We appreciate the valuable comments from the reviewers. We will answer reviewers’ questions from three aspects, i.e.,
the novelty of the paper, algorithm scalability, and model properties. Due to the page limit, we will also address
the other comments in the paper if accepted.

Novelty: In respond to Reviewer 5, this paper’s major novelty is developing a new STL-based learning framework to
effort multivariate RNN models to follow critical model properties, especially targeting the sequential regression
tasks. Our method creates a practical way to ensure the logic rules’ satisfaction in an end-to-end manner. It increases
the robustness of the RNN models. Our approach achieves promising results on real city datasets, i.e., significantly
increasing the satisfaction of model properties (by about four times) and prediction accuracy (by about 18.5%).

We have carefully compared our work with all the related papers pointed out by the reviewers. First, STL, as a powerful
specification language, has been broadly applied to the specification and verification for CPS applications, such as
robotics [1,2], smart cities, healthcare. Therefore, we also choose STL to express the model properties. STL has been
applied to both continuous and discrete signals. Due to the nature of RNN, the traces in this work are discrete with a
finite length. Using STL to specify CPS properties is not our novelty. However, we systematically identify six critical
types of model properties in CPS, which we believe is valuable for users to define model properties in their context
and utilize our work in practice. Second, DNF and many equivalent forms have been used in different contexts. Our
algorithm not only converts STL to DNF, also calculates the satisfaction range for each predicate and thus finds the best
trace closest to a given trace. Besides, we also create algorithms to generate satisfaction traces tailored to deep learning
processes efficiently. Third, introducing formal logic to support learning has been a hot topic and achieved promising
performance in recent years, including our work. Most of the current works focus on reinforcement learning [3,4] and
classification tasks [5,6], which have very different scopes than our paper. Their proposed methods do not apply to our
target problem. For example, paper [5] (already cited in our paper) combines first-order logic with neural networks
using a Teacher-Student network structure targeting NLP (classification) tasks. Paper [7] (a paper rejected by ICLR
2020) does apply to RNN models. It adds a term of constraint to the loss function, and tries to reach globally minimal
robustness over the input space. However, it is much more time-consuming and less robust (a soft constraint enforced
by optimization) comparing to our teacher-student network structure. Different from these papers, our work targets
multivariate RNN-based regression tasks, uses a more representative logic for RNN training, and achieves a stronger
satisfaction of the requirement (satisfaction guarantee with the teacher network at the testing time).

Scalability: In respond to Reviewer 1, the computation time of Algorithm 1 is relevant to the number of predicates in
the STL formula. However, we create algorithms to generate satisfaction traces tailored to deep learning processes
efficiently. The time could increase when there are more predicates, but Algorithm 1 only needs to be executed ONCE
in the pre-process (i.e., before the training phase). Therefore, it will not cause any significant delay in training and
testing phases, even for a large amount of data or long-term prediction. Besides, there are approaches to obtaining
a sub-optimal solution in a reasonable time that can be integrated to Algorithm 1 if needed. In our evaluation, the
pre-processing time for all cases (which have reasonable complexity STL formulas as the real-world applications) is
less than 10 seconds. We will also address it in the paper.

To briefly answer the other questions from Reviewer 1, (1) the reviewer is right about the teacher network; (2) The
return value is a non-negative real number. If a variable satisfies a constrain in a clause, the term will be evaluated to
0; Otherwise, it will return the minimal distance over all the items in the satisfaction of φt (not necessary to be 1). (3)
STLnet is general enough to be applied to transformer-based sequence models. Choosing RNN and its variants is to
show the generalizability of our solution.

Model properties: In respond to Reviewer 4, Model properties broadly exist in real-world applications and systems. In
this paper, we identify several critical types (in Section 2 and evaluation) based on the model properties from existing
papers, systems, and applications in CPS domain. In practice, model properties can be (1) already known by the system
before prediction, e.g., constraints by the physical world, rules followed by the application domains, (2) defined by the
users based on their application (e.g., robotics), (3) mined from the models’ historical behaviors. (We also present a
similar discussion at the beginning of Section 2 in the paper.)

To briefly answer the other questions from Reviewer 4, (1) RMSE itself cannot capture the temporal correlations of the
sequence like eventually, existence, consecutive changes, etc. (2) [0,24] represents 24 hours in a day. Users can choose
to use () or [ ] based on if the beginning and ending hours are included. (3) Alg. 1 has a typo that the epsilon set should
be initialized with CalculateDNF(φt, t, sgn) where t is an element of T.

References: