Analysis of Node2vec

Mohan Bhambhani
Department of Computer Science
Purdue University
mbhambha@purdue.edu

Jennifer Neville
Department of Computer Science
Purdue University
neville@purdue.edu

Abstract

In the paper we have analysed node2vec on various measures on various datasets. We start by doing a bias-variance decomposition of the mean squared error. Bias-variance decomposition has been used in machine learning since the very beginning to understand the fitting of the model. Then we compare the performances label classification task on nodes with different degree. This can help us know about how can we remove bias during the sampling. To get rid of the bias in various tasks on the networks and to compare the performance of node2vec directly on the network we also look at the model error instead of the error on the task.

1 Introduction

1.1 Recent work on embeddings in networks

- **Proximity information:** My recent techniques for node embeddings in networks have explored first order and second order proximity. The popular ones are DeepWalk, node2vec, LINE, SDNE. DeepWalk and node2vec have greater focus on second order proximity. LINE gets embeddings corresponding to the first and second order proximity both separately and concatenates them. In SDNE, the second-order proximity is used by the unsupervised component to capture the global network structure. While the first-order proximity is used as the supervised information in the supervised component to preserve the local network structure. Some techniques like GraRep also look at higher order proximity.
- **Attribute information:** There has been some work on getting combined embedding from node attributes and structure information using semi-supervised learning. But, there has been no work on using edge attributes.
- **Edge embeddings:** Moreover, there is no work on getting edge embedding too, except node2vec suggests some ways to get edge embeddings form node embedddings.
- **Dynamic networks:** Most recently, there has been some work on node embeddings in dynamic networks as most of the networks today are not static. DANE provides an offline method for a consensus embedding first and then leverages matrix perturbation theory to maintain the freshness of the end embedding results in an online manner.

1.2 Node2vec working

- Many random walks are performed to sample the network.
- These random walks are then passed to word2vec.
- On each random walk keeping each node in it as core node, k nodes on either sides of the node on the random walk are taken and their vectors are updated such that they become closer to the core node.

1.3 Parameters of node2vec:

- d: dimension of the embedding
- r: number of walks to be performed from each node
- l: length of each walk
- k: context size
- p: return parameter
- q: in-out parameter

1.4 Similarity to DeepWalk

Node2vec is very similar to Deepwalk with 2 additional parameters. The parameters were introduced in a very convincing way. But, the paper does not explain what properties of the network would have an effect on the best values of these parameters. In fact, in the paper the authors do a cross-validation on 10% of the network to get values for p and q from a predefined small set. There may be better values for p and q outside of this set. So, here to avoid all the hustle with p and q, the values of p and q are set as 1 which is same as DeepWalk.

2 word2vec in Natural Language vs Network

- Natural languages must follow a syntax. This makes words further away also important for context in natural languages. But, in networks this is not the case. There is no syntax and hence, a node is only related to a node very nearer.
- In natural languages, more frequently occurring words are removed so that the other words can connect better. There is no such preprocessing done here.
- Most sentences in English start with frequently occurring words.

3 Bias-variance decomposition

To look at the bias-variance decomposition of the error and how they vary with the change in the parameters, we deviate from measure as accuracy to mean squared error. Experiments were performed on BlogCatelog3 network, with node Label classification task. The task was multi-label, multi-class classification, i.e. each node may belong to multiple classes. So we took into consideration 2 formulations for MSE.

1. True probability mass is distributed evenly among the true classes. Then, for all true classes the square of the error is sumed.
2. True probability mass is distributed evenly among the true classes. Then, for all classes the square of the error is sumed.

We use measure 1 for MSE for all our experiments hereafter.

3.1 Bias calculation

Bias was calculated by taking mean squared error of the mean of the output probability distribution from Logistic Regression on the embeddings of these 10 models to that of the expected output probability distribution. This task was multi-class multi-label so for mean squared error the probability mass was equally distributed among the positive classes. Bias thus calculated is an approximate of the actual bias as we can not deduct the underling function.

