7. Supplementary

7.1. Detail related work

Learnable number embedding. A straightforward approach for incorporating numerical values is using learnable networks to continuously project numerical values into and out of the d-dimensional embedding space. This approach typically involves two small neural networks: a number encoder and a number decoder.

Specifically, the encoding function $\mathcal{G}_{enc}$ uses a number encoder, $f_{enc} : \mathbb{R} \to \mathbb{R}^d$, which maps the input number m to a d-dimensional vector. This vector is then combined with the base embedding for the [NUM] token, for instance, through addition. The decoding function $\mathcal{G}_{dec}$ uses a number decoder, $f_{dec} : \mathbb{R}^d \to \mathbb{R}$, which performs the reverse operation: it takes the final hidden state from the model and projects it back to a scalar numerical value. Both f_{enc} and f_{dec} are typically implemented as Multi-Layer Perceptrons (MLPs). The mapping functions are thus defined as:

$$\begin{aligned} \mathcal{G}_{enc}(\mathbf{x}_{[\mathsf{NUM}]}, m) &= \mathbf{x}_{[\mathsf{NUM}]} + f_{enc}(m) \\ \mathcal{G}_{dec}(\hat{\mathbf{x}}_i) &= f_{dec}(\hat{\mathbf{x}}_i), \\ &\text{if } [\mathsf{NUM}] = \arg\min_{\mathcal{V}} f_{lm}(\hat{\mathbf{x}}_i), \end{aligned} \tag{15}$$

where f_{lm} refers to original language modeling head from LLM.

Although this approach is notable for its simplicity as it parametrizes both mapping functions with learnable MLPS and can theoretically satisfies **C1** requirement of "wide-range of value" and **C3**, it introduces two critical challenges. First, an MLP does not inherently guarantee that the ordinal relationship of numbers is preserved in the high-dimensional embedding space. There is no inductive bias to ensure that the embedding for a number (e.g. the number 3 is closer to 4 than to 30. I.e., the embedding $f_{enc}(3)$ is not guaranteed to be closer to $f_{enc}(4)$ than it is to $f_{enc}(30)$.). The learned geometric relationships can be arbitrary without additional constraints, which makes **C1** "ordinal-preserving" requirement difficult to satisfy. Second, and more fundamentally, the projection network suffers from poor out-of-distribution (OOD) generalization and prone to context noise. An MLP trained on numbers within a specific range cannot be expected to produce meaningful embeddings for numbers it has not seen during training. Furthermore, during decoding, the MLP may struggle to accurately reconstruct the original number from the hidden state, especially if the hidden state has been influenced by other contextual information. Therefore, this approach cannot reliably satisfy part of **C1** and **C2**, making it brittle for applications with unbounded numerical data.

Magnitude-based scaling. The xVal [13] framework injects numerical values by scaling the embedding magnitude of the special token. To encode a numerical value m_i, the method multiplies the base embedding $\mathbf{x}_{[\mathsf{NUM}]}$ with a normalized version of the value, such that the norm of the resulting embedding scales with the magnitude of m_i. During training, a separate regression head f_{num} is trained with a Mean-Squared-Error (MSE) loss to predict the original value m_i from the final hidden state. During inference, a standard language model head f_{lm} predicts the next token. If it predicts [NUM], the number head f_{num} is triggered to generate the numerical value $\hat{m}_i$ from the corresponding hidden state $\hat{\mathbf{x}}_i$. The $\mathcal{G}_{enc}$ and $\mathcal{G}_{dec}$ functions for this approach are defined as:

$$\begin{aligned} \mathcal{G}_{enc}(\mathbf{x}_{[\mathsf{NUM}]}, m_i) &= \tanh(m_i) \cdot \mathbf{x}_{[\mathsf{NUM}]} \\ \mathcal{G}_{dec}(\hat{\mathbf{x}}_i) &= f_{num}(\hat{\mathbf{x}}_i), \\ &\text{if } [\mathsf{NUM}] = \arg\min_{\mathcal{V}} f_{lm}(\hat{\mathbf{x}}_i) \end{aligned} \tag{16}$$

