SAT-RRG: LLM-Guided Self-Adaptive Training for Radiology Report Generation with Token-Level Push–Pull Optimization

Supplementary Material

1. Overview

This supplement provides expanded analysis of SAT–RRG. We first present a disease-wise breakdown, followed by an explanation of the unified LLM design and its computational advantages. We then detail the prompt for error tagging and provide deeper analysis of the training objective, including gradient behavior and numerical illustrations

2. Disease-wise Clinical analysis results across all 14 defined categories

Table A1 provides a disease-wise comparison between our model and the baseline across multiple lexical, semantic, and clinical metrics. Overall, our method achieves **consistent improvements across nearly all disease categories**, demonstrating strong robustness in capturing both textual quality and clinical accuracy. Notably, diseases with complex or subtle radiographic manifestations—such as *Atelectasis*, *Edema*, *Consolidation*, and *Pleural Effusion*—show clear gains in BLEU, ROUGE-L, and METEOR scores.

Semantic and clinical metrics (BERTScore, RadCliQ, and RadGraph) further reinforce this trend, with our model producing more clinically faithful entity relationships and structured findings. Importantly, improvements extend beyond common diseases: challenging or less frequent categories, including *Pneumothorax*, *Support Devices*, and *Cardiomegaly*, also exhibit measurable gains.

These results highlight that our fine-grained reward alignment enables **better generalization across heterogeneous clinical conditions**, improving both narrative accuracy and structured clinical fidelity.

3. Discussion: Using a Single LLM for Both Generation and Error Tagging

Using a single LLM for both generation and error tagging keeps the framework compact and avoids the substantial memory and computational overhead of introducing a second model. Weight sharing also ensures consistent linguistic priors and clinical terminology across stages. This design does not introduce self-loop bias because the tagging pass operates in a strictly frozen, inference-only mode and receives a different input structure—the ground-truth report paired with the model's prediction—forcing genuine semantic comparison rather than reproducing generative tendencies. Since gradients flow only through the generation pass, the referee cannot co-adapt with the generator. Finally, the consistent improvements observed across overall metrics and nearly all disease subcategories provide empirical evidence that this unified design remains effective and does not compromise semantic discrepancy detection

4. Prompt for Error Token Annotation

Instruction. Your task is to compare a *predicted report* with a *ground truth* medical report and identify specific tokens or phrases in the predicted report that are semantically incorrect or conflict with the ground truth. For this task:

1. **Incorrect tokens or phrases** are defined as parts of the predicted report that have a different meaning or contradict the ground truth.
2. Use a **matching pair of `<e>` and `</e>`** to wrap only the incorrect tokens or phrases in the predicted report.
3. Ensure that every error is marked precisely. Do not mark entire sentences—only the specific parts that are incorrect.
4. Preserve the structure of the predicted report. Do not split paragraphs or reformat the text.

Examples.

- **Example 1:**
  - Ground truth: *The lungs are clear and hyperinflated.*
  - Predicted report: *The lungs are clear and hyperinflation is present.*
  - Analysis: The predicted report matches the ground truth report.
  - Output: `The lungs are clear and hyperinflation is present.`
- **Example 2:**
  - Ground truth: *Findings: The lungs are low in volume. No focal airspace consolidation to suggest pneumonia.*
  - Predicted report: *Findings: The lungs are within normal volume. Focal consolidation is noted in the right lower lobe, concerning for pneumonia.*
  - Analysis: The predicted report incorrectly states *within normal volume* and adds *Focal consolidation*, which conflicts with the ground truth.
  - Output: `Findings: The lungs are <e>within normal volume</e>. Focal consolidation <e>is noted</e> in the right lower lobe, concerning for pneumonia.`
- **Example 3:**
  - Ground truth: *Impression: There is evidence of acute cardiopulmonary process.*
  - Predicted report: *Impression: No acute cardiopul-*

Table A1. Disease-wise comparison between our model and the baseline across multiple metrics. For each metric, scores before the "/" denote our model SAT-RRG, while scores after the "/" represent the baseline.