3.2 Variance calculation

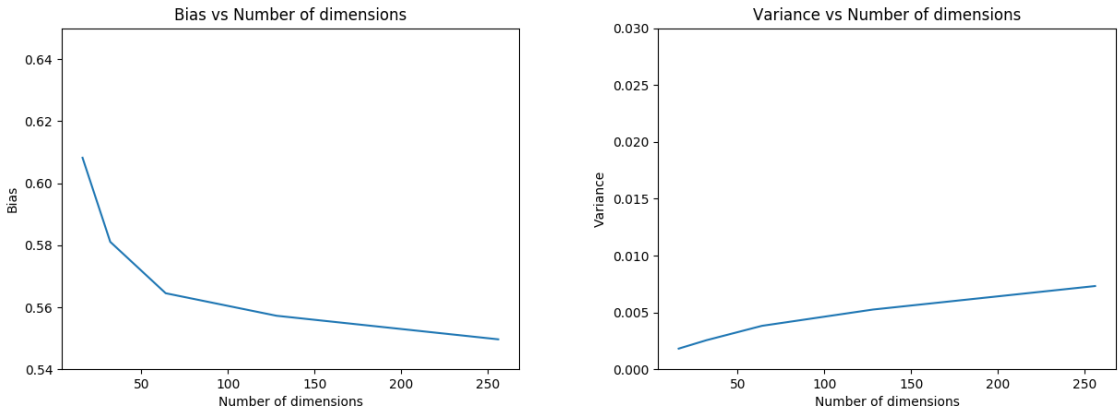
Variance was calculated by taking Mean squared error of the mean output probability distribution with each of the 10 output probability distributions. As the variance of due to negative classes was high, both bias and variance were calculated on the positive classes only.

3.3 Task

For node label classification, embeddings of 90% of the nodes was taken as training data and rest 10% was taken as test data. Logistic regression was used for classification.

3.4 Bias and variance vs dimensions

Bias - As expected. It should decrease as greater dimension will encode more information
Variance - As expected. Explanation similar to dimension reduction reduces variance.

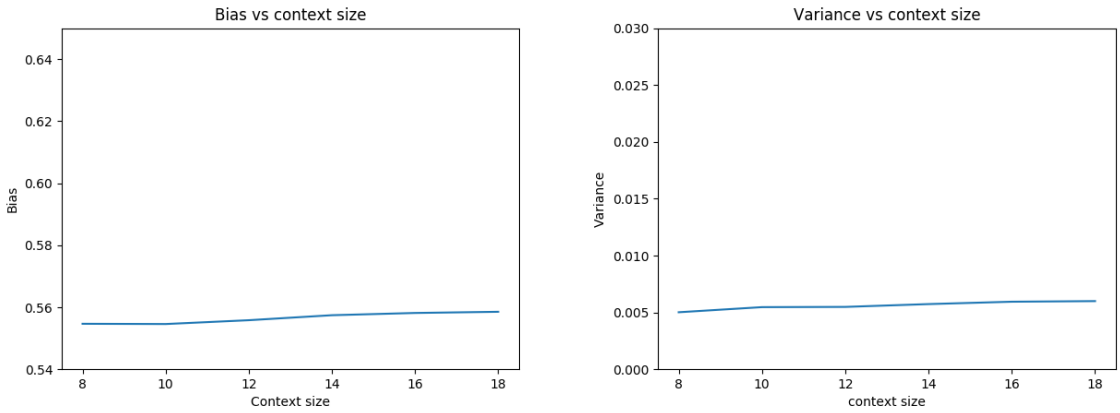


(a) Bias (b) Variance

Figure 1: Blog Catelog - Bias and variance vs dimension of the embedding

3.5 Bias and variance vs context size

Bias - Almost constant at smaller values but slowly rate of increase increases. Because we are adding bias by forcing it develop similarity with nodes that are far away.
Variance - Variance will also increase as the nodes that are far are different for different embeddings. But higher rate of increase was expected.