Although xVal guarantees an **C1** ordinal relationship by directly scaling the embedding with the number's magnitude, it suffers from critical limitations. First, it cannot encode an unbounded range of numbers, as it requires a normalization function like $\tanh$ on the input number to prevent the embedding norm from exploding. This leads to value saturation, where the model cannot distinguish between different large numbers and fail for **C1** "wide-range value" requirement. More importantly, xVal was originally tested on Pre-LayerNorm (LN) [2] architectures like GPT-2 [34]. However, all recent LLMs turns to Post-LN due to training difficulties [42]. As pointed out by Brody et al. [6], the norm of a vector after going through LN is always $\sqrt{d}$, which diminished xVal's scaling utility to modern LLM architecture and failed at **C3**.

7.2. Algorithm outline

Here we describe the algorithm outlines for Rotary Number Encoding (RNE) in Alg. 1 and Decoding (RND) in Alg. 2.

Algorithm 1 Rotary Number Encoding (RNE)

Input: $\mathbf{x}_{[\mathsf{NUM}]} \in \mathbb{R}^d$, value m.
Output: Encoded $\mathbf{x}^m \in \mathbb{R}^d$.

procedure RNE($\mathbf{x}_{[\mathsf{NUM}]}, m$)
  $\Theta : \theta_j \leftarrow B^{-2j/d}$, for $j \in \{0, \ldots, d/2-1\}$.
  2D Rotation: $\mathbf{R}_2(m\theta_j) \leftarrow \begin{bmatrix} \cos m\theta_j & -\sin m\theta_j \\ \sin m\theta_j & \cos m\theta_j \end{bmatrix}$.
  d-dim Rotation: $\mathbf{R}_d(m\Theta) \leftarrow \mathrm{diag}(\mathbf{R}_2(m\theta_j))$.
  return $\mathbf{R}_d(m\Theta)\mathbf{x}_{[\mathsf{NUM}]}$.

7.3. Distinguishability–smoothness trade-off

We analyze how the weighting exponent p affects the geometry of the scalar score $\mathcal{S}(m, p)$. Let $B > 1$, even d, and

$$\begin{aligned} \theta_j &:= B^{-2j/d}, \quad j = 1, \ldots, d/2, \\ w_j(p) &:= \theta_j^{-p} \, \|\mathbf{x}_{[\mathsf{NUM}],j}\|^2. \end{aligned} \tag{17}$$

Algorithm 2 Rotary Number Decoding (RND)

Input: Model output $\hat{\mathbf{x}}_i \in \mathbb{R}^d$, candidates $\mathcal{M}$, parameter p.
Output: Decoded number $\hat{m}$.

procedure RND($\hat{\mathbf{x}}_i, \mathcal{M}, p$)
  $\Theta : \theta_j \leftarrow B^{-2j/d}$ for $j \in \{1, \dots, d/2-1\}$.
  Compute input signal: $\hat{s} \leftarrow \mathcal{S}(\hat{\mathbf{x}}_i, p)$.
  Compute candidate targets: $s'_{m'} \leftarrow \mathcal{S}(m', p)$ for each $m' \in \mathcal{M}$.
  Predict: $\hat{m} \leftarrow \arg\min_{m' \in \mathcal{M}} (\hat{s} - s'_{m'})^2$.
  return $\hat{m}$.

Then

$$\mathcal{S}(m,p) := \sum_{j=1}^{d/2} w_j(p) \cos(m\theta_j). \tag{18}$$

Lemma 7.1 (Derivative and oscillation scale)**.** *The derivative of the score with respect to m is*

$$\begin{aligned} \mathcal{S}'(m,p) &= -\sum_{j=1}^{d/2} w_j(p)\, \theta_j \sin(m\theta_j) \\ &= -\sum_{j=1}^{d/2} \|\mathbf{x}_{[\mathsf{NUM}],j}\|^2 \theta_j^{1-p} \sin(m\theta_j). \end{aligned} \tag{19}$$

Each term oscillates with period $T_j = 2\pi/\theta_j$. For $p < 1$, the factor θ_j^{1-p} increases with θ_j, so high-frequency components (θ_j large, short T_j) dominate $\mathcal{S}'$; for $p > 1$, $1-p < 0$ and low-frequency components (θ_j small, long T_j) dominate.

