Multi-modal Medical Diagnosis via Large-small Model Collaboration

Supplementary Material

1. Reproduction Statement

1.1. Code Implementation

To facilitate the reproduction of our method, we present the core components of our approach here. Specifically, Listing 1 showcases the key code used during the collaborative training process, while Listing 2 provides the implementation details of our adaptive weighting strategy.

```
# Input paired large and small
    feature embeddings
large_logits, small_logits,
    large_proj, small_proj = model(
    large_image, large_text,
    large_timeseries, small_feat)
# Compute losses
if model.weight_type == 'learnable':
    large_weight, small_weight = model
        .learnable_weight()
    logits = large_weight *
        large_logits + small_weight *
        small_logits
    task_loss = task_criterion(logits,
         target)
    sim, contrastive_loss = model.
        contrastive_loss(large_proj,
        small_proj)
else:
    sim, contrastive_loss = model.
        contrastive_loss(large_proj,
        small_proj)

    large_weight, small_weight = model
        .get_adaptive_weight(sim.mean()
        )

    task_loss = task_criterion(
        large_weight * large_logits +
        small_weight * small_logits,
        target)
loss = task_loss.mean() +
    contrastive_weight *
    contrastive_loss.mean()
```

Listing 1. Collaborative Training Implementation

During a training epoch, apart from the Learnable Weight strategy, we first compute the contrastive loss to obtain the similarity scores of the features from large models collaboration and small model. Based on the different weight computation methods (shown in Listing 2), the task loss is then calculated and used as the weight for decision fusion.

```
# Learnable Weight
def learnable_weight(self):
    # Ensure weights sum to 1
    fusion_weights = self.
        fusion_softmax(self.
        fusion_weights)
    large_weight = fusion_weights[0]
    small_weight = fusion_weights[1]
    return large_weight, small_weight

# Inverse Contrastive
def inverse_weight(self, sim,
    temperature=1.0,
                alpha=1.0):
    # Calculate base weights
    scaled_loss = sim / temperature
    inverse_loss = 1.0 / (alpha *
        scaled_loss + 1e-6)

    # Normalize weights
    large_weight = torch.sigmoid(sim)
    small_weight = inverse_loss / (
        inverse_loss + 1)

    # Ensure weights sum to 1
    total_weight = large_weight +
        small_weight
    large_weight = large_weight /
        total_weight
    small_weight = small_weight /
        total_weight

    return large_weight, small_weight

# Gaussian Contrastive
def gaussian_weight(self, sim, sigma
    =1.0):
```

```
    small_weight = torch.exp(-sim**2 /
        (2 * sigma**2))
    large_weight = 1 - small_weight
    return large_weight, small_weight

# Threshold Contrastive
def threshold_weight(self, sim,
    threshold=0.5,
                slope=10):
    x = sim - threshold
    large_weight = 0.5 + 0.5 * torch.
        tanh(slope * x)
    small_weight = 1 - large_weight
    return large_weight, small_weight
```

Listing 2. Adaptive Weight Strategies Implementation

1.2. Model Architecture and Configuration

Large Single-Modal Models Our framework leverages several state-of-the-art pre-trained large models to get large embeddings with different modalities of medical data. For each modality, we carefully select models that have demonstrated strong performance in their respective medical domains. Tab. 1 presents the specific models employed for each modality and their corresponding feature dimensions. These models serve as our modality-specific encoders, providing rich representations that capture the unique characteristics of each data type. The diverse input dimensions reflect the varying complexity and information density across different medical data modalities, which are later unified through our MoME module.

Modality	Model	Output Dimension
X-ray Images	DINOv2	1536
Whole Slide Images (WSI)	CLAM + UNI	1024
CT Scans	Merlin	512
Time Series	MOIRAI	1024
Medical Text	Me-LLaMA-13B	5120
Clinical Data	BioMistral	4096

Table 1. Model Architecture and Configuration

MoME Fusion Module Configuration The Mixture-of-Modality-Experts (MoME) Fusion module processes these multi-modal features through a unified hidden dimension calculated as $d_{\text{small}}//3$, where d_{small} is the hidden dim of small model. The final output dimension is set to d_{small}, which aligns with the feature dimension of the small model pathway to enable effective collaborative learning.

Training Configuration We conduct extensive experiments across multiple medical tasks from two datasets: MIMIC-IV-MM (mortality prediction, long-stay prediction, and readmission prediction) and MMIST (Vital-12 prediction). Through careful hyperparameter optimization, we identify the optimal configuration for each task. Tab. 2 summarizes these task-specific parameters, including the contrastive loss weight (λ), projection feature dimension (d), inverse temperature (τ), inverse alpha (α), Gaussian sigma (σ), threshold value (t), and threshold slope (sl).

