First of all, we would like to thank all the anonymous reviewers for the insightful comments and suggestions!

To Reviewer 1:

(1) Extracted rules. The number of first-order formulas is 29363, 64, 919, 19 on FB15k, WN18, FB15k-237, WN18RR respectively. We show the top-ranked rules and the weights learned by our approach on FB15k in the table below.

Composition		Symmetric	
[Person] ***Place of Birth*** [Location] ∧ [Location] ***In Country*** [Country] ⇒ [Person] ***Nationality*** [Country]	4.06	[Location] ***Adjoins*** [Location]	3.52
[Film] ***Film Actor*** [Actor] ∧ [Actor] ***Dubbing Language*** [Language] ⇒ [Film] ***Film Language*** [Language]	2.41	[Person] ***Spouse*** [Person]	3.46

Inverse		Subrelation	
[Person] ***Played*** [Instrument] ⇒ [Instrument] ***Instrumentalists*** [Person]	8.74	[Award] ***Winner*** [Person] ⇒ [Award] ***Nominee*** [Person]	3.04
[Location] ***In Time Zone*** [Time Zone] ⇒ [Time Zone] ***Time Zone of*** [Location]	5.17	[Person] ***Honored*** [Award] ⇒ [Person] ***Nominated*** [Award]	3.02

(2) Learning algorithms for MLNs. For learning MLNs, we tried L-BFGS as you suggested, and we got similar results as gradient descent. The reason is that learning MLNs with pseudolikelihood is a convex optimization problem, and therefore both gradient descent and L-BFGS are able to converge to the global optima, although L-BFGS is faster.

(3) Comparison with RotatE. The comparison with RotatE on the two larger datasets (i.e., FB15k and WN18) is shown in the right table, where we use RotatE as our KGE model. The improvement of pLogicNet over RotatE is not as significant as over TransE, as RotatE can already implicitly model most important rules on the current datasets. However, with MLNs, our framework is general to incorporate any logic rules, many of which could not be modeled by RotatE. This could be quite advantageous in some domains.

Algorithm	FB15k					WN18				
	MR	**MRR**	**H@1**	**H@3**	**H@10**	**MR**	**MRR**	**H@1**	**H@3**	**H@10**
RotatE	**40**	0.797	74.6	83.0	88.4	309	0.949	94.4	95.2	95.9
pLogicNet	42	**0.815**	**77.6**	**83.8**	**88.7**	**256**	**0.950**	**94.5**	**95.3**	**96.1**

To Reviewer 2:

(1) Related work. Thanks for pointing out these papers! The first two papers focus on inductive logic programming combined with neural networks, whereas our paper focuses on statistical relational reasoning (Markov Logic Networks in specific) combined with knowledge graph embedding for knowledge graph reasoning. The third paper is the most relevant one to ours, which uses Horn clauses to regularize the predictions made by knowledge graph embeddings. In contrast, our approach is built on top of the principled probabilistic framework, Markov Logic Networks, which can effectively handle the uncertainty of logic rules and triplets. We will discuss these papers in details in the revised draft.

(2) MLN settings. Thanks for the suggestions! Our approach is indeed quite general and can be applied to those MLN settings. We will add the results under the MLN settings in the future.

To Reviewer 3:

(1) Conceptual comparison. Thanks for the correction!! Both RUGE and NNE-AER are elegant methods for combining logic rules and knowledge graph embedding. As you corrected, both methods can also model the uncertainty of logic rules by using soft rules. However, our method has some key differences from RUGE and NNE-AER.

First, our approach models the uncertainty of logic rules under the principled framework of Markov Logic Networks. In all the three methods, each rule is associated with a weight to measure the uncertainty. In both RUGE and NNE-AER, the weights of rules are initialized by other algorithms (i.e., AMIE+) and then fixed during training. In contrast, our approach can dynamically learn and adjust the weights of logic rules according to the downstream reasoning tasks.

Second, both RUGE and NNE-AER are specifically designed for the task of knowledge graph reasoning. In contrast, as both Reviewer 1 and 2 pointed out, our approach is a general mechanism. Besides knowledge graph reasoning, our approach can also be applied to many other tasks in statistical relational reasoning. Therefore, our approach could be valuable to the statistical relational learning community, and no existing approaches have studied combining statistical relational learning methods (Markov Logic Networks in particular) with knowledge graph embedding methods.

(2) Apples-to-apples comparison. Thanks for pointing this out!! We have done some experiments under the same settings as used in RUGE and NNE-AER. Following both methods, we use ComplEx as the KGE model. To compare with RUGE, we only use Horn clauses of length at most 2, corresponding to our inverse and composition rules (see Table 1 of the RUGE paper). To compare with NNE-AER, we only use the approximate entailment, corresponding to our inverse and subrelation rules (see Table 1 of the NNE-AER paper). We present the results in the following table, where our approach consistently performs better. The reason is that our approach can dynamically infer the uncertainty of logic rules based on the downstream reasoning tasks. For the results of our approach on FB15k, one may notice that the results in the right table are higher than those in the left table. This is because in the right table, no composition rules are used. As the composition rules on the FB15k dataset are very noisy, incorporating them will decrease the results.

Algorithm	FB15k			
	MRR	**H@1**	**H@3**	**H@10**
RUGE	0.768	70.3	81.5	86.5
pLogicNet	**0.786**	**73.3**	**82.0**	**88.2**

Algorithm	FB15k				WN18			
	MRR	**H@1**	**H@3**	**H@10**	**MRR**	**H@1**	**H@3**	**H@10**
NNE-AER	0.803	76.1	83.1	87.4	0.943	94.0	94.5	94.8
pLogicNet	**0.827**	**79.2**	**84.6**	**89.3**	**0.950**	**94.3**	**95.6**	**96.3**