PRIOR: Prototype Representation Joint Learning from Medical Images and Reports (Supplementary Material)

Pujin Cheng [1,2], Li Lin [1,3], Junyan Lyu[1,4], Yijin Huang [1,5],
Wenhan Luo [6], Xiaoying Tang [1,2,✉]
[1]Department of Electronic and Electrical Engineering, Southern University of Science and Technology
[2]Jiaxing Research Institute, Southern University of Science and Technology
[3]Department of Electrical and Electronic Engineering, The University of Hong Kong
[4]Queensland Brain Institute, The University of Queensland
[5]School of Biomedical Engineering, University of British Columbia
[6]Shenzhen Campus of Sun Yat-sen University

1. Introduction

In this document, we provide supplementary material for our paper "PRIOR: Prototype Representation Joint Learning from Medical Images and Reports". We first provide pre-training details. Then we present details of fine-tuning on different downstream tasks. Finally, we perform detailed analysis of each key component in PRIOR, including sentence-wise prototype memory bank (SPB), local alignment module (LAM), and cross-modality conditional reconstruction (CCR).

2. Pre-training Details

2.1. Data Preprocessing

We pre-train the proposed PRIOR on the MIMIC-CXR-JPG dataset. We remove all lateral-view images and images whose corresponding reports have fewer than four sentences. Finally, we end up with 182475 image-report pairs.

For image preprocessing, we first normalize the intensities of all images into $[0, 1]$ and then calculate their mean (0.4755) and standard deviation (0.3011). Z-score normalization is employed. All images are resized to 224×224 before being fed into a model of interest.

For report preprocessing, we utilize all sentences from the *Findings* and *Impression* sections. We employ BioClinicalBERT's tokenizer [1] implemented in the Transformer library [11] to tokenize each report into a sequence of tokens. We then pad all reports to have the same length of 256 tokens.

✉ Corresponding author: <tangxy@sustech.edu.cn>.

2.2. Single-modality Encoder Architecture

We employ ResNet50 as the image encoder, which is implemented in the TorchVision library [6] and gets pre-trained on ImageNet. For global representation, we apply an attention pooling layer to obtain a 2048-dimensional vector. For local representation, we take the feature map $f \in \mathbb{R}^{7\times7\times2048}$ of the last layer of ResNet50.

We adopt BioClinicalBERT [1] as the report encoder, which is implemented in the Transformer library [1] and gets pre-trained on the MIMIC-CXR dataset. We take the hidden state from the last layer as the token-level representation. After that, we gather all token representations in the same sentence via self-attention pooling to serve as the sentence-level representation $q_i \in \mathbb{R}^{M_R^i \times 768}$, where M_R^i is the number of sentences in the i-th report. Finally, we use an additional self-attention pooling layer to obtain the global representation over all sentence-level representations.

To embed linguist and visual representation into the same dimension, we attach four independent MLPs to embed both local and global report/image representation into an embedding space of dimension 768.

2.3. Cross-modality Interaction Architecture

The cross-modality interaction architecture mainly contains three components, namely sentence-wise prototype memory bank, cross-modality alignment, and cross-modality conditional reconstruction.

Sentence-wise prototype memory bank. Each prototype in SPB is initialized with the Standard Gaussian distribution and L1 normalization. For stable convergence, the temperature coefficient in Gumbel-softmax reparameterization decays from 0.9 to 0.01.

Cross-modality alignment. We employ the contrastive loss implemented in PyTorch-Lightning [3] as the global alignment loss and the local image-to-report alignment loss. We adopt the loss proposed elsewhere [2] as the local report-to-image alignment loss.

Cross-modality conditional reconstruction. We adopt five up-sampling blocks for image reconstruction. Each block consists of two convolution operations with a size of 3×3 and a stride of 1×1, followed by a ReLU activation function and a batch normalization layer. For report reconstruction, we adopt a decoder with 6 BERT layers in the same setting as that of BioClinicalBERT [1]. We employ the bipartite matching algorithm to match the predicted prototypes with the ground-truth ones, which is implemented in the SciPy library [10].

