Adaptive Test-Time Training For Predicting Need For Invasive Mechanical Ventilation In Multi-Center Cohorts

Xiaolei Lu , Shamim Nemati

Divisions of Biomedical Informatics University of California San Diego La Jolla, CA 92122

Background

- Acute Hypoxemic Respiratory Failure (AHRF) is a life-threatening condition requiring intensive care unit (ICU) care, often caused by conditions like ARDS and pneumonia

- Effective respiratory support is critical to prevent disease progression and reduce mortality

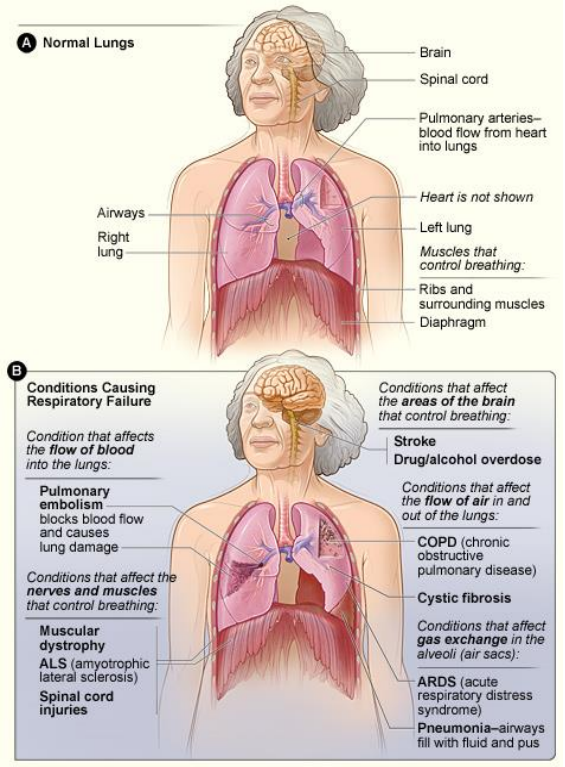


Fig.1. Illustration of normal lung anatomy and functions (A) compared to conditions causing respiratory failure (B), such as ARDS, pneumonia, and other impairments affecting blood flow, airway mechanics, or gas exchange. Image source: https://en.wikipedia.org/wiki/Respiratory_failure

Why early prediction of IMV need matters

- **Proactive intervention**
  - Identify high-risk patients early to initiate optimal respiratory support (e.g., HFNC or NIV)
- **Timely escalation**
  - Prevent delays in transitioning to advanced therapies when needed
- **Better patient outcomes**
  - Avoid unnecessary IMV and associated complications, improving recovery and survival rates

Challenges in predicting and personalizing respiratory support

- **Identifying high-risk patients**
  - **Clinical variability**: Patient heterogeneity in disease progression complicates identifying those at risk of IMV
  - **Incomplete data**: Missing or delayed clinical information impacts early prediction accuracy
  - **Existing approach**: Traditional methods (e.g., ROX index) fail to capture complex physiological relationship

Vent.io: A machine learning model for predicting respiratory failure

- Build a machine learning model to predict IMV need 24 hours in advance for critically ill ICU patients
- Implement Vent.io in real-time using EHR data for silent evaluation in a live clinical environment
- Introduce a Predetermined Changed Control Protocol (PCCP) for model monitoring and adaptation

Lam, J. Y., Lu, X., Shashikumar, S. P., Lee, Y. S., Miller, M., Pour, H., Boussina, A. E., Pearce, A. K., Malhotra, A., and Nemati, S. Development, deployment, and continuous monitoring of a machine learning model to predict respiratory failure in critically ill patients. JAMIA open,7(4):ooae141, 2024.

Test-Time Training for model generalization

UC San Diego

Predictive models in real-world deployment

- Domain shifts across hospitals degrade performance (e.g., ~12% AUC drop in Vent.io between UCSD and MIMIC-IV datasets)
- Existing solutions like transfer learning and domain adaptation are computationally expensive and require labeled target data

Test-Time Training (TTT)

- Dynamically adapts models during prediction using an auxiliary task
- Requires no labeled target data or pre-training on target domain
- Addresses variability across institutions and patient populations

[1] Lam, J. Y., Lu, X., Shashikumar, S. P., Lee, Y. S., Miller, M., Pour, H., Boussina, A. E., Pearce, A. K., Malhotra, A., and Nemati, S. Development, deployment, and continuous monitoring of a machine learning model to predict respiratory failure in critically ill patients. JAMIA open,7(4):ooae141, 2024.

