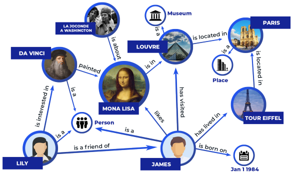
Deep Learning on Graphs with Differential Privacy

Sina Sajadmanesh
Idiap Research Institute
Swiss Federal Institute of Technology Lausanne (EPFL)

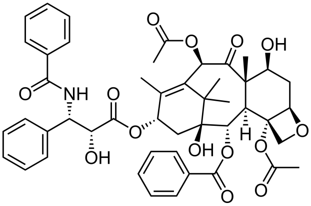
Joint work with Ali Shahin Shamsabadi, Aurélien Bellet, and Daniel Gatica-Perez

Imperial College London, March 2023

Graphs are Ubiquitous



Knowledge Graphs

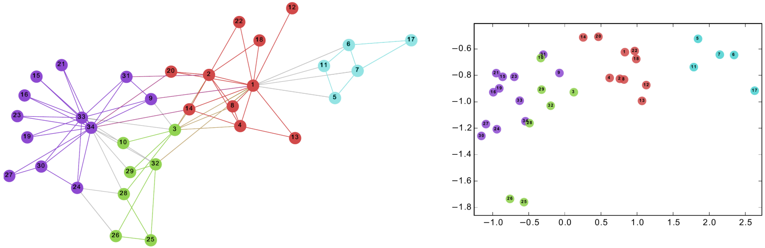


Molecules

Social Networks

Image source (from left to right): https://yashuseth.blog/2019/10/08/introduction-question-answering-knowledge-graphs-kgqa/, https://en.wikipedia.org/wiki/Terpenoid, https://yashuseth.blog/2019/10/08/introduction-question-answering-knowledge-graphs-kgqa/

- We need to learn representation of nodes in a low-dimensional space
  - Similar nodes in the graph should be mapped close to each other in the embedding space



- **Graph Neural Networks** (GNNs) are state-of-the-art representation learning algorithms for graphs.

Image source: [Perozzi et al., 2014]

Privacy Concerns

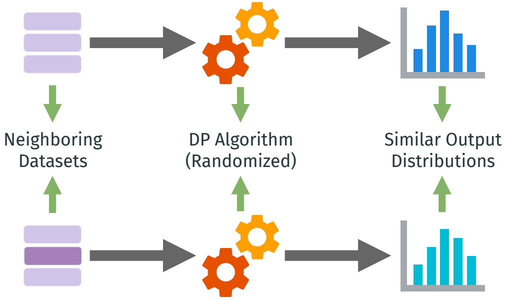
- Graph data could be privacy-sensitive
  - e.g., users' personal attributes, financial transactions, medical/biological networks, . . .
- Graph-based ML algorithms are vulnerable to privacy attacks
  - e.g., **link stealing attack** [He et al., 2021] or **membership inference attack** [Olatunji et al., 2021]

We need privacy-preserving machine learning algorithms for graph data!

Differential Privacy [Dwork et al., 2006]

Randomized algorithm A is (ϵ, δ)-differentially private if for all neighboring datasets $D \simeq D'$ and all sets of outputs S:

$$\Pr[A(D) \in S] \leq e^{\epsilon} \Pr[A(D') \in S] + \delta$$



Differential Privacy [Dwork et al., 2006]

Randomized algorithm A is (ϵ, δ)-differentially private if for all neighboring datasets $D \simeq D'$ and all sets of outputs S:

$$\Pr[A(D) \in S] \leq e^{\epsilon} \Pr[A(D') \in S] + \delta$$

- The neighboring relation captures what is protected
  - **Standard DP:** D and D' differ by at most **one record**
  - **Edge-level DP:** D and D' are graphs differing by at most **one edge**
  - **Node-level DP:** D and D' are graphs differing by at most **one node** (and all its adjacent edges)

DP-SGD Algorithm [Abadi et al., 2016]

input : Data $\{\vec{x}_1 \ldots, \vec{x}_N\}$, learning rate η, batch size B, epochs T, clipping threshold C, noise variance σ^2,

Initialize $\vec{\theta}_0$ randomly

for $t \in [T \cdot \frac{N}{B}]$ **do**

Sample a batch $\vec{B}_t$ by selecting each $\vec{x}_i$ independently with probability $\frac{B}{N}$

For each $\vec{x}_i \in \vec{B}_t$: $\vec{g}_t(\vec{x}_i) \leftarrow \nabla_{\vec{\theta}_t} L(\vec{\theta}_t, \vec{x}_i)$ // compute per-sample gradients