Disease	**Bleu_1**	**Bleu_2**	**Bleu_3**	**Bleu_4**	**ROUGE_L**	**METEOR**	**BERT_Score**	**RadCliQ**	**RadGraph**
Atelectasis	0.26 / 0.23	0.15 / 0.13	0.09 / 0.08	0.06 / 0.05	0.17 / 0.16	0.14 / 0.12	0.27 / 0.28	1.41 / 1.42	0.19 / 0.19
Cardiomegaly	0.24 / 0.21	0.14 / 0.11	0.08 / 0.07	0.05 / 0.04	0.16 / 0.15	0.14 / 0.12	0.26 / 0.27	1.43 / 1.44	0.18 / 0.18
Consolidation	0.24 / 0.19	0.13 / 0.10	0.08 / 0.06	0.05 / 0.04	0.15 / 0.14	0.14 / 0.11	0.25 / 0.24	1.55 / 1.55	0.20 / 0.17
Edema	0.25 / 0.20	0.13 / 0.10	0.08 / 0.06	0.05 / 0.03	0.16 / 0.14	0.15 / 0.13	0.27 / 0.25	1.39 / 1.45	0.20 / 0.17
Fracture	0.21 / 0.18	0.12 / 0.09	0.06 / 0.05	0.04 / 0.03	0.15 / 0.14	0.13 / 0.12	0.25 / 0.22	1.55 / 1.57	0.18 / 0.18
Pneumonia	0.23 / 0.15	0.12 / 0.07	0.06 / 0.03	0.04 / 0.01	0.14 / 0.13	0.12 / 0.10	0.22 / 0.20	1.57 / 1.65	0.18 / 0.15
Pneumothorax	0.23 / 0.15	0.12 / 0.07	0.06 / 0.03	0.04 / 0.01	0.16 / 0.12	0.12 / 0.10	0.24 / 0.21	1.54 / 1.65	0.16 / 0.15
LungLesion	0.19 / 0.12	0.10 / 0.06	0.05 / 0.03	0.03 / 0.01	0.13 / 0.10	0.10 / 0.08	0.20 / 0.17	1.54 / 1.61	0.15 / 0.13
NoFinding	0.27 / 0.19	0.17 / 0.10	0.12 / 0.08	0.10 / 0.05	0.22 / 0.21	0.24 / 0.22	0.42 / 0.42	1.71 / 1.70	0.34 / 0.31
PleuralEffusion	0.26 / 0.21	0.13 / 0.10	0.09 / 0.07	0.06 / 0.04	0.16 / 0.15	0.14 / 0.13	0.27 / 0.24	1.39 / 1.43	0.20 / 0.19
PleuralOther	0.26 / 0.22	0.14 / 0.11	0.09 / 0.07	0.06 / 0.04	0.17 / 0.14	0.14 / 0.12	0.24 / 0.22	1.66 / 1.67	0.17 / 0.16
SupportDevices	0.26 / 0.22	0.13 / 0.10	0.08 / 0.06	0.05 / 0.03	0.15 / 0.14	0.13 / 0.13	0.26 / 0.25	1.61 / 1.50	0.17 / 0.18
EnlargedCardior	0.22 / 0.22	0.11 / 0.13	0.06 / 0.08	0.04 / 0.05	0.15 / 0.16	0.13 / 0.14	0.24 / 0.26	1.61 / 1.50	0.17 / 0.18
AirspaceOpacity	0.22 / 0.17	0.11 / 0.09	0.07 / 0.05	0.04 / 0.03	0.15 / 0.13	0.12 / 0.11	0.22 / 0.21	1.59 / 1.63	0.17 / 0.16

monary process.
  - Analysis: The predicted report incorrectly negates the cardiopulmonary process described in the ground truth.
  - Output: `Impression: <e>No acute</e> cardiopulmonary process.`
- **Example 4:**
  - Ground truth: *The patient has a 12-cm calcified granuloma unchanged from the prior study.*
  - Predicted report: *The patient has a mass in the lower lung field.*
  - Analysis: The predicted report incorrectly describes a *mass in the lower lung field*, which conflicts with the *12-cm calcified granuloma* in the ground truth.
  - Output: `The patient has a <e>mass in the lower lung field</e>.`

Analyze the following reports and return the predicted report with incorrect tokens or phrases wrapped in matching pairs of `<e>` and `</e>`. Focus only on semantic differences, and ensure no extra modifications are made to the predicted report.

5. Error-Aware Training Objectives Analysis

CE supervises the ground-truth token, while our loss acts on the model's own belief. Standard cross-entropy (CE) loss supervises the log-probability of the *ground-truth* token $y^*_{b,t}$:

$$\mathcal{L}_{\text{CE}} = -\sum_{b,t} \log p_{b,t}(y^*_{b,t}), \tag{1}$$

where $p_{b,t}(v)$ is the model's predicted distribution over the vocabulary. CE cares only about whether the probability of the correct token increases; all incorrect tokens share the remaining mass $1 - p_{b,t}(y^*_{b,t})$ and are treated equally. Thus, CE has no way to know *which* wrong token the model currently prefers.