2.4. Training Details

We train PRIOR for 100 epochs on NVIDIA A100 GPUs with a batch size of 128. The total training process consists of three stages: (1) In stage 1, we train the cross-modality alignment module for 20 epochs, which is directly trained on sentence-wise representation without SPB. (2) In stage 2, we jointly train SPB and the cross-modality alignment module for 30 epochs. (3) In stage 3, we jointly train SPB, the cross-modality alignment module, and the cross-modality conditional reconstruction module for 50 epochs. We use the Adam optimizer [5] with a learning rate of $1e-5$ and a weight decay of $1e-6$. A cosine annealing scheduler is employed to adjust the learning rate. We employ grid search to identify the optimal hyper-parameter combination that performs the best on downstream tasks.

3. Fine-tuning Details

We evaluate the proposed PRIOR on five downstream tasks, namely supervised classification, zero-shot classification, image-to-text retrieval, semantic segmentation, and object detection. For each task, we employ grid search to identify the best-performing hyper-parameters. We here only provide the specific hyper-parameters for VLP methods since they are our essential objects of interest; all VLP methods share the same set of hyper-parameters. For hyper-parameters employed in non-VLP methods, please refer to our source code for details. The intensity of each image is normalized into $[0, 1]$ via Z-score normalization using the mean and standard deviation of the training set. We resize all images into 224×224. A cosine decay scheduler is used to adjust the learning rate.

3.1. Supervised Classification

For the downstream task of supervised classification, we fine-tune the image encoder with an additional fully-connected layer on the RSNA dataset, the SIIM Pneumothorax dataset, and the CheXpert dataset. The hyper-parameters of all VLP methods are listed in Table 1. Since each dataset involves either binary classification or multi-label classification, we use the binary cross-entropy loss as the loss function. The model with the highest AUC-ROC on the validation set is selected for testing.

Table 1. Hyper-parameter details for supervised classification.

Dataset	Training Data Ratio	Learning Rate	Epochs	Batch Size
RSNA	100%	$1e-4$	5	64
	10%	$1e-4$	5	64
	1%	$1e-4$	10	64
SIIM	100%	$1e-5$	5	64
	10%	$1e-5$	20	64
	1%	$1e-4$	20	64
CheXpert	100%	$1e-6$	5	256
	10%	$1e-5$	5	256
	1%	$1e-4$	5	256

3.2. Zero-shot Classification

We conduct zero-shot classification on the CheXpert 5x200 dataset, which is based on the similarity between manually designed prompt sentences and images. We generate the prompt simply through using "*X is observed*", where *X* represents the name of a specific disease. For generalization, we do not use any ensemble method. We calculate the similarity between the prompt and the image via the sum of the global alignment loss and the local cross attention weight. Note that before summing the two scores, we employ the softmax function to normalize each of them into $[0, 1]$.

3.3. Image-to-text Retrieval

Similar to zero-shot classification, we conduct image-to-text retrieval on the CheXpert 5x200 dataset, which is based on the similarity between queried reports and images. Note that because CheXpert does not have public reports, we sample 1000 exclusive reports from MIMIC-CXR dataset for each of 5 diseases.

3.4. Semantic Segmentation

We adopt U-Net [8] as the segmentation network. Different from the original U-Net, we employ U-Net equipped with a ResNet50 backbone, which is implemented in the Segmentation Models Pytorch library [4]. We present the hyper-parameter details in Table 2. The Dice loss is used for training, which performs better than either the cross-entropy loss or a combination of the two losses according to our experimental results. The model with the highest Dice score on the validation set is selected for testing.

3.5. Object Detection

We adopt Faster-RCNN [7] as the object detection network. The training and evaluation scripts are based on Pytorch-Lightning Flash [3]. The hyper-parameter details

Table 2. Hyper-parameter details for semantic segmentation and object detection.