$\tilde{\vec{g}}_t(\vec{x}_i) \leftarrow \text{clip}(\vec{g}_t(\vec{x}_i), C)$ // clip gradients to max norm C

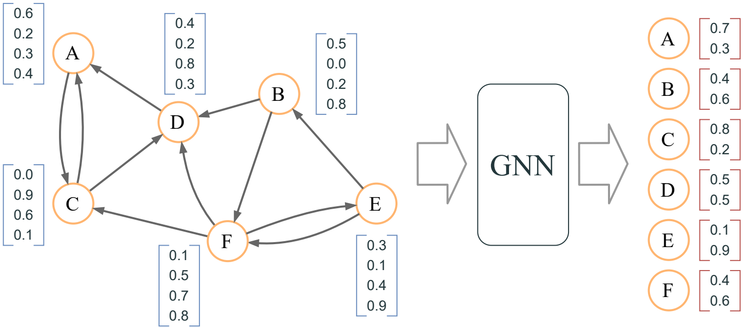
$\tilde{\vec{g}}_t \leftarrow \frac{1}{B} \left(\sum_{\vec{x}_i \in \vec{B}_t} \tilde{\vec{g}}_t(\vec{x}_i) + \mathcal{N}(0, \sigma^2 \vec{I}) \right)$ // add Gaussian noise with variance σ^2

$\vec{\theta}_{t+1} \leftarrow \vec{\theta}_t - \eta \tilde{\vec{g}}_t$ // SGD step

end

output: $\vec{\theta}_{\frac{TN}{B}}$

GNNs Learn Node Embeddings



A: Adjacency matrix

X: Input node features

Y: Predicted node labels

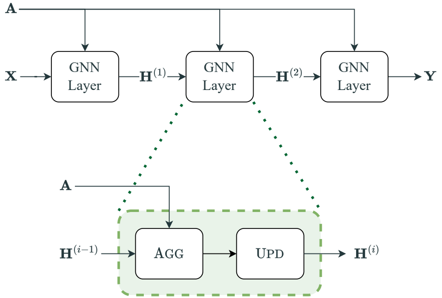
$H^{(i)}$: Hidden node representations of layer i

Agg: Aggregation function

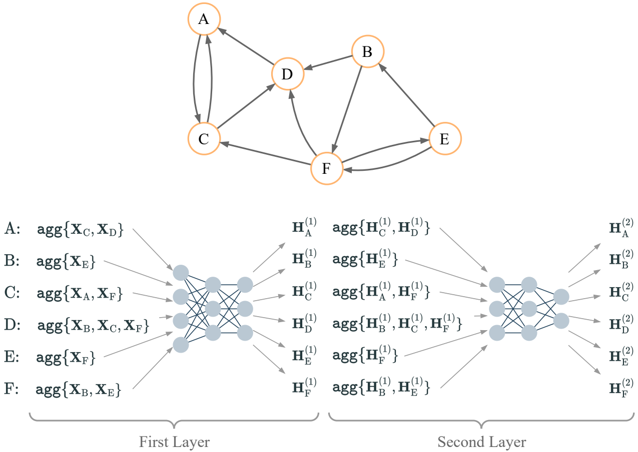
- e.g., summation: $\mathrm{AGG}(H, A) = A^T \cdot H$