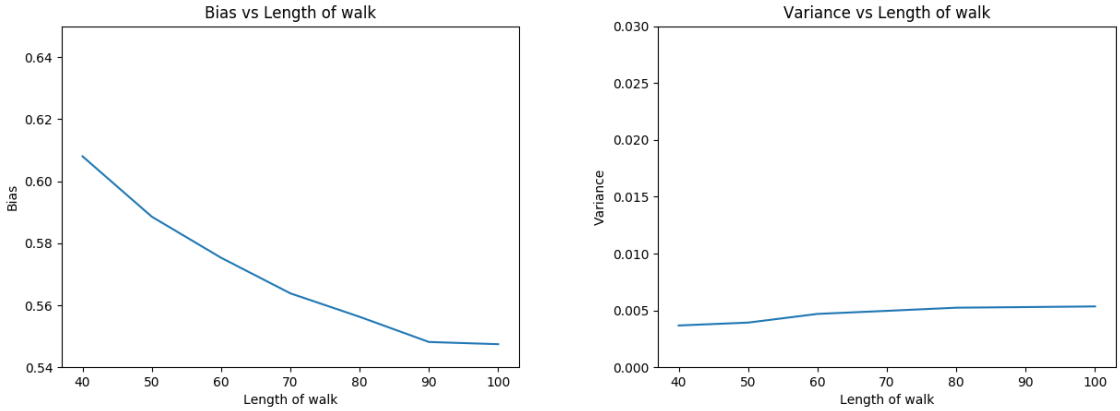


(a) Bias (b) Variance

Figure 2: Blog Catelog - Bias and variance vs context size

3.6 Bias and variance vs length of walk

Bias - Bias should either decrease or remain constant. Here it decreases because we have more data points from unseen parts of the original function.
Variance - Variance should decrease as we have more data. But here it increases. This may be happening because we have more data but with random walks of greater length there is more data but it will not be in the same neighbourhood.

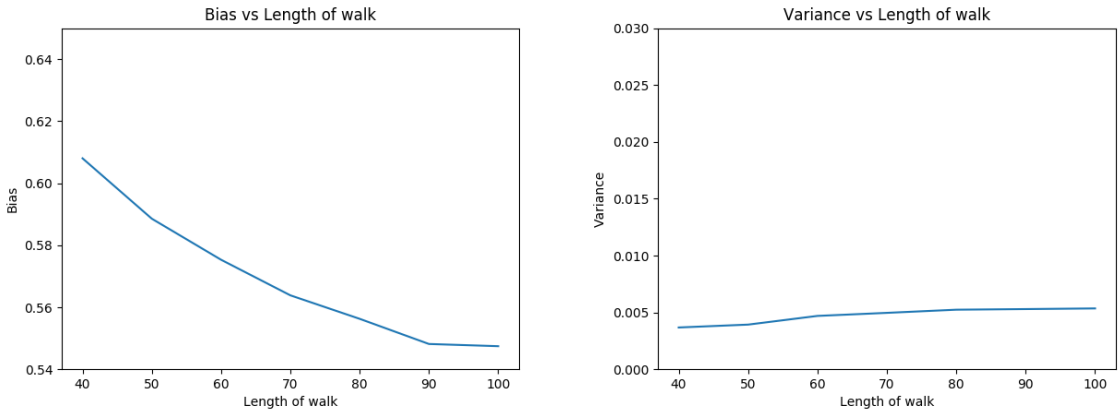


(a) Bias (b) Variance

Figure 3: Blog Catelog - Bias and variance vs length of walk

3.7 Bias and variance vs number of walks from each node

Trend was same as that of length of walk as sampled data increases in both.



(a) Bias (b) Variance

Figure 4: Blog Catelog - Bias and variance vs number of walks from each node

3.8 Other experiments

1. Similar experiments were performed on different networks generated from BlogCatelog network by randomly removing 10% edges. Trends of bias and variance thus obtained were same with a small shift up on the y-axis, as the density of the graph has now reduced.

2. Similar experiments were also conducted for different values of p and q. The trends were the same, with small shifts along the y-axis.