Dataset	Training Data Ratio	Learning Rate	Epochs	Batch Size
SIIM Segmentation	100%	$1e-3$	50	128
	10%	$1e-4$	50	128
	1%	$1e-4$	100	16
RSNA Detection	100%	$5e-5$	5	16
	10%	$1e-4$	20	16
	1%	$1e-4$	5	16

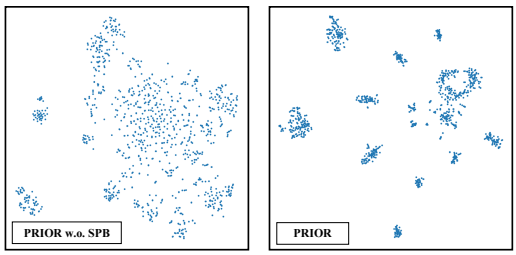

Figure 1. The t-SNE visualization comparison between the sentence-level embeddings without and with SPB pre-training.

are also tabulated in Table 2. We follow the original Faster-RCNN setting in terms of the loss function [7]. The model with the highest mAP (0.5:0.95) on the validation set is selected for testing.

4. Component Analysis

In this section, we perform further analysis on each component of PRIOR, including SPB, LAM, and CCR.

4.1. Analysis on SPB

To demonstrate the effectiveness of SPB, we utilize several synonymous sentences to demonstrate the querying distribution over the memory bank, as shown in Figure 2. It is worth noting that SPB highlights the most relevant prototype in a sharp distribution over SPB, which means that SPB can effectively capture the high-level semantic information of the sentence of interest.

Furthermore, we visualize the sentence-wise embeddings via t-SNE [9] on 1000 randomly selected sentences from MIMIC-CXR in Figure 1. Apparently, the sentence-wise embeddings with SPB pre-training are more compact and well-separated. In other words, SPB significantly improves the quality of the sentence embeddings, which benefits cross-modality interaction.

4.2. Analysis on LAM

In Figure 3, we visualize the attention weights of LAM on CheXpert 5x200. We first generate a sentence in the form of "*X is observed*", where *X* represents the name of a specific disease. Then the cross-modality attention map is obtained by the proposed LAM over the generated sentence and the corresponding image. Clearly, LAM can effectively localize the affected region of the given disease.

Table 3. The influence of the activation functions in LAM.

Activation function	CheXpert Classification	SIIM Segmentation
Softmax	86.01 ± 0.68	45.65 ± 1.31
Sigmoid	**86.16 ± 0.64**	**46.01 ± 1.03**

Unlike natural image captioning, some sentences in medical reports are irrelevant to medical images, such as "AP single view of the chest has been obtained". Softmax assigns a constant probability summing up to 1 when aligning sentence with image, resulting in an average representation of image and failing to represent local features precisely. Instead, Sigmoid assigns probability to each pixel separately. Our experiments demonstrate the effectiveness of the Sigmoid function in cross-modality attention, as shown in Table 3.

4.3. Analysis on CCR

CCR is a key component in our PRIOR, which can effectively capture fine-grained cross-modality information. We visualize representative reconstructed images from the CCR module in Figure 4. We find that the reconstructed images can well maintain the low-level information that is related to the report descriptions. Meanwhile, the reconstructed images successfully reveal the severity of the corresponding disease as well as the lesion locations.

For sentence prototype reconstruction, we compare the query distribution of the original report and the predicted distribution from CCR in Figure 5. The distribution is obtained from all sentences in the report of interest. We observe that the predicted distribution is similar to the original distribution, which demonstrates that the CCR module can effectively capture cross-modality information and reconstruct sentence prototypes.

