To Reviewer #2 and #3: Thanks for your positive comments. We will involve your suggestions in the revised version.

To Reviewer #1: Thank you for the comments. You have some misunderstandings on this paper.

C1: It is similar with (conditional) graph generation. R1: Our basic idea is different from the existing graph generative models. The latter ones aim to model the distribution of the observed data, while our method (GSNN) treats the classification function as a stochastic one and attempts to model the distribution of the stochastic classification function, which is a new idea for semi-supervised learning on graph data.

C2: It is similar to RGCN. R2: Our model is different from RGCN. RGCN models the node representations as Gaussian distributions, while GSNN models the uncertainty of the classification function by introducing the random latent variable $z$. Specifically, for GSNN, concatenating the sampled $z$ and the node feature $x$ is an implementation way to introduce randomness into the classification function, whose objective is not to model the node feature. Note that a sampled instance of $z$ could instantiate a classification function, which acts on all nodes in the graph. In other words, feature vectors for all nodes share the same instance of $z$.

C3: Why sample from prior instead of the posterior? R3: We agree that sampling from the posterior is a usual practice. Actually, our model is also designed following this practice as shown in Fig. 1 of the main paper. In the training phase, based on the reparameterization trick, we could sample $z$ from the posterior and optimize the model parameters. However, in the testing phase, to infer the missing labels, we still need to sample multiple instances of $Y_L$ from $q_{\phi}(Y_L | A, X, Y_L)$ before sampling $z$ from the posterior $q_{z\phi}$, which involves $q_{net1}$, $q_{net2}$ and $p_{net}$, making it inflexible. We notice that the second item of the ELBO objective function in Eq. (6) is the opposite of the KL-divergence between the posterior $q_{z\phi}$ ($z | A, X, Y$) and the prior $p(z)$, which can be seen as a regularizer to encourage the posterior to be close to the prior. Under this observation, we here adopt a simpler method, directly sampling $z$ from the prior $p(z)$, to construct multiple classification functions. This method only involves $p_{net}$ and is more flexible. The test results tell us these two kinds of methods are comparable.

C4: Concern about the mutual information between $z$ and the node. R4: The meaning of the introduced random latent variable $z$ is different from that in VAE or CVAE. As shown in Eq. (2) and lines 101 to 106 of the main paper, $z$ is not to distinguish different types of nodes, but to introduce randomness into the classification function $g_z$ (i.e., $p_{net}$). $z$ can be viewed as a parameter of $p_{net}$. Given a sampled $z$ from $z$, $p_{net}$ will specify an instance of classification function to classify all nodes in the graph. In other words, an instance of $z$ corresponds to a classification function, not a node in the graph. It is not the case of VAE on MNIST.

C5: The comparison is not fair since GSNN needs to sample $z$ for $L = 40$ times and it is 40 times slower than GNNs. R5: Since our proposed model is a Bayesian one, it is a standard practice to sample multiple samples of $z$ for estimating the proposed GSNN model. As to the efficiency, during the training phase, the number of the sampled instance of $z$ is set to 1 as mentioned in lines 236-237, which would not increase the time complexity compared with vanilla GNNs. While in the inference phase, as shown in Eq. (11), we need sample $L$ instances of $z$ to achieve the Monte Carlo estimation. However, the inference only requires $L$ feedforwards of $p_{net}$, which runs very fast. We plot Fig. A to show the change of the performance w.r.t. $L$. We can observe that the average classification accuracy keeps high and stable when $L > 10$ for all three datasets.

C6: Why the model can alleviate structure attacks by noise injection in feature? R6: (1) The performance gain benefits from modeling the randomness of the classification function, rather than the node feature. Please refer to R2. (2) We agree that modeling the graph structure is another potential way to deal with adversarial attacks, which, however, is not the main target of this paper. Besides, we included BGCN and G$^2$NN, which model the distribution of the graph structure, as our baselines. The experimental results show that the proposed GSNN achieves better performance.

C7: Why not use $Y_L$ as input for $q_{net1}$. R7: As shown in lines 148-149 and Fig. 1 in the main paper, $Y_L$ is used as the supervision information for training $q_{net1}$ in the Eq. (10), which serves as the input of $g_{net1}$ indirectly.

C8: How to select labeled nodes in the label-scarce scenario. R8: We select a certain percentage of nodes completely randomly. We will include it in our revised version.

C9: Suggest PPNP as a baseline. R9: Thank you for your suggestion. We will include PPNP as a baseline.

C10: Why performance is higher in the label-scarce and adversarial attack settings than the original graphs? R10: The dataset partition method under the standard experimental scenario is different from these under the label-scarce and adversarial attack scenarios. For example, on PubMed, the total number of labeled nodes in the case of 0.5% under the label-scarce scenario is still more than that under the standard experimental scenario.