Upd: Learnable update function

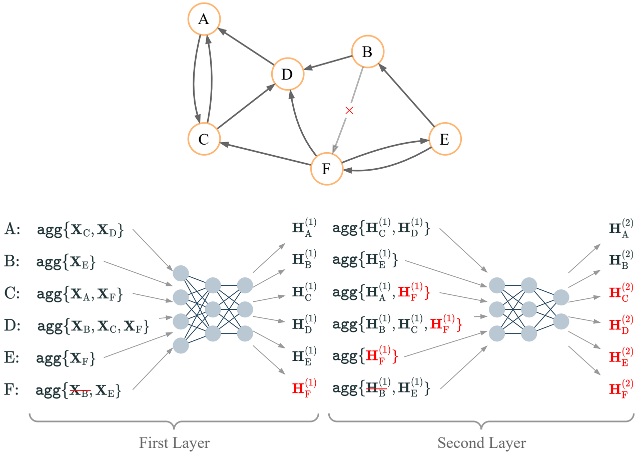
- e.g., an MLP



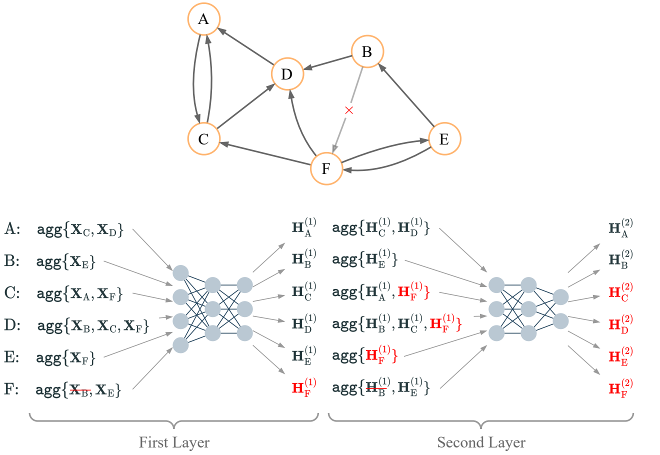
DP GNN Challenges: Exploding Sensitivity



DP GNN Challenges: Exploding Sensitivity

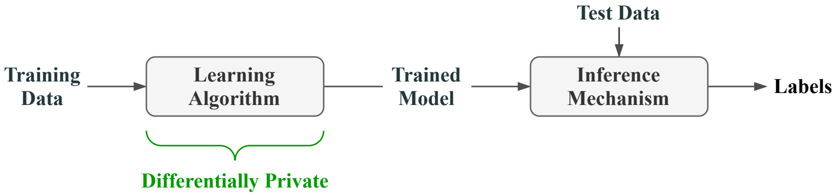


DP GNN Challenges: Exploding Sensitivity



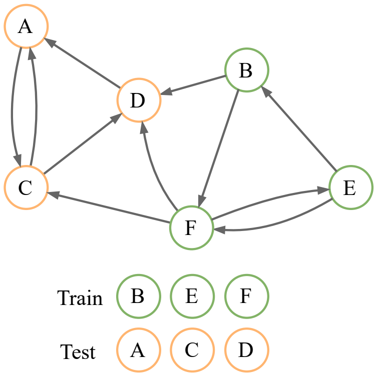
The number of affected outputs = $\mathcal{O}(\text{max degree}^{\text{num layers}})$

Private Learning: Standard Neural Nets



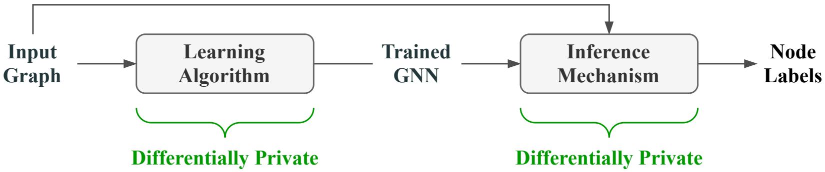
Inference is independent of the training data

- GNN re-uses graph data for inference
- Private information leaks at inference, even with a private model



Differentially Private GNN Challenges: Inference Privacy

Private Learning: Graph Neural Nets



Both training and inference should be private

Our Approach: Aggregation Perturbation

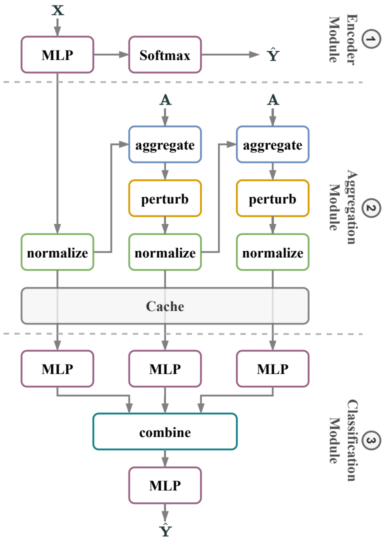
- **Aggregation Perturbation:** adding noise to output of the aggregation step
  - Prevents the exploding sensitivity problem by composing differentially private aggregation steps
  - Ensures inference privacy
- Applying aggregation perturbation to the conventional GNNs is costly
  - Every forward pass of the model consumes privacy budget
  - The excessive noise results in poor performance

Our Approach: Aggregation Perturbation

- **Aggregation Perturbation:** adding noise to output of the aggregation step
  - Prevents the exploding sensitivity problem by composing differentially private aggregation steps
  - Ensures inference privacy
- Applying aggregation perturbation to the conventional GNNs is **costly**
  - Every forward pass of the model consumes privacy budget
  - The excessive noise results in poor performance

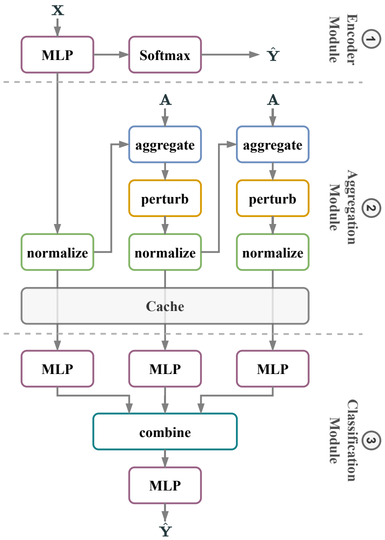
Need to tailor the GNN architecture to the private learning setting!