References

[1] Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*, 2019. 1, 2

[2] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021. 2

[3] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019. 2

[4] Pavel Iakubovskii. Segmentation models pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2019. 2

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[6] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th*

ACM international conference on Multimedia, pages 1485–1488, 2010. 1

[7] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 2, 3

[8] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2

[9] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 3

[10] Pauli Virtanen and Ralf etal Gommers. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. 2

[11] Thomas Wolf and Lysandre Debut et al. Transformers: State-of-the-art natural language processing. pages 38–45, Online, Oct. 2020. Association for Computational Linguistics. 1

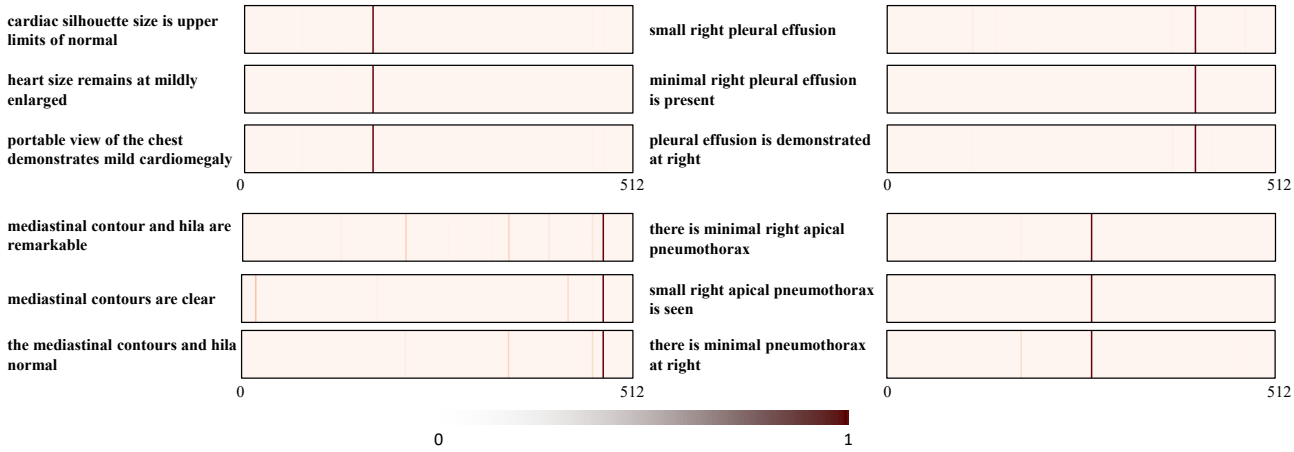

Figure 2. Representative examples demonstrating that SPB can cluster sentences with similar information. The horizontal axis represents the sentence index in the memory bank, and the color bar quantifies the querying score.