Lemma 7.1 shows that increasing p shifts $\mathcal{S}'(m,p)$ from high- to low-frequency components, making $\mathcal{S}(m,p)$ smoother in m.

Phase assumption. We now quantify typical changes in the score between nearby values of m. We assume that, as m ranges over a large interval, the phases $m\theta_j \bmod 2\pi$ explore $[0, 2\pi)$ approximately uniformly, so that

$$\mathbb{E}_m[\sin(m\theta_j)] \approx 0, \quad \mathbb{E}_m[\cos(m\theta_j)] \approx 0, \quad \mathbb{E}_m[\sin^2(m\theta_j)] \approx \tfrac{1}{2}. \tag{20}$$

Lemma 7.2 (Typical dot-product gap and RMS scale)**.** *Let $\Delta\mathcal{S}_\delta(m,p) := \mathcal{S}(m+\delta,p) - \mathcal{S}(m,p)$. Under the phase assumption,*

$$\mathbb{E}_m\left[\Delta\mathcal{S}_\delta(m,p)^2\right] \approx 2\sum_{j=1}^{d/2} w_j(p)^2 \sin^2\left(\tfrac{\theta_j\delta}{2}\right), \tag{21}$$

and for $\theta_j|\delta| \ll 1$ for all j,

$$\begin{aligned} \Delta_{\text{typ}}(\delta,p) &:= \sqrt{\mathbb{E}_m[\Delta\mathcal{S}_\delta(m,p)^2]} \\ &\approx |\delta|\, C(p), \quad C(p) := \left(\tfrac{1}{2}\sum_{j=1}^{d/2} w_j(p)^2\theta_j^2\right)^{1/2}. \end{aligned} \tag{22}$$

Moreover, the typical scale of the score is

$$\sigma_S(p)^2 := \mathbb{E}_m[\mathcal{S}(m,p)^2] \approx \tfrac{1}{2}\sum_{j=1}^{d/2} w_j(p)^2. \tag{23}$$

We measure distinguishability between consecutive integers by the *relative dot-product gap*:

Definition 1 (Relative dot-product gap)**.** *For consecutive integers ($\delta = 1$), define*

$$\begin{aligned} r(p) &:= \frac{\Delta_{\text{typ}}(1,p)}{\sigma_S(p)} \\ &\approx \sqrt{\frac{\sum_{j=1}^{d/2} w_j(p)^2\theta_j^2}{\sum_{j=1}^{d/2} w_j(p)^2}} \\ &= \sqrt{\frac{\sum_{j=1}^{d/2} \|\mathbf{x}_{[\mathsf{NUM}],j}\|^4 \theta_j^{2(1-p)}}{\sum_{j=1}^{d/2} \|\mathbf{x}_{[\mathsf{NUM}],j}\|^4 \theta_j^{-2p}}}. \end{aligned} \tag{24}$$

Proposition 1 (Distinguishability–smoothness trade-off)**.** *Assume at least two distinct θ_j have $\|\mathbf{x}_{[\mathsf{NUM}],j}\| > 0$. Then:*

1. *Increasing p shifts $\mathcal{S}'(m,p)$ towards lower frequencies (Lemma 7.1), so $\mathcal{S}(m,p)$ becomes smoother in m.*
2. *The relative gap $r(p)$ is strictly decreasing in p.*

Hence larger p produces smoother but less distinguishable scalar scores, and smaller p produces more oscillatory but more separated scores between neighboring values.

