `models/contrastive.py`.

6.2 Implementation Requirements

1. Complete the `ModalityEncoder`:
    - Design modality-specific feature extractors
    - Implement projection head architecture
    - Ensure proper feature normalization
2. Complete the `ContrastiveLearning` class:
    - Implement InfoNCE loss calculation
    - Add temperature scaling mechanism
    - Handle positive and negative pair creation

6.3 Training Instructions

- Use **train.py** with the corresponding shell script **train_contrastive.sh** for training
- Use best-performing backbone from Task 1
- Results stored in `experiments/contrastive/`
- Monitor contrastive loss convergence

6.4 Deliverables

- Completed implementation code
- Training curves showing contrastive loss
- Feature space visualization and analysis

6.5 Open Questions

Analyze how contrastive learning helps align representations from different modalities. What are the key factors affecting the quality of learned representations?

7 Task 4: Cross-Modal Retrieval (25 points)

7.1 Objective

Implement cross-modal retrieval using learned representations from previous tasks in `models/retrieval.py`.

7.2 Implementation Requirements

1. Complete the `ModalityEncoder`:
    - Implement feature extraction for retrieval
    - Add feature normalization
    - Ensure consistent feature dimensions
2. Complete the `ModalityRetrieval` class:
    - Implement database building functionality
    - Add nearest neighbor search
    - Calculate retrieval metrics (mAP, R@K)

7.3 Evaluation Instructions

- Use **retrieval.py** with the corresponding shell script **evaluate_retrieval.sh** for evaluation
- Results will be saved in `retrieval_results.json`
- Compare performance across different modality pairs

7.4 Deliverables

- Completed implementation code
- Retrieval performance metrics
- Analysis of success and failure cases

7.5 Open Questions

What factors affect retrieval performance between different modality pairs? Provide examples and analysis of both successful and failed retrievals.

8 Additional Notes

8.1 Code Modification

- Students may modify `train.py` and **data/dataset.py** if needed
- **Shell scripts should be adjusted for local paths (change things like "your data path" into your local paths) and learning parameters**

8.2 Evaluation Criteria

- Code implementation quality and documentation
- Training curves and performance metrics
- Depth of analysis in open questions
- **There is no target classification accuracy to be achieved. Since the dataset is small, overfitting is possible and expected. The main goal is for you to understand and make use of different multi-modal learning techniques and be able to analyze the experimental results.**

9 Submission Requirements

9.1 Required Files

Submit as a single zip file that includes the following:

1. Completed code for all four tasks
2. Training curves for Tasks 1-3
3. Evaluation results for Task 4
4. Written analysis for each open question

9.2 Format

- Code should be well-documented and follow Python style guidelines
- Analyses should be clear and supported by experimental results