Adaptive Test-Time Training

- **Theoretical insights**
  - Derived mutual information bounds linking auxiliary and main tasks
- **Dynamic feature importance masking**
  - Prioritizes critical features for IMV prediction during SSL training
- **Prototype-augmented test-time alignment**
  - Enhances adaptability to unseen domains with Partial Optimal Transport

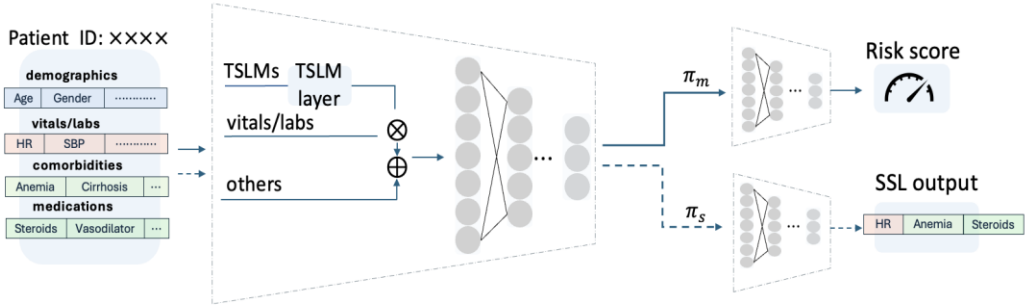


Fig.8. Adaptive Test-Time Training architecture.

Adaptive Test-Time Training

Dynamic SSL

- **SSL loss**
  - **Reconstruction:** $L_{\text{recon}} = \frac{1}{d}\sum_{j}(x_j - \hat{x}_j)^2$
  - **Masked feature modeling:** $L_{\text{mfm}} = \frac{1}{|M|}\sum_{j\in M}(x_j - \hat{x}_j)^2$
- **Dynamic feature masking**
  - **Compute global importance:** $I_j = \frac{1}{n_s}\sum_{n}\left|\frac{\partial Y}{\partial x_j}x_j\right|$
  - **Convert to mask prob:** $p_{m,j} = \frac{I_j - \min I}{\max I - \min I}$

Adaptive Test-Time Training

Prototype-Guided Adaptation (Training Stage)

- Prototype learning

  For each feature embedding z_i:

$$L_{\text{proto}}(z_i; \mathbf{P}) = \|z_i - p_{\mathcal{A}(z_i)}\|_2^2$$

- Prevent collapse (balanced assignment)

  Avoid assigning all samples to one prototype:

$$L_{\text{reg}}(\mathbf{P}) = \sum_{j=1}^{k} \left(\frac{1}{n_s} \sum_{i=1}^{n_s} \mathbb{I}(\mathcal{A}(z_i) = j) - \frac{1}{k} \right)^2$$

- Final training objective

$$L = \sum_{i=1}^{n_s} [L_{\text{main}}(x_i, y_i) + L_{\text{ssl}}(x_i) + \lambda_{\text{proto}} L_{\text{proto}}(z_i)] + \lambda_{\text{reg}} L_{\text{reg}}(\mathbf{P})$$

Adaptive Test-Time Training

POT-Based Test-Time Adaptation

- Create augmented test embeddings

  Augment the test embedding with k-1 noise-perturbed duplicates: $z' = \{z', z'_1, ..., z'_{k-1}\}$

  Each duplicate: $z'_{j,d} \sim z'_d + \mathcal{N}(0, \sigma_d^2)$

  Variance from prototype spread: $\sigma_d^2 = \frac{1}{k}\sum_{j=1}^{k}(p_{j,d} - \mu_d)^2$

- POT loss (Partial Transport)

$$L_{\text{test}} = L_{\text{ssl}} + \lambda_{\text{ot}} \sum \gamma_{ij} C_{ij}$$

  - γ_{ij}: transported mass
  - $C_{ij} = \|z'_i - p_j\|_2^2$: cost to match prototype
  - POT → only part of test embedding aligns to prototypes (prevents overfitting)

Adaptive Test-Time Training

POT-Based Test-Time Adaptation

- Create augmented test embeddings

  Augment the test embedding with k-1 noise-perturbed duplicates: $z' = \{z', z'_1, ..., z'_{k-1}\}$

  Each duplicate: $z'_{j,d} \sim z'_d + \mathcal{N}(0, \sigma_d^2)$

  Variance from prototype spread: $\sigma_d^2 = \frac{1}{k}\sum_{j=1}^{k}(p_{j,d} - \mu_d)^2$

- POT loss (Partial Transport)

$$L_{\text{test}} = L_{\text{ssl}} + \lambda_{\text{ot}} \sum \gamma_{ij} C_{ij}$$

  - γ_{ij}: transported mass
  - $C_{ij} = \|z'_i - p_j\|_2^2$: cost to match prototype
  - POT → only part of test embedding aligns to prototypes (prevents overfitting)

Adaptive Test-Time Training

Main results

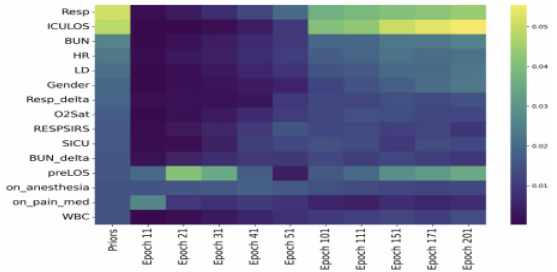


Figure 2: Feature importance evolution during training. The heatmap shows the changes in feature importance in the initial epochs and final epochs.