Proof. (1) follows directly from Lemma 7.1. For (2), write

$$\begin{aligned} r(p)^2 &= \frac{N(p)}{D(p)}, \\ N(p) &= \sum_{j=1}^{d/2} a_j \theta_j^{2(1-p)}, \\ D(p) &= \sum_{j=1}^{d/2} a_j \theta_j^{-2p}, \\ a_j &:= \|\mathbf{x}_{[\mathsf{NUM}],j}\|^4 > 0. \end{aligned} \tag{25}$$

Differentiating and symmetrizing over (j,k) gives

$$\begin{aligned} \frac{d}{dp} r(p)^2 &= \frac{N'(p)D(p) - N(p)D'(p)}{D(p)^2} \\ &= -\sum_{j<k} a_j a_k (\ln\theta_j - \ln\theta_k)(\theta_j^2 - \theta_k^2)\theta_j^{-2p}\theta_k^{-2p}. \end{aligned} \tag{26}$$

Since the θ_j are strictly decreasing, each term is negative whenever $a_j, a_k > 0$, so $r(p)^2$ and hence $r(p)$ are strictly decreasing in p. □

7.4. Precision-dependent choice of weighting p

We now consider a practical constraint on the choice of p imposed by the numerical precision of the floating-point format used to represent $\mathcal{S}(m, p)$. In modern LLMs, full precision (fp32) is often not used for inference due to computational and memory constraints. Instead, lower-precision formats such as fp16, bfloat16, or even fp8 are commonly employed. These formats have limited precision, which can lead to quantization errors when representing the scalar scores $\mathcal{S}(m, p)$.

We start relate the distinguishability measure $r(p)$ to the numerical precision of the floating-point format used to represent $\mathcal{S}(m, p)$. Recall that $\Delta_{\text{typ}}(1, p)$ denotes the typical change in the score between neighboring integers m and $m + 1$, and $\sigma_S(p)$ denotes the typical scale (RMS) of the score. Robust scalar decoding requires that the score change between neighbors is large compared to the quantization step of the format.

Unit roundoff and quantization step. To quantify the limits of numerical precision, we consider the **unit roundoff**, denoted by u. For a standard base-2 floating-point format with n_{sig} significand bits (including the implicit leading bit), the resolution is determined by the distance between 1.0 and next representable floating-point number. This distance is given by:

$$u = 2^{-(n_{\text{sig}}-1)}. \tag{27}$$

This value u represents the relative quantization error. Consequently, for a scalar score with a typical magnitude $\sigma_S(p)$, the absolute spacing between representable values is approximately $u \cdot \sigma_S(p)$.

In our setting, we want the typical separation between the scores m and $m + 1$ to be strictly resolvable, that is much larger than this quantization step:

$$\Delta_{\text{typ}}(1, p) \gg u, \sigma_S(p) \leftrightarrow r(p) := \frac{\Delta_{\text{typ}}(1, p)}{\sigma_S(p)} \gg u. \tag{28}$$

Theorem 7.3 (Precision-dependent constraint on p)**.** *For each floating-point format (fixed u), there exists a unique $p_{\text{crit}}(u)$ such that*

$$r\big(p_{\text{crit}}(u)\big) = u, \tag{29}$$

with

$$\begin{aligned} p < p_{\text{crit}}(u) &\;\Rightarrow\; r(p) > u \quad (\textit{neighbors well-separated}), \\ p > p_{\text{crit}}(u) &\;\Rightarrow\; r(p) < u \quad (\textit{neighbors can collapse}). \end{aligned} \tag{30}$$

Combined with Proposition 1, this yields a three-way tradeoff between smoothness, distinguishability, and numerical precision.

Proof. By Proposition 1, $r(p)$ is continuous and strictly decreasing in p. Hence, for any u in the range of r, there is a unique solution of $r(p) = u$, which we denote $p_{\text{crit}}(u)$. The implications for $p < p_{\text{crit}}(u)$ and $p > p_{\text{crit}}(u)$ follow directly from the inequalities $r(p) \gtrless u$. □

Corollary 1 (Recommended p for common precisions (example))**.** *For common IEEE-like formats, the effective spacing u is determined by the number of significand bits n_{sig}. The relationship is given by $u \approx 2^{-(n_{sig}-1)}$. For typical formats:*

- *fp32: $n_{sig} = 24 \Rightarrow u \approx 2^{-23} \approx 1.2 \times 10^{-7}$,*
- *fp16: $n_{sig} = 11 \Rightarrow u \approx 2^{-10} \approx 10^{-3}$,*
- *bfloat16: $n_{sig} = 8 \Rightarrow u \approx 2^{-7} \approx 8 \times 10^{-3}$,*
- *fp8 variants: $n_{sig} \approx 3{-}4 \Rightarrow u \approx O(10^{-2}{-}10^{-1})$.*

