Paper ID 6878: Response to Reviewers

Firstly, a big thankyou to all Reviewers for their time and constructive comments. We are able to address ALL raised queries and concerns (detailed below). Any points not directly addressed below will be corrected as a matter of course.

For Reviewers 2 & 3 on the presented data augmentation trick: As the prior for node embedding $\boldsymbol{\pi}_i^{(L)}$ is not a conjugate prior for the Ber-Poisson likelihood in SDREM, we can not obtain a Gibbs sampling update for $\boldsymbol{\pi}_i^{(L)}$. To circumvent this, we introduce an auxiliary latent counting vector $\boldsymbol{X}_i$. On one hand, each element X_{ik} in $\boldsymbol{X}_i$ is generated by π_{ik} only; on the other hand, the whole vector $\boldsymbol{X}_i$ can be regarded as a draw from a Multinomial distribution (with $\boldsymbol{\pi}_i$ as event probabilities). In this way, Gibbs sampling is then permitted for the Ber-Poisson likelihood function in this setting. We will clarify this novel construction in the revised version.

For Reviewers 2 & 3 on scalability: As SDREM uses Gibbs sampling, it does have some limitations in scaling to large networks. However the computational cost is only $\propto$ the number of positive links. We will clarify in the text.

Response to Reviewer 1

Reviewer 1 queries the lack of comparison with two specific and relevant methods. However, each requested comparison method (i.e. the GCN of reference [1] and the HLFM of reference [3], in Reviewer 1's comments) **has already been executed and compared** in the paper. Please see our responses for **2)–4)** below to explain this.

1) Referring to other positive embedding methods: As mentioned (Lines 208–209, 176–177) our positive-valued embedding method is related to the previous positive-valued embedding methods: Gamma Belief Networks [30] and Dirichlet Belief Networks [28]. We will improve the clarity of the text in the revised version.

2) and 3) Comparison with GCN: The GCN algorithm we discussed (Lines 127–134) and quantitatively compared against (Fig. 4) is actually Reviewer 1's requested GCN algorithm! We accidentally used the wrong reference in the text when citing the GCN algorithm. Thank you for your considered comment that allows us to spot and correct this.

4) Comparison with HLFM: Our reference [11] (HLFM) is reference [3] in Reviewer 1's comments. So we have already compared our SDREM with the HLFM in the 3rd paragraph in Section 4 (Lines 199-204) and in Fig. 4.

5) Test details: The testing relational data are not used when constructing the information propagation matrix (i.e. we set $\beta_{i'i}^{(l)} = 0$ if $R_{i'i}$ is testing data). We will clarify this in the revised version.

6) Same size of node embedding: For modelling simplicity, we used equal sizes for the node embeddings. However, by using merge-and-split (Beta distributed splitting ratio) operations on the elements of the Dirichlet distributions, the node embeddings can be of different sizes while still permitting Gibbs sampling on all variables. We will clarify this.

7) Why does R_{ij} not follow Bernoulli distribution? Actually, R_{ij} DOES follow the Bernoulli distribution when we integrate out the latent integers Z (see Lines 136-137). We will improve the text on this point.

Response to Reviewer 2

Analysis for intuitive understanding: Thankyou for the suggestion. We will provide some interpretable visualisations on some embedding outcomes in the revised version and comment that this is an advantage of the Bayesian model.

Reporting AUC-PR (Precision-Recall) values: This is a fair point (thank you). We have now calculated AUC-PR values for our analyses, and the results are consistent with our previous conclusions. We will update our results and discussion to incorporate this additional qualitative assessment.

Response to Reviewer 3

Comparisons with graph VAE methods (VGAE): As Mehta et al (2019) was available on arXiv ONLY 9 days before the NeurIPS submission deadline, we missed it when submitting SDREM. VGAE has a larger computational complexity ($\mathcal{O}(N^2)$). It uses parameterized functions to construct the deep network architecture and the probabilistic nature occurs in the output layer as Gaussian random variables only. In contrast SDREM constructs fully probabilistic deep architectures (with Dirichlet random variables at each layer). We will highlight these differences in the revision.

Evaluating SDREM on "cold-start" problems: Thankyou for the idea. We ran a quick experiment (following the recommended settings and with train:test = 9:1). The average AUC values are: Citeer (0.653), Cora (0.667), PPI (0.837), Pubmed (0.761), showing the effectiveness of our SDREM in using feature information. We will update the paper.

Diagnosing the convergence of MCMC algorithms: Actually 2000 iterations were adequate here (mixing was good). We used visual trace plots and standard convergence tests (on AUC & negative log-likelihood) to assess convergence.

Open source code: Yes, we will of course open source the code once the paper is published.