Our method instead operates on the model's *own predicted token*:

$$\hat{y}_{b,t} = \arg\max_{v} p_{b,t}(v), \qquad \log p^{\text{pred}}_{b,t} = \log p_{b,t}(\hat{y}_{b,t}), \tag{2}$$

allowing us to directly manipulate the model's belief in its chosen word.

For example, suppose the ground-truth phrase is *"no pleural effusion"*, but the model predicts *"consolidation is present, no pleural effusion"*. CE only encourages assigning higher probability to "no" and "effusion", but cannot reduce the model's mistaken belief that "present" is correct. Our loss, however, sees (i) which token was chosen (`consolidation`, `is`, `present`), and (ii) how confident the model is in these wrong choices.

How push–pull correction works: a concrete example. During training, the LLM referee highlights the conflicting span:

`<e>consolidation is present</e>,`

no pleural effusion.

Tokens inside this span are treated as *error tokens*, and the rest as *correct tokens*. ETAPL applies a *repulsive* gradient on erroneous predictions, reducing their probabilities, while CTAL applies an *attractive* gradient on correct predictions, strengthening them.

Table A2 illustrates how probabilities change in one update step:

Here, `consolidation` starts at 0.80 probability, meaning the model is strongly convinced it is correct—even though it contradicts the reference. ETAPL applies a positive gradient on $\log p^{\text{pred}}$, pushing the logit downward and

Table A2. Example of token-level push–pull behavior. ETAPL pushes down the confidence of erroneous predictions, while CTAL pulls up the confidence of correct ones.

token	init prob	type	updated prob
`consolidation`	0.80	error (penalize)	0.55
`is`	0.30	error (penalize)	0.15
`present`	0.61	error (penalize)	0.42
`no`	0.92	correct (reinforce)	0.96
`pleural`	0.79	correct (reinforce)	0.86
`effusion`	0.71	correct (reinforce)	0.83

reducing the probability to 0.55. Conversely, correct tokens such as `no` and `pleural` receive negative gradients on $\log p^{\text{pred}}$, pulling their logits upward and strengthening the model's confidence.

This prediction-driven update is fundamentally different from CE: the model is taught not only to increase the likelihood of the ground-truth token, but also to *explicitly unlearn* its own incorrect beliefs and *reinforce* its correct ones, forming the basis of our push–pull optimization.

Step-by-step loss computation on token probabilities. The push–pull behavior in Table A2 can be written explicitly in terms of token probabilities p^{pred}. For an error token (e.g., `consolidation`) with initial probability $p^{\text{pred}}_{\text{err}} = 0.80$, ETAPL contributes

$$\ell^{\text{ETAPL}}_{\text{err}} = w^{\text{err}} \log p^{\text{pred}}_{\text{err}}, \tag{3}$$

where $w^{\text{err}} > 0$ is the focal–entropy weight (detached from gradients). Since $\log p^{\text{pred}}_{\text{err}} < 0$, we have $\ell^{\text{ETAPL}}_{\text{err}} < 0$, and the gradient with respect to $p^{\text{pred}}_{\text{err}}$ is

$$\frac{\partial\, \ell^{\text{ETAPL}}_{\text{err}}}{\partial\, p^{\text{pred}}_{\text{err}}} = \frac{w^{\text{err}}}{p^{\text{pred}}_{\text{err}}} > 0. \tag{4}$$

Thus, during gradient descent, decreasing $p^{\text{pred}}_{\text{err}}$ (e.g., from 0.80 to 0.55) lowers $\ell^{\text{ETAPL}}_{\text{err}}$ and therefore decreases the overall loss. Intuitively, the model is rewarded (lower loss) when it *reduces* its confidence in an erroneous prediction.

For a correct token (e.g., `no`) with probability $p^{\text{pred}}_{\text{cor}} = 0.92$, CTAL contributes

$$\ell^{\text{CTAL}}_{\text{cor}} = -\, w^{\text{cor}} \log p^{\text{pred}}_{\text{cor}}, \tag{5}$$

with gradient

$$\frac{\partial\, \ell^{\text{CTAL}}_{\text{cor}}}{\partial\, p^{\text{pred}}_{\text{cor}}} = -\, \frac{w^{\text{cor}}}{p^{\text{pred}}_{\text{cor}}} < 0. \tag{6}$$

Here, increasing $p^{\text{pred}}_{\text{cor}}$ (e.g., from 0.92 to 0.96) makes $\log p^{\text{pred}}_{\text{cor}}$ less negative and thus decreases $\ell^{\text{CTAL}}_{\text{cor}}$, again reducing the loss. In other words, the model is rewarded when it *raises* its confidence in correct tokens.