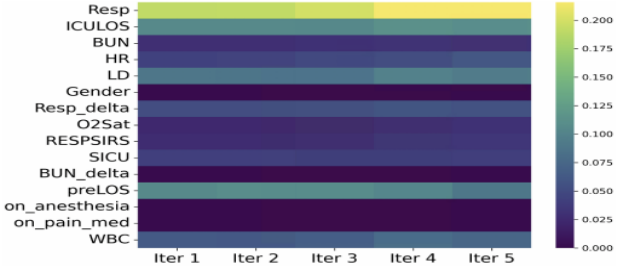
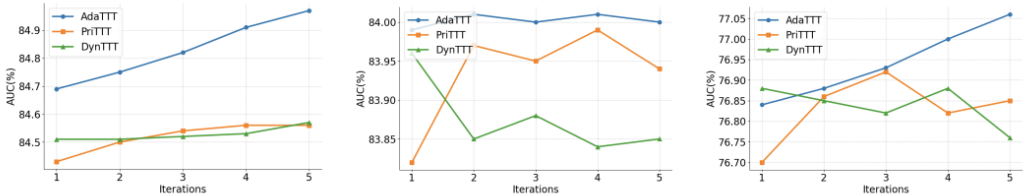


Figure 3: An example of feature importance evolution during test-time training. The heatmap shows the changes in feature importance across different iterations.

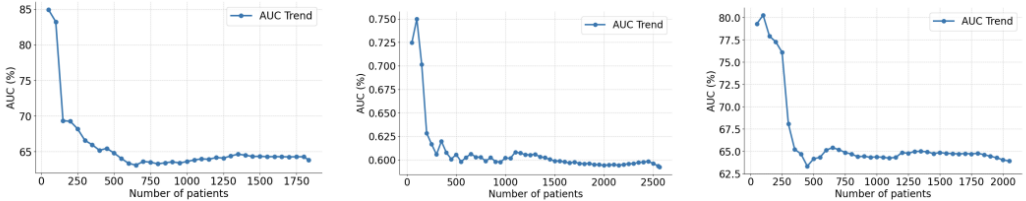
Adaptive Test-Time Training

Main results



(a) Site A. (b) Site B. (c) MIMIC-IV.

Figure 4: Evaluation of the number of gradient updates for test-time training on different test cohorts.



(a) Site A. (b) Site B. (c) MIMIC-IV.

Figure 5: Cumulative AUC trend over an increasing number of patients.

Conclusion

- We propose **AdaTTT** for test-time adaptation in EHR
- Combines:
  - Dynamic self-supervision
  - Prototype-guided alignment
  - Partial Optimal Transport
- Achieves **consistent gains across multi-center cohorts**
- Enables **robust deployment under domain shift**

UC San Diego

THANK YOU!

Appendix: Vent.io TSLM layer

- TSLM layer gives each clinical feature a **learnable time-decay function**:

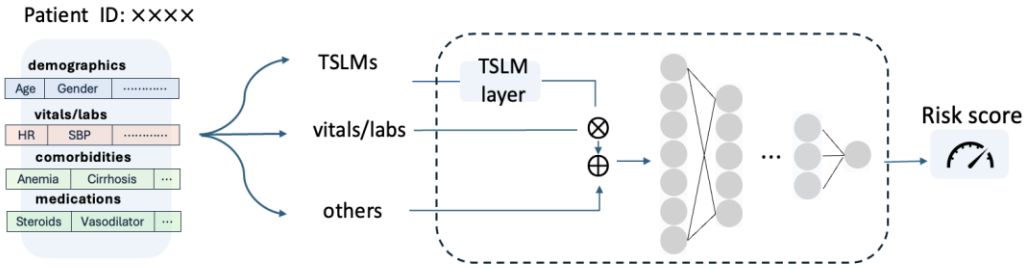


Fig.5.Vent.io architecture.

- when a feature was last updated
- the irregular sampling pattern
- the recency trend of each variable
- and how quickly past measurements become obsolete

$$x'_j = x_j \cdot 2(1 - \sigma(\theta_j^2 \cdot \mathrm{tslm}_j))$$

These are the clinically meaningful parts of “history” in ICU data.