Thanks for the constructive comments. Let us categorize them into groups and answer due to the page limit.

Reasoning behind ANTLR: We would like to emphasize that our main contribution is to discover the complementary nature of two main approaches for SNN training and provide a justification to combine them, not just randomly adding two different gradients. A spike-train of the neuron i can be represented in two different ways: $\Phi_i = \{S_i[t] \text{ for } t = 0, 1, \cdots, T-1\}$(*activation-based representation*) and $\Phi_i = \{\hat{t}_{i,k} \text{ for } k = 0, 1, \cdots, N-1\}$ (*timing-based representation*). One of the main arguments of our paper (in Section 3.1) is that the change in the state of neuron i can be described as $\Delta\Phi_i = \Delta\Phi_{i,\text{gen}} + \Delta\Phi_{i,\text{remov}} + \Delta\Phi_{i,\text{shift}}$, where $\Delta\Phi_i = \{\Delta S_i[t] \text{ for } t = 0, 1, \cdots, T-1\}$, $\Delta\Phi_{i,\text{gen}} = \{\Delta S_i[t] = +1 \text{ for } t \in \mathcal{T}_{\text{gen}}\}$, $\Delta\Phi_{i,\text{remov}} = \{\Delta S_i[t] = -1 \text{ for } t \in \mathcal{T}_{\text{remov}}\}$, and $\Delta\Phi_{i,\text{shift}} = \{\Delta S_i[t_{\text{before}}] = -1, \Delta S_i[t_{\text{after}}] = +1 \text{ for } \{t_{\text{before}}, t_{\text{after}}\} \in \mathcal{T}_{\text{shift}}\} = \{\Delta\hat{t}_{i,k} \text{ for } k = 0, 1, \cdots, N-1\}$. To update any parameter x for desired network output, $\frac{\partial\Phi_i}{\partial x}$ must be precisely computed so that it satisfies $\Delta x \cdot \frac{\partial\Phi_i}{\partial x} \approx \Delta\Phi_i$. Due to the limitation in considering the reset path for gradient computation, $\frac{\Delta\Phi_{i,\text{shift}}}{\Delta x}$ cannot be precisely estimated/predicted by the activation-based gradient $\frac{\partial\Phi_i}{\partial x}\big|_{\text{act}}$. In contrast, the timing-based gradient $\frac{\partial\Phi_i}{\partial x}\big|_{\text{tim}}$ cannot estimate $\Delta\Phi_{i,\text{gen}}$ and $\Delta\Phi_{i,\text{remov}}$. Those changes cannot even be described in the *timing-based representation*. The solution we proposed with ANTLR is to compensate the inaccuracy in gradients by adding two imperfect and complementary values as: $\frac{\partial\Phi_i}{\partial x}\big|_{\text{act}} \approx \frac{\Delta\Phi_{i,\text{gen}}}{\Delta x} + \frac{\Delta\Phi_{i,\text{remov}}}{\Delta x}$, $\frac{\partial\Phi_i}{\partial x}\big|_{\text{tim}} \approx \frac{\Delta\Phi_{i,\text{shift}}}{\Delta x}$, $\frac{\partial\Phi_i}{\partial x}\big|_{\text{ant}} = \lambda_{\text{act}}\frac{\partial\Phi_i}{\partial x}\big|_{\text{act}} + \lambda_{\text{tim}}\frac{\partial\Phi_i}{\partial x}\big|_{\text{tim}} \approx \lambda_{\text{act}}\frac{\Delta\Phi_{i,\text{gen}}}{\Delta x} + \lambda_{\text{act}}\frac{\Delta\Phi_{i,\text{remov}}}{\Delta x} + \lambda_{\text{tim}}\frac{\Delta\Phi_{i,\text{shift}}}{\Delta x}$

Layer-wise *vs* network-wise summation: As Reviewer #2 pointed out, one can run the activation-based method and the timing-based method separately and use the summation of those two gradients for parameter update. However, there are major differences between that approach and ANTLR. First of all, depending on the type of the loss function, one of learning methods may provide zero gradients for every parameter (Table 1). Then combining the two methods by network-wise summation can be meaningless. Moreover, if the activation- and timing-based gradients are not combined in each layer, errors in the gradients are accumulated through the back-propagation along multiple layers.

