

1 Thank you for your detailed comments. We will make all clarifications below in the next version.

2 **R1:** • We have violations after CI since we do early stopping – satisfying them till end can sometimes hurt overall  
3 performance since the model is not perfect. • We believe our novelty lies in the successful application of existing  
4 constraint optimization ideas, mostly from non-neural world, in learning state-of-the-art Deep Learning models. While  
5 general approach may be similar, details differ substantially. We note that our formulation in Sec 3.2 can handle any  
6 kind of complex constraints, not just linear. Further, though constraints in our experiments are linear in output variables,  
7 they are highly non-linear in the weight space. Learning the constraints automatically is a direction for future work.

8 **R2:** [originality] #1,2: There are important differences compared to Diligenti et al. (a) Unlike us, they do not work  
9 with full Lagrangian formulation. Their approach is simply modifying the loss, and can not handle hard constraints; also  
10 see the discussion in [sig&qual#2]. (b) Our formulation in Sec 3.2 is generic, and does not assume *any* specific form of  
11  $f(w)$ . It can work with arbitrary (including non-linear) constraints over the output variables, e.g. constraints over real  
12 valued variables. There, the use of Hinge function is critical to ensure no penalty is paid when the constraints are not  
13 violated, and also reduce the total number of dual variables. This is in contrast with Diligenti, which requires a specific  
14 form for the constraint function: i.e.  $0 \leq f(w) \leq 1$ . (c) While it is true that our conversion of Boolean constraints to  
15 soft-logic (Sec 3.3) is similar to Diligenti’s, we note that both are based on prior work (Brocheler et al. UAI-10).

16 **#4:** We said that Diligenti and Mehta are task specific since they only experiment on a single task. Will clarify these  
17 points. Lee et al. 2017 (and the AACL-19 version) deal only with the constrained inference, not learning. We will  
18 cite the two additional references (thanks!). **#3:** Jin et al. assume arbitrary non-convex non-concave form for a twice  
19 differentiable function over which min-max has to be performed, and hence, it is also applicable to our formulation. Our  
20 criteria for convergence (Algorithm 1) is when the change in both  $w$  and  $\Lambda$  parameters is less than some (respective)  
21 thresholds. We missed a slight detail: we increase  $l$  over successive  $\Lambda$  updates using an AP. Then, in the limit  $l \rightarrow \infty$   
22 and using a slight variation of Theorem 28 in Jin et al., we are guaranteed to converge to a local min-max point (Defn.  
23 14 in Jin et al.) of Lagrangian up to inclusion of some degenerate points. We will add this important theorem in the paper.  
24 In practice, our algorithm converges very fast – it takes less than 2x time of training without constraints. This is not too  
25 much cost to pay for doing hard constraint optimization and also getting significantly better results. • We see a large  
26 variation in  $\lambda$  values depending on the experiment, e.g., for NER they vary from 0.02 to 3.2.  $\lambda$  for a constraint depends  
27 on its degree of violation and not necessarily on its human perceived importance. Our algorithm is also significantly  
28 faster than doing a grid search over a single  $\lambda$  hyperparameter, which in addition to being restrictive, will also be much  
29 more expensive – each training run would be as costly as the no-constraint training (also see [sig&qual#2]).

30 **[clarity] #2:**  $f_k(w)$ s used in 3.2 for the case of Boolean constraints is same as  $f(w)$ s in 3.3.  $f(w) \geq 1$  ensures that  
31  $f(w)$  stays equal to 1 (since we wanted to write inequality constraint). **#3:** Yes, all our constraints are linguistically  
32 important. Will add examples in appendix. The first condition in line 242 is correct since the span should end (and not  
33 start) at index  $k$ . **#4:** see response to [experiments #1] **#5:** We reduced the number of NER tags to 9 since we wanted to  
34 work with ‘higher level’ NER tags (see line 262). **#6:**  $\lambda_k$  is positive since each  $h_k(w) \geq 0$  by construction, hence, the  
35 equality  $h_k(w) = 0$  is equivalently the inequality  $h_k(w) \leq 0$ .

36 **[sig&qual] #1.1:** The goal of our expts was different from Mehta’s. We were interested in examining whether the  
37 model can learn effectively in presence of constraints. So we performed our expts *without* additional Viterbi decoding  
38 at the end. Mehta used Viterbi in all expts – this explains difference in baselines. Nevertheless, based on your feedback,  
39 we ran our expts with Viterbi decoding and could replicate their baseline (67.28%). We obtained 69.11 using CL  
40 (supervised), which is 1.09 pt higher than their reported 68.02 for CL. Interestingly, our supervised performance is  
41 0.25 pt higher than their semi-supervised learning (68.86)! Further, our CL model violates 14.36% of the syntactic  
42 constraints as opposed to 20.49% and 19.38% for their constrained supervised and semi-supervised models, respectively.  
43 We will report comparison for 10% data setting in final version. **#1.2:** The reason for higher performance than Murty et  
44 al on 5% data is already explained in lines 323-325. Our B+H number for 100% data is lower than theirs due to a bug in  
45 their code. Fixing this bug resulted in somewhat lower performance than reported in their paper. Nevertheless, even if  
46 we take the number reported by them (75), our CL achieves 83.5 which is about 8 pts higher. **#1.3:** SCL performs 5 pts  
47 higher than B+H+T in Murty et al. Will add this comparison. **#2:** We performed experiments using a fixed  $\lambda$  (decided  
48 using grid search) in Sec 5.3 for 5% data. It could only achieve a performance of 72.3 (vs. 78.4 for CL). This clearly  
49 demonstrates that fixed  $\lambda$  may not be enough for performance, in addition to requiring expensive grid search.

50 **[experiments] #1.2:** There was a typo in Table 1 (supplement) - last entry of SCL column for F1-score should be 75.66  
51 (instead of 74.88). This makes it consistent with Fig 1(a) in the main paper and also answers the concern about SCL  
52 doing worse than CL (which is not the case). The performance difference of various algorithms at 26k and 37k in  
53 Figure 1(a) is not stat. significant. **#3:** We worked with dual decomposition as our CI for NER. We could not get Lee  
54 et al.’s code despite repeated requests. **#4:** Violations are less on smaller amount of data for SCL, since most labels are  
55 predicted as ‘others’ and it is easy to satisfy constraints.

56 **R3:** • see R2 response [originality]#3. • Strong duality doesn’t hold since original objective and constraints can be  
57 highly non-convex (in  $w$ ). • The performance difference in Fig 1(a) at 26k and 37k is not stat. significant; it is quite  
58 plausible for two algorithms to have same/similar performance, while one satisfies more constraints than the other. •  
59 Our method is not just for NLP – exploring other application domains is a direction for future work.