We would like to thank the reviewers for the insightful remarks and comments. We address several concerns of the reviewers as follows.

Main theoretical results and practical implications: R1 and R3. Our main theoretical result is that the Max-Product algorithm can be exactly parameterized by the FGNN using number of parameters that are polynomial w.r.t. number of rank-1 tensors required to represent the higher order potentials (Corollary 7 in the main text). In worst case, it may require exponentially many parameters for arbitrary higher order potentials. However, we believe that there will be multiple practical applications that have reasonablely sized representation. This will have to be verified empirically – we have demonstrated this for error correcting codes which showed improved performance under bursty channel, and for human motion prediction which showed state-of-the-art performance. Furthermore, in computer vision applications, commonly used handcrafted higher order potentials, *e.g.* Potts model or Robust Potts Model [DOI:10.1007/s11263-008-0202-0], can often be exactly or approximately represented using a moderate number of rank-1 tensors. FGNN is also advantageous when the potential functions are unknown and only the features corresponding to the factor are provided. In this case, we can use FGNN to learn the approximation of potential functions from data, where the size of the network can be decided based on the computational budget or amount of available data (to control over-fitting). As FGNN can exactly represent the Max-Product, it should be at least as powerful as the Max-Product algorithm. It can represent a set of algorithms including the Max-Product, so by learning from data it may potentially learn a better inference algorithm if Max-Product is not optimal. In our experiment, we show that FGNN outperforms the Max-Product algorithm.

Possible over-fitting for synthetic data: R1. In the synthetic data experiment, our goal is to train a better MAP inference algorithm, and thus the MAP assignment is used as the target for all neural network based algorithms. In the experiment, the number of all possible configurations is $2^{30} \approx 10^9$ for a length-30 binary chain structured MRF, and our training set only has 90000 items. In this case, it is unlikely that a neural network that simply over-fits the training set can have a good performance on the test set.

Testing on novel graph structures: R3 and R1. To address R3's concern that the algorithm is tested only on the same graph structure as seen in training, we conducted a new experiment to train the FGNN on fixed length-30 MRFs using the same protocol as Dataset3, and test the algorithm on 60000 random generated chain MRF whose length ranges from 15 to 45 (the potentials are generated using the same protocol as Dataset3). The result is in Table 1, which shows that the model trained on fixed size MRF can be generalized to MRF with different graph structures. This also further addresses the overfitting issue raised by R1.

Chain length	AD3	FGNN
(15, 25)	88.95%	94.31%
(25, 35)	88.18%	93.64%
(35, 45)	87.98%	91.50%

Table 1: Accuracy on dataset with different chain size.

Factor without trivial representation of fixed dimension: R3. In the experiment on human motion prediction, the potential functions are actually unknown and their representations are fully learned from data.

Variational inference: R1. The synthetic problems are MAP inference problems. As variation inference is designed for marginal inference, it cannot be directly used. Indirectly however, by using the zero-temperature technique, the variational inference can be applied to do MAP inference. By applying the zero-temperature technique, the objective function of the variational inference becomes equivalent to that of AD3 and MPLP. Particularly, the AD3 algorithm is guaranteed to converge to the optimal of that objective and thus it can be viewed as a variant of variational inference.

Improve the writing: R1. We will revise Section 3 to say that Corollary 7 is the main theoretical results of the paper and do careful proofreading. For Algorithm 1, Φ_{VF} and Θ_{VF} are parameters of the $\mathcal{Q}$ and $\mathcal{M}$ net in the Variable to Factor module, respectively. Φ_{FV} and Θ_{FV} are parameters of the $\mathcal{Q}$ and $\mathcal{M}$ net in the Factor to Variable module.

Aggregation function: R2. We choose the "max" aggregation function because theoretically "max" is invariant to the duplication of a element in the set, while "sum" or "average" is not. In real applications such as human motion prediction, different factor may have different size, but for better parallelization we may need to pad all factor to the same size. In this case, we may simply duplicate a node in factor to do this. We replaced the "max" aggregation with "sum" aggregation in the LDPC experiment and typical result is shown in Figure 1, where both algorithm achieve almost the same performance.

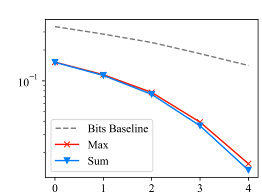

Figure 1: Comparison of "sum" and "max" aggregation.

Training time: R2. The training time is in Table 6 of the supplementary file.

Representativeness of the LDPC data: R2. In this experiment, as previous work [Kim et al., 2018, Zarkeshvari and Banihashemi, 2002] we do not have any assumption on the input signal, thus both the train and test set the input signal are uniformly sampled from $\{0, 1\}^{48}$. We can add error bars in the next version to indicate how well the results represent uniformly distributed inputs.

Relation with Yoon *et al.* and publishing code: R3. We will add discussion on relation with Yoon *et al.*, which is on pairwise potentials, as well as detailed experiment settings and the url to the source code in the next version.