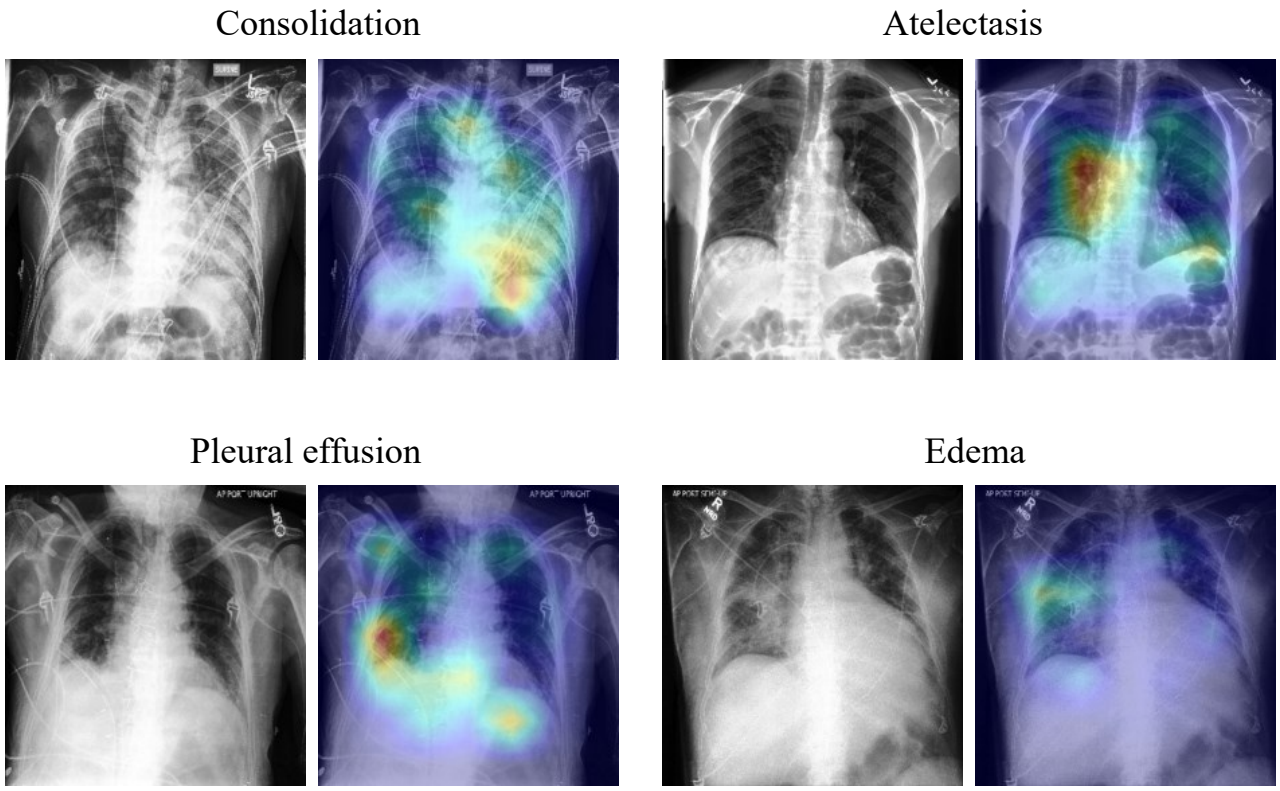

Figure 3. Representative examples of cross-modality attention maps related to different diseases.

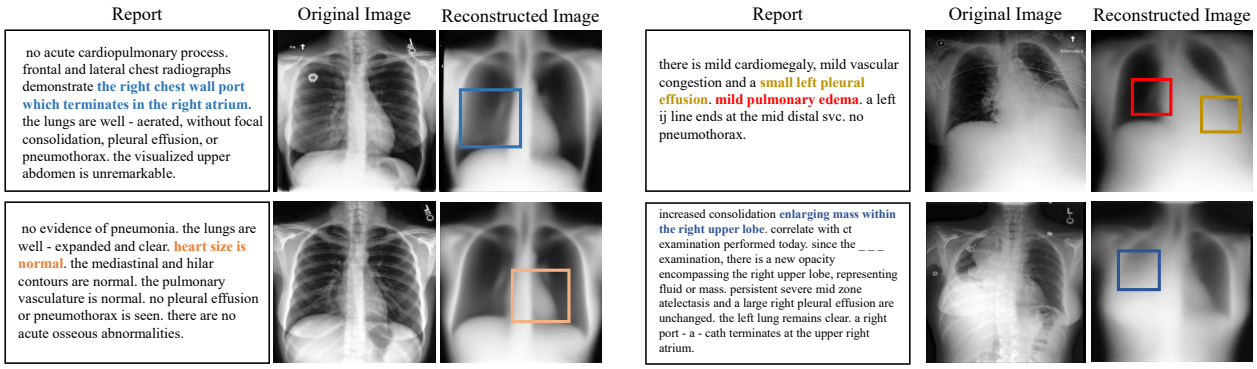

Figure 4. Representative examples of reconstructed images from CCR.