For a typical configuration ($B = 5 \cdot 10^5$, $d = 4096$) and roughly uniform block energies, numerical evaluation of $r(p)$ with these u values yields:

$$\begin{aligned} &\textit{fp32:} && r(p) \gg u \;\forall p \textit{ (no practical constraint);} \\ &\textit{fp16:} && r(p) \gg u \textit{ for } p \lesssim 0.3; \\ &\textit{bfloat16:} && r(p) \gg u \textit{ for } p \lesssim 0.2; \\ &\textit{fp8:} && r(p) \lesssim u \;\forall p \;\; (\textit{scalar score insufficient}). \end{aligned} \tag{31}$$

Conclusion: Thus, for scalar dot-product decoding, small p (e.g., $p \in [0, 0.3]$) is critical when using lower-precision formats like fp16 and bfloat16 to ensure the score separation exceeds the quantization noise. Conversely, fp32 imposes essentially no restriction, while fp8 possesses insufficient resolution for scalar decoding regardless of p, necessitating either full-vector decoding or higher-precision accumulation. This theoretical analysis justifies our ablation results in Tab. 4 and explains why $p \approx 0.2 - 0.3$ is optimal for optimal operating point for standard inference hardware.

7.5. Dataset curation

We curated a dataset with 25,645 non-contrast Chest computed tomography (CT) images from 14,218 unique patients, each with its corresponding radiology reports. The radiology report contains a "Findings" section and an "Impresson" section. The "Findings" section describes the detailed observations from the CT scan, while the "Impression" section provides a concise summary of the key findings and their clinical significance. For dataset splitting, we follow a 80-20 patient-wise approach to ensure that the training and testing sets are completely disjoint. We finally have 19,944 training samples with 11,059 patients and 5,701 testing samples with 3,159 patients.

To create a comprehensive training dataset for our model, we employed a two-stage generation process using two different LLMs, followed by CT-RATE [15] and HuatuoGPT [9] curation process. First, we used Llama-3.1 [14] to generate basic VQA pairs (multiple-choice, short-answer, summarization) for information extraction. Second, to enhance diversity, we used Gemini-2.5 [38] to simulate various clinical scenarios to generating questions with different tones and formats. The comprehensive generation process covering both simple queries and detailed, long-form responses, improving model's instruction following capabilities. The full generation process and examples are illustrated in Fig. 6.

For evaluation on radiology measurement estimation task, we combined traditional regular expression filtering and LLM based filtering to locate label of interested and their measurments. We first use regular expression to extract numerical values and their related text from the raw radiology report. Then we use LLM to filter out low-frequency labels and identify them as label of interested. The selected labels are: **Pulmonary Nodule**, **Ascending aorta**, **Aortic root**, **Aortic arch**, **Descending aorta**, **Aortic hiatus**, **Pulmonary Artery**, **Lymph Node**, **Thyroid nodule**, **Renal cyst** and **Adrenal lesion**. Then we use LLM to extract additional context related to the numerical value, forming meta-data query consists of number, unit, label and context. Finally, we use LLM to generate QA pairs based on meta-data. The full pipeline is illustrated in Fig. 5.

7.6. Medical Grounding Evaluation Details

We presented the full table from MedSeq-Bench [45] at Tab. 7. The table includes the performance of various general-purpose MLLMs across different grounding tasks, measured by Interaction-Over-Union (IOU) and Acc@5 metrics. IOU quantifies the overlap between the predicted bounding box and the ground truth bounding box. Acc@5 measure the proportion of predictions whose IOU with the ground truth exceeds 0.5. The results are categorized into two main sections: Image Difference Grounding and Image Consistency Grounding, with specific tasks listed under each category. The average performance across all tasks is also provided for each model.