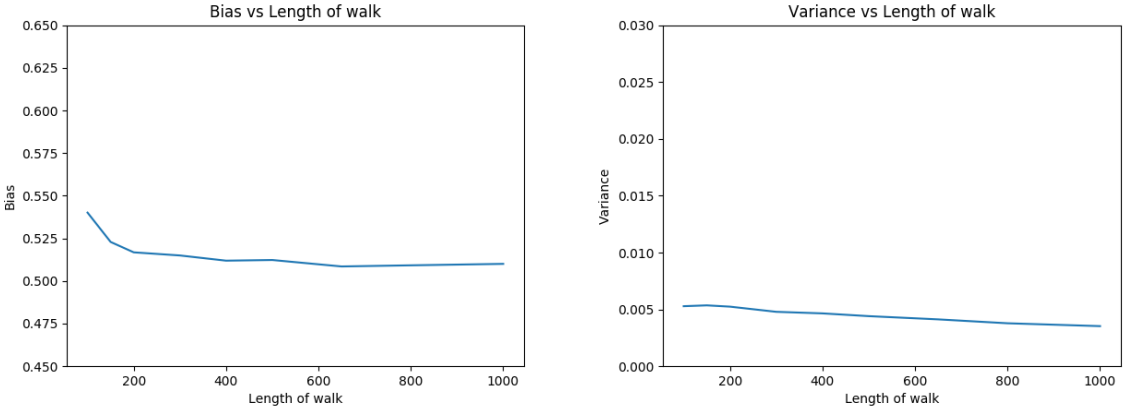
4 Node2vec vs Matrix factorisation(GraRep)

In GraRep, it has been shown that getting embeddings through node2vec and from matrix factorisation of the powers of the probability transition matrix same. First 4 powers of the probability transition matrix were taken. If SVD is applied on sum of these 4 matrices MSE of 0.587 was obtained on the embeddings. And if SVD is applied on all these matrices separately and (1/4)th features are taken from each and concatenated the MSE obtained was 0.688. MSE of 0.55 has been achieved using node2vec.

5 Methods to reduce bias

5.1 Longer random walks

Bias flattens very fast (at 200) and converges at 0.515. Also, there is a slight decrease in variance observed as expected.

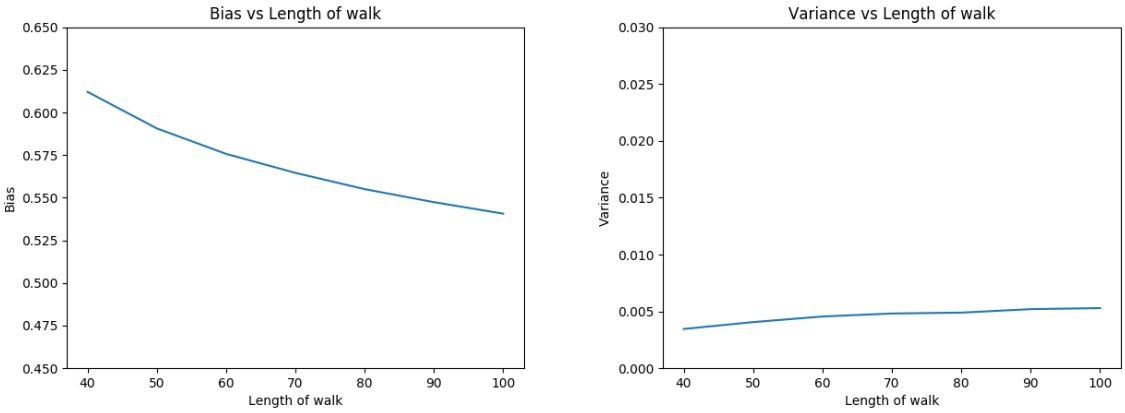


(a) Bias (b) Variance

Figure 5: Blog Catelog - Bias and variance vs length of walk

5.2 Remove first 100 nodes in Random walk

Bias and variance vs number of walks per node, where each walk was of length l obtained by deleting first 100 nodes of the walk of length l+100.