Parameter	MIMIC-IV-MM			MMIST ccRCC
	Mortality	Longstay	Readmission	Vital-12
λ	0.59	0.50	0.67	0.50
d	2048	256	2048	128
τ	0.59	0.50	0.78	0.07
α	0.71	1.00	0.41	1.00
σ	0.84	1.00	0.66	0.10
t	0.57	0.60	0.94	0.15
sl	9.80	10.00	19.76	10.00

Table 2. Optimal Training Configuration for Different Tasks

1.3. Ablation Studies on Hyperparameters Configuration

To investigate the impact of different hyperparameters on model performance, we conduct comprehensive ablation studies on λ and d. Tab. 3 shows the fixed parameter settings used during these analyses. For each task, we maintain these parameter values constant while varying the target parameter (λ or d) to ensure controlled experimental conditions.

Parameter	Task		
	Mortality	Long-stay	Readmission
Analyzing λ:			
d	128	128	128
τ	1.0	1.0	1.0
α	1.0	1.0	1.0
σ	0.2	0.2	0.2
t	2.5	2.5	2.5
sl	10	10	10
Analyzing d:			
λ	0.5	0.5	0.5
τ	1.0	1.0	1.0
α	1.0	1.0	1.0
σ	0.2	0.2	0.2
t	2.5	2.5	2.5
sl	10	10	10

Table 3. Other hyperparameter Settings for Ablation Studies of hyperparameter λ and d

2. Broader Impacts

The widespread adoption of large medical AI models has been significantly hindered by two major challenges: extensive computational requirements and limited availability of paired multi-modal medical datasets. While many healthcare institutions lack access to high-end computing infrastructure (such as 80GB A100 GPUs) necessary for training and fine-tuning large models, they also struggle with insufficient paired multi-modal training data for effective model development. AdaCoMed addresses both challenges simultaneously: it enables effective multi-modal medical diagnosis on more accessible hardware platforms (such as consumer-grade RTX 4090 GPUs), while leveraging the rich medical knowledge already encoded in existing large single-modal models, rather than solely relying on limited paired multi-modal datasets.

Beyond computational and data efficiency, our approach demonstrates significant implications for healthcare democratization and accessibility. By reducing both hardware requirements and dependency on paired multi-modal training data, AdaCoMed makes state-of-the-art diagnostic capabilities more accessible to a broader range of medical facilities. Our framework effectively transfers knowledge from well-trained large models to more efficient architectures, allowing healthcare providers to benefit from the extensive medical knowledge captured by these models without the need for massive paired dataset collection or high-end computing resources. This could lead to more equitable access to AI-assisted medical diagnostics across different regions and healthcare systems, particularly benefiting institutions with limited data collection capabilities.

3. Limitations and Future Explorations

Our work currently faces several limitations. First, the diagnostic model remains a black box, lacking interpretability analysis for multimodal medical data, which limits our understanding of how the model arrives at its decisions. Additionally, the model's performance on tasks with highly imbalanced data is not yet sufficient for clinical application, as it struggles with robustness in these challenging scenarios. In the future, we plan to address these limitations by conducting interpretability studies and validating the model on more real-world medical data through collaborations, aiming to enhance its clinical reliability and generalizability.

4. Dataset

4.1. Dataset Statistic

Tab. 4 summarizes the key characteristics of the datasets employed in our study, encompassing task specifications, data distributions, and modality information.

Dataset	Task	Modalities	Subjects	Pos:Neg
MIMIC-IV-MM	Mortality	XRay, Note, Time	11,483	1:6.6
	Longstay		10,069	1.5:1
	Readmission		11,483	1:22.5
MMIST ccRCC	12-month Survival	CT, WSI, Clinical	248	15:1

Table 4. Detailed characteristics of the experimental datasets, including task types, input modalities, number of subjects, and class distribution ratios.

4.2. Dataset Preprocessing

We carefully preprocess each modality in our datasets to ensure optimal model performance:

X-rays Processing For X-ray images, we implement a standardization pipeline that maintains aspect ratio while ensuring consistent dimensions. This involves zero-padding to achieve uniform dimensions of 256×256 pixels. Our training augmentation includes dynamic cropping to 224×224 regions and pixel value normalization to ensure stable model training.

Text Processing Clinical notes undergo comprehensive preprocessing to improve textual quality. This includes removing line breaks, standardizing whitespace, and text normalization through tokenization.