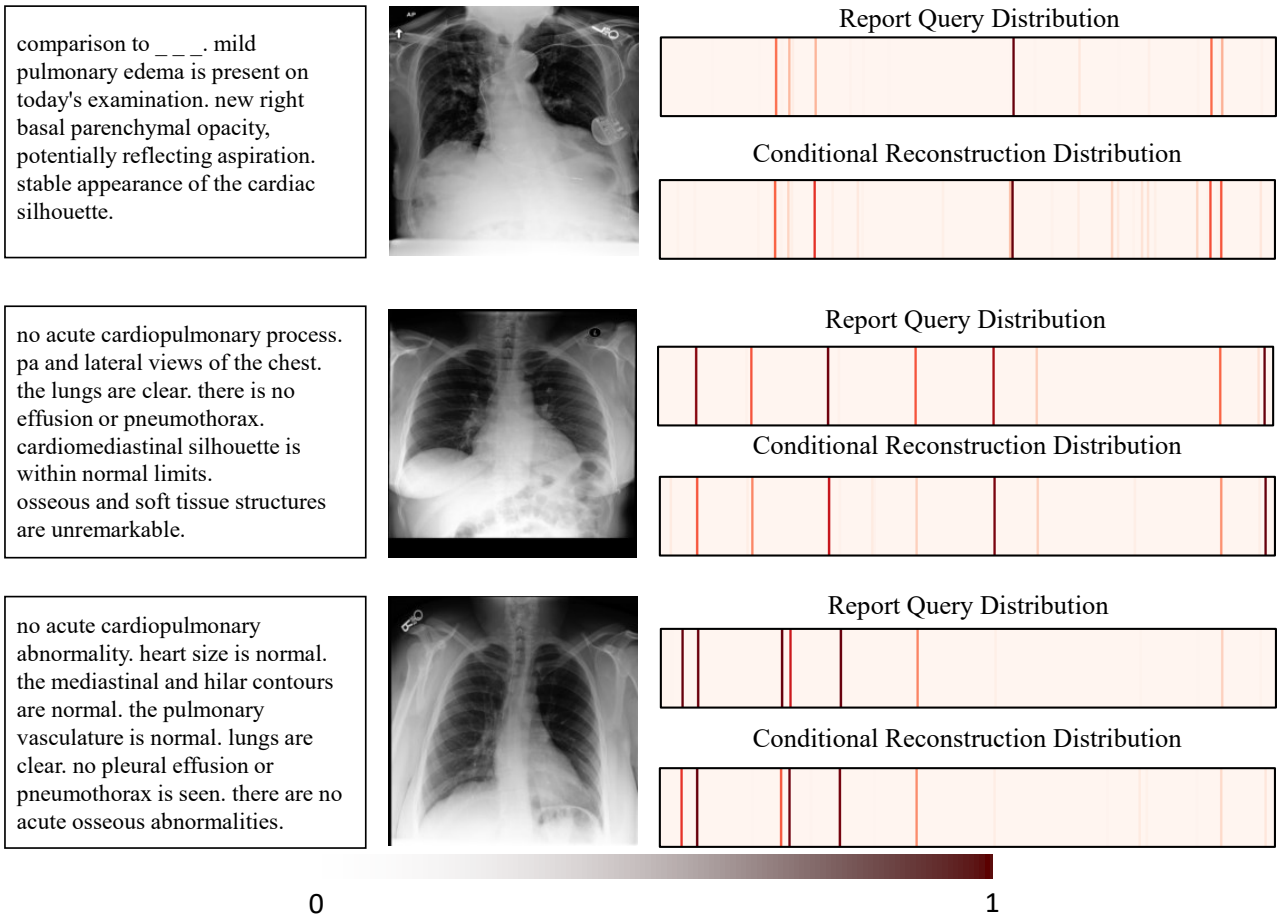

Figure 5. Representative examples of conditional reconstruction for sentence-wise prototypes. From left to right, the first column shows the original reports, the second column shows the original images, the top panel of the third column shows report distributions over the memory bank, and the bottom panel of the third column shows the predicted sentence representation distributions. Note that the top and bottom panels of the third column are very similar to each for all provided examples.