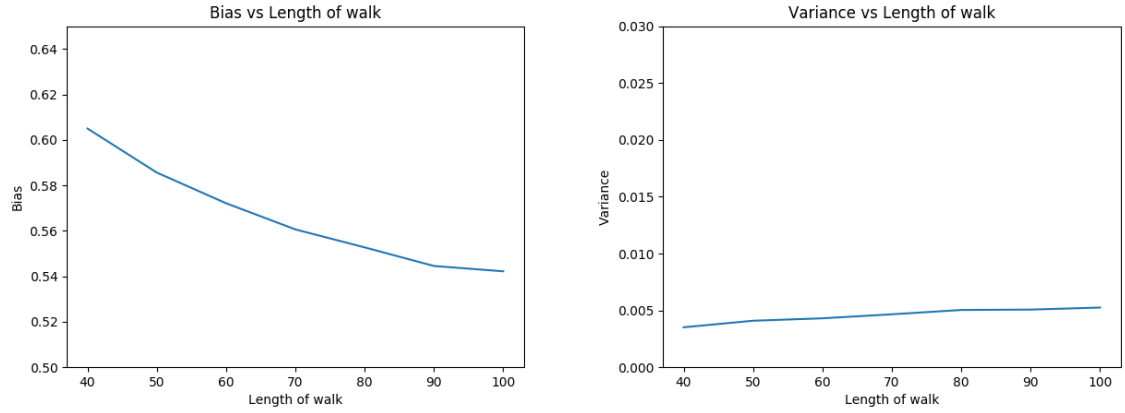


(a) Bias (b) Variance

Figure 6: Blog Catelog - Bias and variance vs length of walk

5.3 Number of walks form each node proportional to its degree

Out of 333,983 edges in BlogCatelog randomly 100,000 were sampled and random walks were started from each of these nodes of length 80. Results are almost similar to random node sampling.

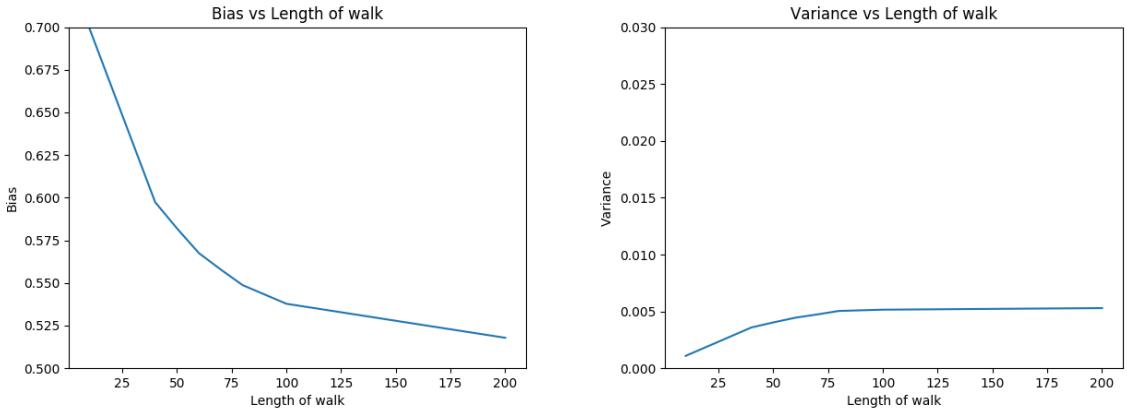


(a) Bias (b) Variance

Figure 7: Blog Catelog - Bias and variance vs length of walk

5.4 Sample 2 Random walks for a RW and concatenate them

For sampling a random walk of size l, 2 random walks were sampled of size l/2 starting at the node. First one was reversed and concatenated to the second one. By this we ensure enough examples are generated for each node. Here, we see steep decrease in bias as compared to before and an steep increase in variance.

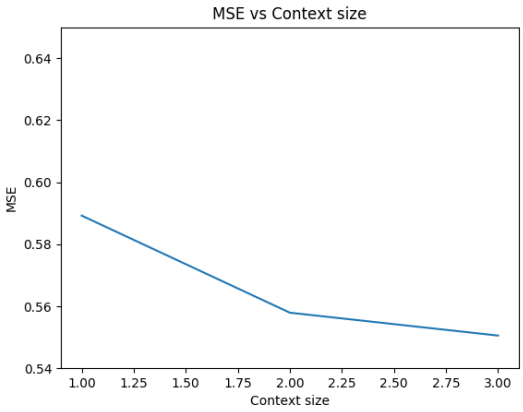


(a) Bias (b) Variance

Figure 8: Blog Catelog - Bias and variance vs length of walk

6 MSE vs small fixed k

In the gensim implementation, the k taken as input is actually the max context size. A random number is generated for each core node in each random walk between 1 and max context size and that is taken as the context size of the node. In the figure it has been shown how mean squared error varies for small k.