Timeseries Processing We extract and organize Electronic Health Records (EHR) into structured time series data to capture the temporal evolution of patient conditions. The demographic features are processed through a systematic categorization approach, including discretizing continuous variables into meaningful bins. For vital signs, we select key physiological indicators comprising both numerical and categorical measures. Laboratory tests are curated to focus on the most clinically relevant parameters, while procedure records are standardized to capture major clinical interventions. This comprehensive processing ensures all clinical variables are in a consistent, analyzable format while preserving their medical significance.

Whole Slide Images (WSI) Processing For processing large single-modal model inputs, we follow the CLAM framework to extract features from WSI images. Due to the high resolution of WSI images, CLAM performs segmentation and patching before feeding the data into the UNI model. By default, we set the step size and patch size to 256. For data processing of small multimodal model inputs, we directly utilize the preprocessed representations provided by MMIST ccRCC dataset.

CT Scans Processing As Merlin supports the processing of 3D CT images, we apply it directly to extract features from CT scans without additional preprocessing. Since the MMIST ccRCC dataset provides multiple CT scans for a single sample, we use the scan recommended by the dataset contributors. For data processing of small multimodal model inputs, we also directly utilize the preprocessed representations provided by MMIST ccRCC dataset.

Clinical Data Processing The clinical data in the MMIST ccRCC dataset includes various numeric attributes. Since TableLLM accepts only text inputs, we convert the clinical data into text using the following prompt: 'The patient is a male/female of Asian/Black or African American/Hispanic or Latino/White/other race, diagnosed at age #age#, who has a/no VHL mutation and a/no PBMR1 mutation with/without a TTN mutation. Tumor characteristics include tumor stage #stage# with node involvement at stage #stage# and pathological metastasis at stage #stage#. The overall tumor stage is #stage# and the tumor grade is #grade#.'

For data processing of small multimodal model inputs, we process the clinical data into embeddings. Specifically, categorical embeddings are created for attributes such as 'gender,' 'cancer history,' 'VHL mutation,' 'PBMR1 mutation,' 'TTN mutation,' and race categories ('Asian,' 'Black or African American,' 'Hispanic or Latino,' 'White,' 'other'). Ordinal embeddings are generated for attributes like 'ajcc path tumor pt,' 'ajcc path nodes pn,' 'ajcc clin metastasis cm,' 'ajcc path metastasis pm,' 'ajcc path tumor stage,' and 'grade.' Numerical embeddings are used for 'age diag.'

4.3. Implementation of Downstream tasks

Mortality Mortality prediction serves as a crucial clinical prognostic task in intensive care settings. This task aims to forecast in-hospital mortality by predicting whether a patient will survive their hospital stay. Specifically, we leverage multimodal clinical data collected during the first 48 hours post-admission, to generate a binary classification outcome indicating the patient's survival status at discharge. This early prediction capability is particularly valuable for clinical decision-making and resource allocation, as it enables healthcare providers to identify high-risk patients and implement timely interventions during the critical initial period of hospitalization.

Longstay The length of patient stay (Longstay),

defined as the duration between admission and discharge. We formulate this as a binary classification task to predict extended hospitalizations. Specifically, utilizing multimodal data from the first 48 hours of admission, our models aim to identify patients likely to require prolonged hospitalization (>7 days). To establish a more distinct classification boundary and reduce ambiguity in our analysis, we excluded cases with hospital stays shorter than 3 days. The prediction outcome is binary, where positive cases represent stays exceeding 7 days, and negative cases indicate shorter durations (3-7 days). This predictive capability is particularly valuable for hospital administrators and clinicians, as early identification of potentially extended stays enables more efficient resource allocation and care planning.

Readmission Readmission, defined as an unplanned return to hospital within 30 days of discharge, represents a significant healthcare quality indicator. We formulate this as a binary classification task using multimodal data collected during the first 48 hours of the initial hospitalization. The model aims to predict whether a patient will require readmission within 30 days following their discharge. This early prediction capability is particularly valuable for healthcare providers, as it enables the identification of high-risk patients and the implementation of preventive interventions before discharge, potentially reducing unnecessary readmissions and improving patient care quality.

Vital-12 Vital 12 is a task aimed at predicting patient survival within 12 months, representing a critical indicator of long-term outcomes in healthcare. We formulate this as a binary classification task using multimodal data collected during the initial hospitalization period. The model predicts whether a patient will survive beyond 12 months after their initial diagnosis.