7.7. Limitations

The score-matching decoder searches over a fixed candidate set $\mathcal{M}$, reintroducing a form of discretization; however, the step size of $\mathcal{M}$ is configurable per task and training uses a continuous regression loss. For large coordinate ranges (e.g., 0–1024 in high-resolution grounding), aliasing may occur at the highest rotational frequencies; this can be mitigated by adjusting p or adopting frequency scaling strategies analogous to YaRN. Currently, RNED supports only non-negative values; handling signed quantities (e.g., Hounsfield units) would require a sign-magnitude decomposition, which we leave as future work. Our primary evaluation uses the in-house Opport-CT dataset; broader validation on additional public benchmarks is an important future direction. Finally, RNE is applied only at the input embedding layer, without an explicit guarantee that intermediate transformer layers preserve the rotational structure; nonetheless, the end-to-end regression loss ensures that the output embedding carries sufficient numerical information for accurate decoding.

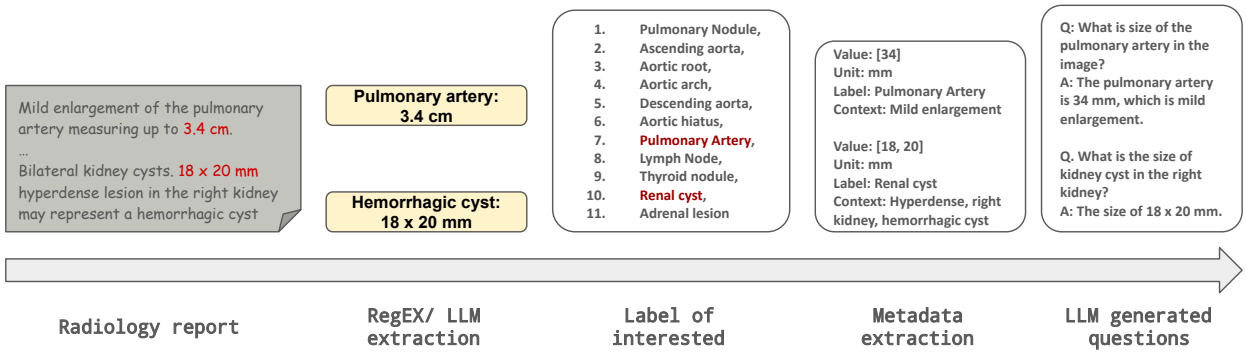

Figure 5. Pipeline for generating numerical decoding testing set. 1. We first using regular expression and LLM to extract numerical value and its related text from raw Radiology report. 2. Based on the extracted text, we conclude the 11 interested labels and remove low-frequency texts. 3. We extract additional context related to numerical value, forming meta-data query consists of number, unit, label and context. 4. We use LLM to generate QA pairs based on meta-data.