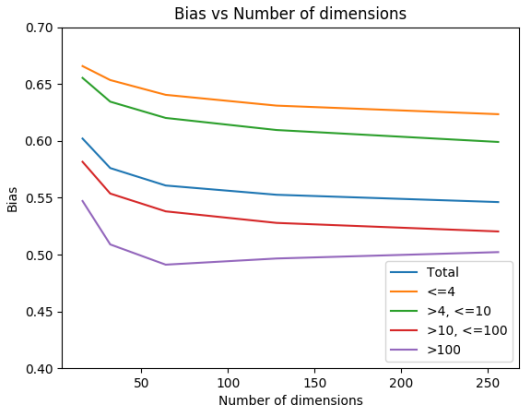


(a) Nodes with degree 1-3

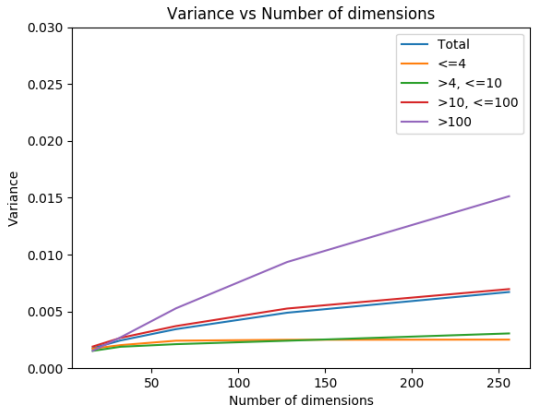
Figure 9: Mean squared error for small k

7 Degree-wise bias-variance decomposition

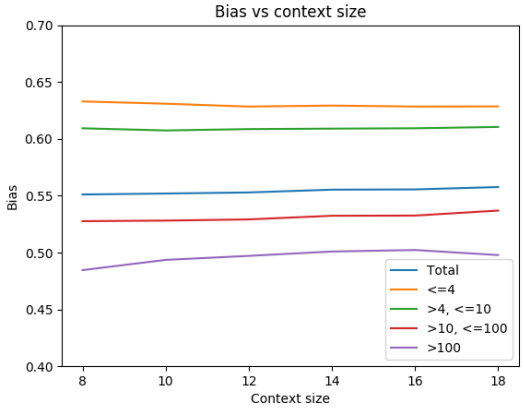
Nodes were categorized with respect to their degrees into 4 categories. First class included all nodes with less than or equal to 4 degree, second with nodes having degree between 5 and 10, third with having degree between 11 and 100 and fourth with nodes having degree greater than 100. Bias and variance were calculated of these categories separately.