5. Data ratio Experiments

While we presented the detailed analysis of the longstay prediction task in the main text, we conducted data ratio experiments across all tasks in the MIMIC-IV-MM dataset. Here, we report the results for the other two critical tasks: mortality (Fig. 1) and readmission (Fig. 2) prediction. These results demonstrate the models' performance and robustness under different data availability scenarios, complementing our main findings from the longstay task analysis.

6. Stability Experiments

To evaluate the robustness and reliability of our proposed model, we conducted stability experiments on the MIMIC-IV-MM dataset. As shown in Tab. 5, we performed five independent runs using identical parameter settings and calculated the mean and standard deviation for each metric. The results demonstrate strong stability across all three tasks (mortality prediction, longstay prediction, and readmission prediction) and evaluation metrics. The consistently small standard deviations across all metrics suggest that our model produces reliable and reproducible results regardless of random initialization, indicating the robustness of our approach.

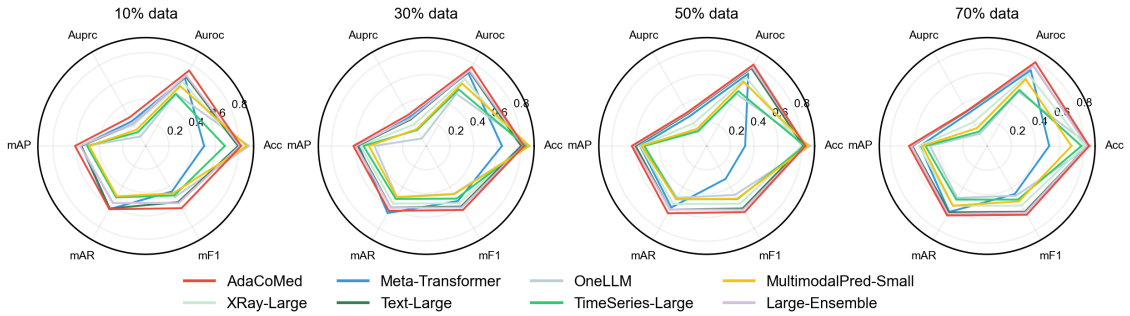

Figure 1. Performance comparison of different methods on different training data ratios of MIMIC-IV-MM Mortality task. Each subplot represents the performance of various training data ratios (10%, 30%, 50%, 70%). The radial axes show the performance value for each metric, ranging from 0 to 0.8 with intervals of 0.2.

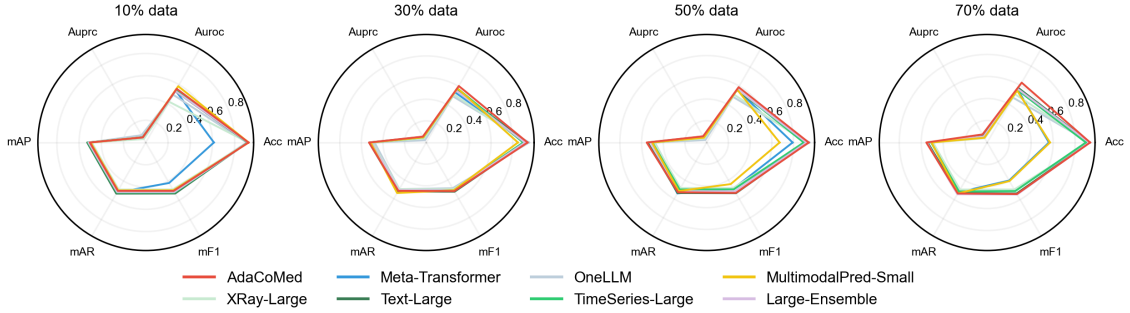

Figure 2. Performance comparison of different methods on different training data ratios of MIMIC-IV-MM Readmission task. Each subplot represents the performance of various training data ratios (10%, 30%, 50%, 70%). The radial axes show the performance value for each metric, ranging from 0 to 0.8 with intervals of 0.2.

Task	Acc	Auroc	Auprc	mAP	mAR	mF1
Mortality	0.8502±0.0066	0.7948±0.0058	0.3636±0.0022	0.6448±0.0161	0.6334±0.0128	0.6386±0.0143
Longstay	0.6792±0.0039	0.7316±0.0009	0.7774±0.0026	0.6696±0.0026	0.6702±0.0018	0.6698±0.0024
Readmission	0.9390±0.0045	0.6054±0.0076	0.0688±0.0063	0.5354±0.0013	0.5186±0.0036	0.5224±0.0031

Table 5. Stability experiment on the MIMIC-IV-MM dataset. Each value represents the mean ± standard deviation from five runs using the same set of parameters to evaluate the stability of our model.