Coefficients λ: We introduced the coefficients $\lambda_{\text{act}}, \lambda_{\text{tim}}$ to balance the gradients from two methods because the scale of the activation-based gradient can be arbitrarily changed by the hyper-parameters of the surrogate derivative. Even though we simply used $\lambda_{\text{act}}, \lambda_{\text{tim}} = 1$ in this work for convenience, the optimal coefficients should further be studied.

Adding/removing spikes: Reviewer #2 mentioned that timing-based methods can also add/remove spikes when a single spike constraint is relaxed. Timing-based methods may unintentionally generate/remove spikes as a result of parameter update while they try to shift the spike timing to reduce the loss. However, these *unintended* generation/removal of spikes do not contribute to training, as the timing-based method cannot estimate them in gradient computation.

No-spike penalty: The no-spike penalty (Line 191-192) has been used in the timing-based methods because they cannot train parameters when a neuron does not emit any spike (dead neuron problem). It is implemented by encouraging *every neuron* to emit at least one spike. One of the main advantages of ANTLR is that it can solve the dead neuron problem more efficiently, using variant of the count loss $\{\min(\sum_\tau S_d[\tau], 1) - 1\}^2$ (d is the index of the desired class label) that penalizes only *output neurons* without no spike. With the help of the activation-based part, ANTLR can add/remove spikes in both hidden and output neurons while allowing some neurons not to emit any spikes. Reviewer #2 commented that addition of the count loss would decrease the firing activity but it actually *increases* the spiking activity because it still penalizes some neurons without spikes.

Comparison with related works: In the submission, we did not report the comparison with other works which did not report the sparsity of spikes since it is not fair to compare each method solely by accuracy results. The accuracy of SNNs highly depends on the number of spikes used. To support the argument, we compared previous results of fully-connected SNNs on N-MNIST classification tasks with our experimental results (Table R1). Even though related works did not report exact results of spike numbers, our experiments using similar settings imply that previous works with high accuracy were benefited from the large amount of spike usage.

Experimental Settings: As the reviewers mentioned, our experimental results used different settings for each method. We reported the best (in terms of accuracy and efficiency) results from each method after we tested every option available to each method. ANTLR can provide similar accuracy compared to the activation-based method when the same setting is used (Table R1). However, this situation is not desirable for efficient use of SNNs because it requires larger number of spikes.

Table R1: Comparison of fully-connected SNNs on N-MNIST

Method	Type*	Accuracy [%]	Loss**	# Target spikes	# Spikes/sample
Lee et al. [27]	S	98.66	C	not fixed	N/A
Jin et al. [26]	S	98.84±0.02	C	35 / 5	N/A
SLAYER [10]	A	98.89±0.06	C	60 / 10	N/A
STBP [11]	A	98.78	C	300 / 0	N/A
SRM-based***	A	97.73±0.14	C	10 / 0	436±17
SRM-based***	A	**98.30±0.06**	C	60 / 10	**6536±120**
ANTLR	A&T	97.73±0.09	C	10 / 0	415±14
ANTLR	A&T	**98.05±0.10**	C	60 / 10	**6638±130**
Timing***	T	94.10±0.51	L	-	2166±294
ANTLR	A&T	96.58±0.25	L	-	**111±8**

* A (activation-based), T (timing-based), and S (scalar-mediated, refer to Section 5), ** C (count loss) and L (latency loss), *** Our implementation of existing approaches

Experiments using larger datasets: We agree that our experimental results are only from relatively small datasets. We wanted to focus on fundamental limitations of existing learning methods in gradient computation in this work. Experiments and analysis of ANTLR on larger datasets remain as a future study.