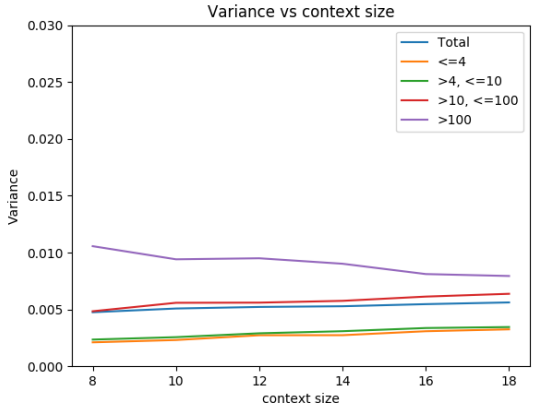
(a) Bias vs number of dimensions



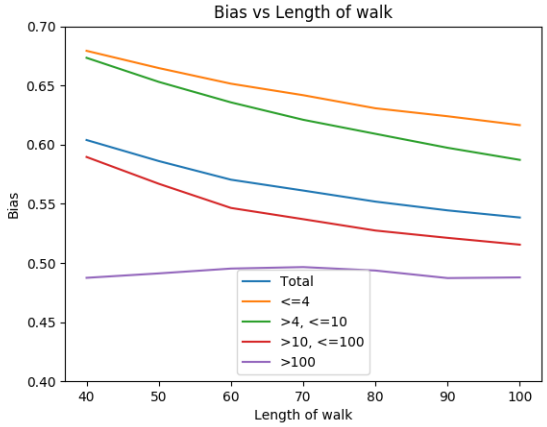
(b) Variance vs number of dimensions



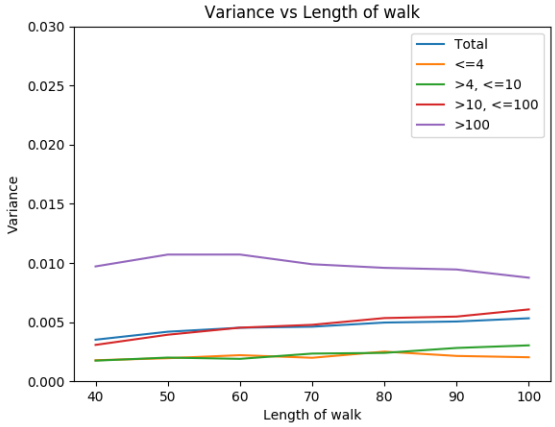
(c) Bias vs context size



(d) Variance vs context size



(e) Bias vs length of walks



(f) Variance vs length of walks

Figure 10: BlogCatelog - Bias and variance trends for different category of nodes

8 Model error comparison

8.1 Mean Squared error calculation

For calculation of the mean squared error from the model, random walks of size 80 were sampled starting from each node. Taking each node on the random walk as core, for each node in its context 5 negative samples were sampled. These were considered as the examples of the core node. Now, for given 2 nodes n_1 and n_2, the probability that n_2 lies in the context of n_1 is,

$$p(n_1, n_2) = \sigma(f(n_1) \cdot f(n_2)),$$

,where σ is the sigmoid function. $f(n)$ is the embedding of node n and $p(n_1, n_2)$ is the probability. Hence, mathematically the mean squared error for nodes in category 1 is,

$$MSE = \frac{\sum_{s \in V} \sum_{l=1}^{80} \mathbb{I}(deg(x_l^s) < 4)[\sum_{k=-5}^{5}[(1 - p(x_l^s, x_{l+k}^s))^2 + \sum_{D^-}(0 - p(x_l^s, x_{l+k}^s))^2]]}{\sum_{s \in V} \sum_{l=1}^{80} \mathbb{I}(deg(x_l^s) < 4)(1 + n)}$$

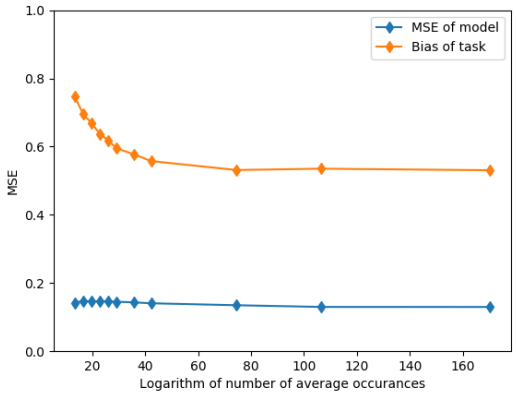
,where, n is the number of negative examples sampled for each node, x_l^s is the l^{th} node occuring in the test random walk starting from node s and $deg(n)$ gives the degree of the node n. Similarly, error was calculated for other categories as well.

8.2 Number of occurrences of nodes

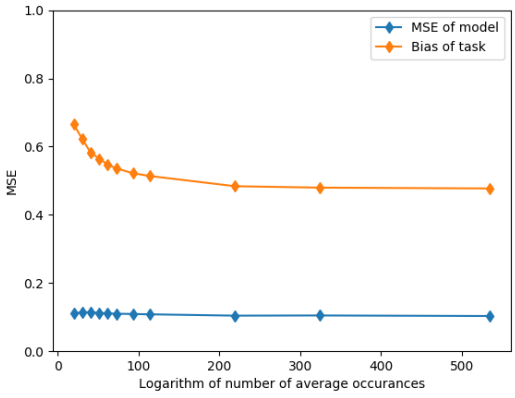
In the random walks sampled for training the neural network, the mean number of occurrences of nodes of each category was calculated. Mathematically it is shown as follows,

$$\frac{\sum_{i=1}^{r} \sum_{s \in V} \sum_{l=1}^{80} \mathbb{I}(deg(x_l^s) < 4)}{\sum_{s \in V} \mathbb{I}(deg(s) < 4)}$$