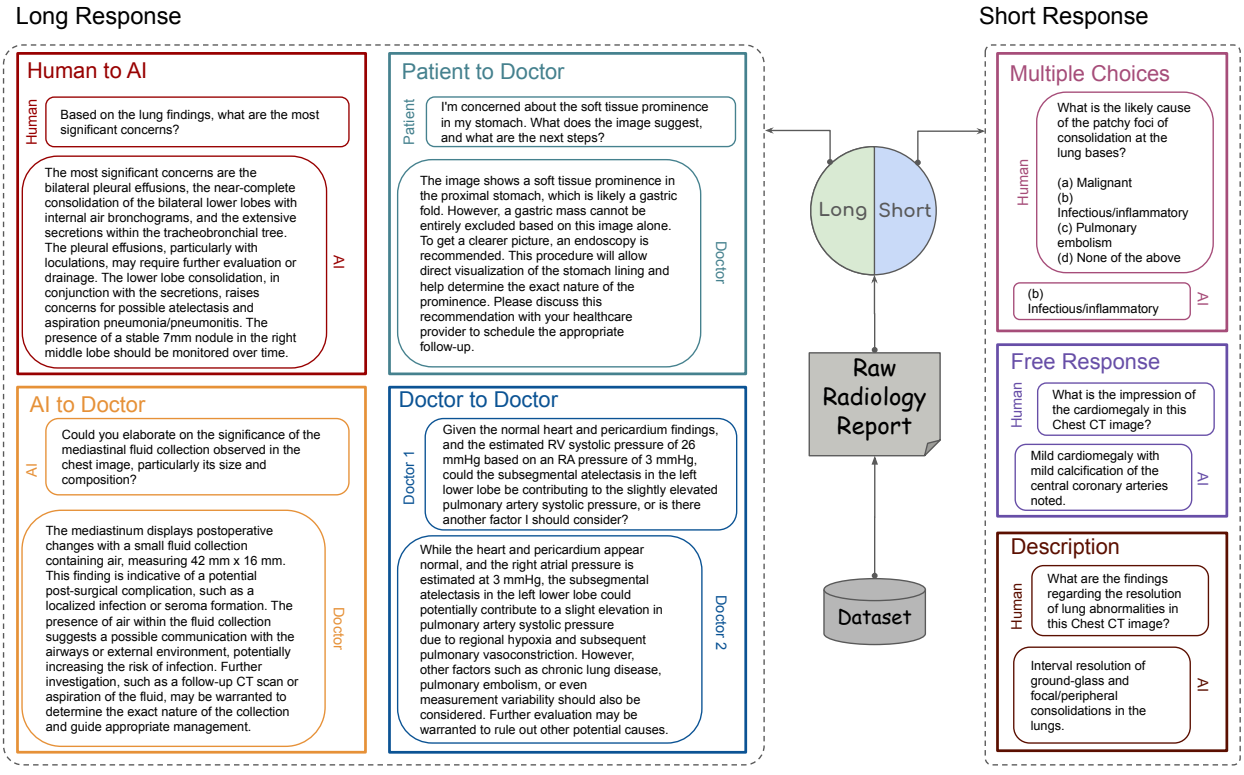

Figure 6. Sample figure illustrating the training data generation. Each Radiology report is augmented to generate multiple VQA examples that represent different perspectives. On the left, we have several examples of "long responses" that use a detailed description under different prompt scenarios that make model generated results be detailed and diverse. On the right, we show shorter question examples with multiple choice answers, a brief description and a free response that focuses on a specific information directly extracted from the report.