After weighting and normalization,

$$\mathcal{L}^{\text{ETAPL}} = \frac{\sum m^{\text{err}} c^{\text{err}} \ell^{\text{ETAPL}}}{E}, \qquad \mathcal{L}^{\text{CTAL}} = \frac{\sum m^{\text{cor}} c^{\text{cor}} \ell^{\text{CTAL}}}{C}, \tag{7}$$

and combining them into

$$\mathcal{L}_{\text{EA-FE}} = \lambda_{\text{err}}\, \mathcal{L}^{\text{ETAPL}} + \mathcal{L}^{\text{CTAL}}, \tag{8}$$

the optimizer will consistently: (1) push down p^{pred} for tokens in error masks (to reduce $\mathcal{L}^{\text{ETAPL}}$), and (2) pull up p^{pred} for tokens in correct masks (to reduce $\mathcal{L}^{\text{CTAL}}$). Finally, the total objective

$$\mathcal{L}_{\text{total}} = \alpha\, \mathcal{L}_{\text{CE}} + (1-\alpha)\, \mathcal{L}_{\text{EA-FE}} \tag{9}$$

balances this prediction-driven push–pull mechanism with standard likelihood learning: CE increases the probability of ground-truth tokens, while EA–FE explicitly decreases p^{pred} for erroneous tokens and increases p^{pred} for correct ones.

Numerical illustration of how changing p affects the loss. To clearly demonstrate how ETAPL and CTAL respond to the probability changes in Table A2, we compute all loss terms explicitly. For simplicity, entropy and focal weights are temporarily set to 1 and normalization is omitted so that only the effect of changing p^{pred} is examined.
Combined EA–FE loss. Setting $\lambda_{\text{err}} = 1$ for illustration:

$$\mathcal{L}_{\text{EA-FE}} = \mathcal{L}^{\text{ETAPL}} + \mathcal{L}^{\text{CTAL}}.$$

Initial:

$$\mathcal{L}_{\text{EA-FE,init}} \approx -0.640 + 0.221 = -0.420.$$

Updated:

$$\mathcal{L}_{\text{EA-FE,new}} \approx -1.121 + 0.126 = -0.995.$$

This numerical example shows that *decreasing the probability of erroneous tokens and increasing the probability of correct tokens jointly reduce ETAPL and CTAL, thus lowering the EA–FE loss*. This is precisely the intended push–pull behavior that drives semantic refinement during training.

Initial Probabilities

ETAPL (error tokens)

$$
\begin{aligned}
p_{\text{err}} &= \{0.80, 0.30, 0.61\},\\
\log p_{\text{err}} &\approx \{-0.223, -1.204, -0.494\},\\
\mathcal{L}_{\text{init}}^{\text{ETAPL}} &= \tfrac{1}{3}\sum \log p_{\text{err}} \approx -0.640.
\end{aligned}
$$

CTAL (correct tokens)

$$
\begin{aligned}
p_{\text{cor}} &= \{0.92, 0.79, 0.71\},\\
\log p_{\text{cor}} &\approx \{-0.083, -0.236, -0.342\},\\
\mathcal{L}_{\text{init}}^{\text{CTAL}} &= -\tfrac{1}{3}\sum \log p_{\text{cor}} \approx 0.221.
\end{aligned}
$$

Updated Probabilities

ETAPL (error tokens)

$$
\begin{aligned}
p_{\text{err,new}} &= \{0.55, 0.15, 0.42\},\\
\log p_{\text{err,new}} &\approx \{-0.598, -1.897, -0.868\},\\
\mathcal{L}_{\text{new}}^{\text{ETAPL}} &= \tfrac{1}{3}\sum \log p_{\text{err,new}} \approx -1.121.
\end{aligned}
$$

CTAL (correct tokens)

$$
\begin{aligned}
p_{\text{cor,new}} &= \{0.96, 0.86, 0.83\},\\
\log p_{\text{cor,new}} &\approx \{-0.041, -0.151, -0.186\},\\
\mathcal{L}_{\text{new}}^{\text{CTAL}} &= -\tfrac{1}{3}\sum \log p_{\text{cor,new}} \approx 0.126.
\end{aligned}
$$

Table A3. Example computation of ETAPL and CTAL before and after the push–pull update.