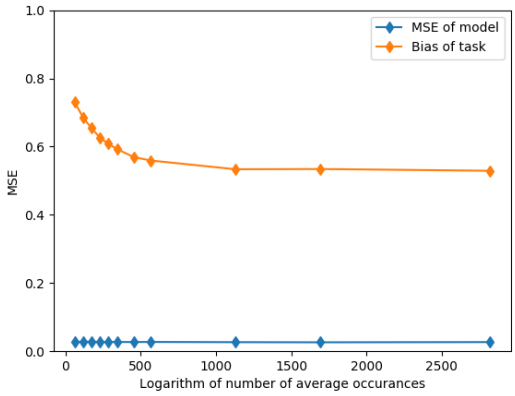
, where, r is the number of walks performed from each node. Throughout the experiments r was set as 10.



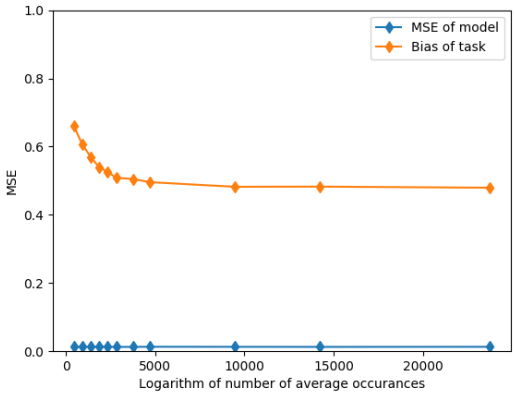
(a) Nodes with degree 1-3



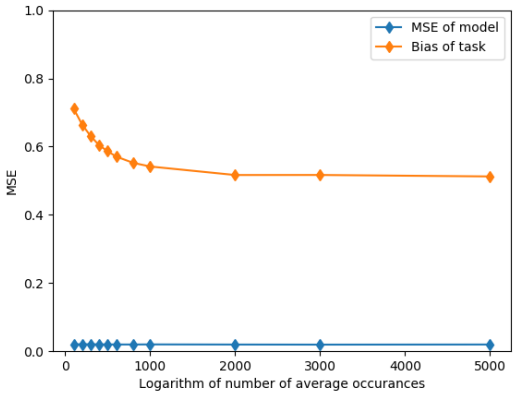
(b) Nodes with degree 4-10



(c) Nodes with degree 11-100



(d) Nodes with degree greater than 100

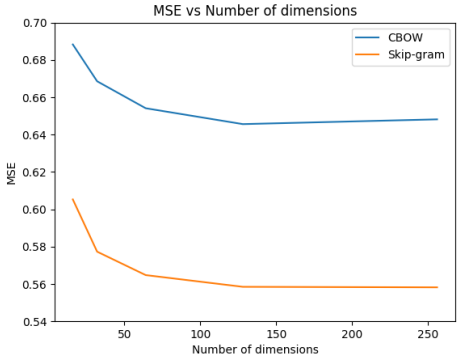


(e) All nodes

Figure 11: BlogCatelog - Bias of task and MSE of model vs logarithm of number of occurrences of nodes for different category of nodes

9 Skip-gram vs CBOW

Comparison of Skip-gram and CBOW on the random walks performed. CBOW does not work well here.



(a)

Figure 12: Skip-gram vs CBOW

10 Model error decomposed

Model error was decomposed into 2 parts, for the positive and negative examples. It was observed as the training progresses the error for positive examples increases while that for negative examples decreases. Initially the predicted probability of whether the context node lies in the context of the core node is around 0.5. As the training progresses the predicted probability decreases for both negative and positve examples.