Model	Size	Metric	Image Difference Grounding		Image Consistency Grounding						Avg.
			Reg. DG	Non-reg. DG	Multi-view	Obj. Track.	Concept	Patch	Cross-modal	Referring	
General-purpose MLLMs											
Qwen2.5-VL [3]	3B	IoU	0.59	1.62	7.12	21.32	6.98	27.36	10.02	12.99	10.94
		Acc@5	0.30	1.30	3.90	16.80	0.80	3.40	1.65	6.82	4.20
Qwen2.5-VL [3]	7B	IoU	0.88	1.25	8.48	22.41	4.22	28.87	16.29	12.58	12.31
		Acc@5	0.30	0.00	3.73	17.80	1.00	5.70	4.45	6.21	4.90
Qwen2.5-VL [3]	32B	IoU	2.69	3.48	7.35	19.12	6.53	26.92	12.59	18.71	12.47
		Acc@5	1.40	1.20	2.61	13.40	1.30	7.10	4.90	11.67	5.71
Qwen2.5-VL [3]	72B	IoU	4.37	3.46	7.22	13.11	10.33	26.45	16.32	20.19	13.35
		Acc@5	2.60	0.80	2.78	7.70	3.50	6.30	7.00	14.10	6.12
MiniCPM-V-2_6 [43]	8B	IoU	1.36	1.50	15.82	24.03	9.90	28.65	12.72	12.44	13.24
		Acc@5	0.00	0.00	5.20	18.50	2.10	12.20	3.30	3.64	5.27
MiniCPM-O-2_6 [43]	8B	IoU	1.69	1.63	12.11	15.25	9.88	22.96	9.53	8.82	10.12
		Acc@5	0.10	0.00	2.43	9.60	1.70	9.20	2.35	2.02	3.23
mPLUG-Owl3 [44]	7B	IoU	2.12	2.55	15.64	15.62	6.80	30.42	17.06	11.92	13.22
		Acc@5	0.00	0.00	3.64	4.40	0.80	3.60	4.80	5.47	3.19
Mantis-Idefics2 [20]	8B	IoU	0.49	0.62	18.69	28.04	6.27	10.26	9.59	6.05	9.90
		Acc@5	0.00	0.00	8.59	23.50	0.50	1.10	0.95	0.54	3.91
LLaVA-OneVision [21]	7B	IoU	1.09	0.01	9.26	10.50	11.33	22.20	19.08	17.11	12.39
		Acc@5	0.00	0.00	1.13	3.20	1.80	5.30	6.70	5.67	3.47
LLaVA-OneVision [21]	72B	IoU	2.58	2.87	11.74	9.61	10.95	32.38	16.24	15.43	13.21
		Acc@5	0.80	0.90	1.39	2.30	3.30	20.30	5.40	6.68	5.18
InternVL2 [11]	8B	IoU	0.18	0.38	17.34	26.45	5.56	10.36	6.23	15.73	10.24
		Acc@5	0.00	0.00	7.03	21.20	0.80	0.70	1.00	7.69	4.59
InternVL2 [11]	76B	IoU	0.15	0.15	10.00	15.56	3.39	6.64	2.83	15.69	6.88
		Acc@5	0.00	0.00	3.90	11.80	0.40	1.10	0.75	9.92	3.53
InternVL2.5 [10]	8B	IoU	0.26	0.38	13.52	20.82	1.96	5.25	4.70	9.56	7.04
		Acc@5	0.00	0.00	3.56	13.80	0.00	0.00	0.85	3.44	2.56
InternVL2.5 [10]	78B	IoU	0.24	0.32	9.16	16.18	4.32	11.86	5.48	10.67	7.29
		Acc@5	0.10	0.10	2.08	10.00	0.50	2.30	1.25	4.52	2.55
InternVL3 [48]	8B	IoU	1.07	1.20	14.36	13.30	6.43	18.73	4.73	15.16	9.26
		Acc@5	0.30	0.00	4.42	6.50	0.90	4.60	1.15	7.42	3.19
InternVL3 [48]	14B	IoU	0.66	0.71	13.24	19.77	8.60	13.17	10.87	14.57	10.53
		Acc@5	0.00	0.00	5.31	13.00	2.10	2.40	3.70	7.76	4.41
InternVL3 [48]	38B	IoU	0.98	1.76	12.99	19.27	7.63	17.76	6.47	16.59	10.37
		Acc@5	0.10	0.20	4.79	13.60	2.10	2.90	1.75	10.05	4.44
InternVL3 [48]	78B	IoU	0.20	0.53	6.35	13.03	3.57	11.81	3.34	12.76	6.44
		Acc@5	0.00	0.00	2.43	8.00	0.90	2.50	0.85	8.10	2.90
Migician [24]	7B	IoU	15.26	14.49	18.16	21.38	14.23	28.87	21.41	25.30	20.29
		Acc@5	7.80	6.10	7.84	14.90	7.20	13.70	12.15	18.02	11.39
Medical-domain specialized MLLMs											
HuatuoGPT-Vision [9]	7B	IoU	1.35	1.84	10.42	14.57	7.99	15.52	9.46	9.60	8.97
		Acc@5	0.00	0.20	2.78	9.20	0.80	2.30	2.15	1.82	2.36
HuatuoGPT-Vision [9]	34B	IoU	1.44	2.15	9.41	13.25	6.43	14.53	10.60	8.60	8.57
		Acc@5	0.00	0.00	1.65	8.30	0.70	1.40	2.60	1.75	2.09
MedSeq-Grounder [45]	7B	IoU	**83.29**	**83.72**	55.03	62.10	74.11	**85.25**	78.77	60.43	72.55
		Acc@5	**93.20**	**94.10**	60.19	67.20	82.60	**98.80**	82.75	65.59	79.71
MedSeq-Grounder + RNED	7B	IoU	82.83	81.13	**59.26**	**63.15**	**75.37**	84.11	**83.54**	**64.29**	**74.21**
		Acc@5	92.30	91.95	**65.06**	**68.60**	**83.52**	96.80	**87.00**	**71.70**	**82.33**

Table 7. Performance on image difference and image consistency grounding tasks on MedSG-Bench. We report IoU and Acc@5.