GNN with Aggregation Perturbation (GAP) [Sajadmanesh et al., 2022]



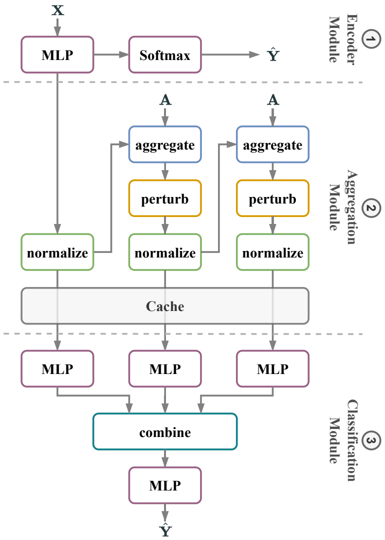
GNN with Aggregation Perturbation (GAP) [Sajadmanesh et al., 2022]

1. Encoder Module
    - Learns to encode node features into lower-dimensional representations
    - Does not use graph adjacency information



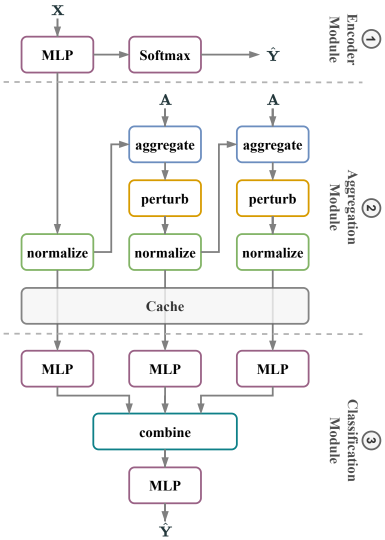
GNN with Aggregation Perturbation (GAP) [Sajadmanesh et al., 2022]

1. Encoder Module
   - Learns to encode node features into lower-dimensional representations
   - Does not use graph adjacency information
2. Aggregation Module
   - Computes aggregated node representations at multiple hops privately using the aggregation perturbation approach
   - Uses graph adjacency information

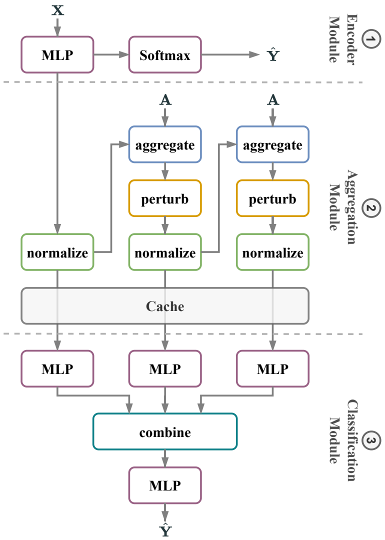


GNN with Aggregation Perturbation (GAP) [Sajadmanesh et al., 2022]

1. Encoder Module
    - Learns to encode node features into lower-dimensional representations
    - Does not use graph adjacency information
2. Aggregation Module
    - Computes aggregated node representations at multiple hops privately using the aggregation perturbation approach
    - Uses graph adjacency information
3. Classification Module
    - Learns to perform node-wise classification based on aggregated node representations
    - Does not re-use graph adjacency information

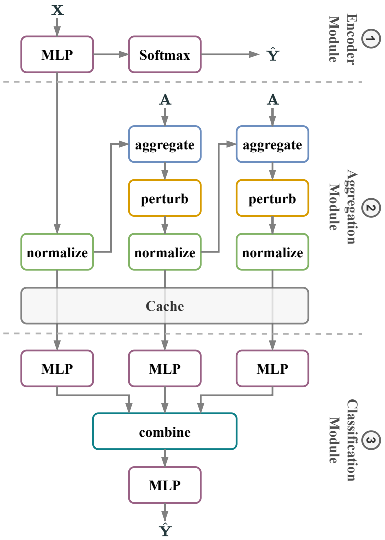


✓ Edge-level DP

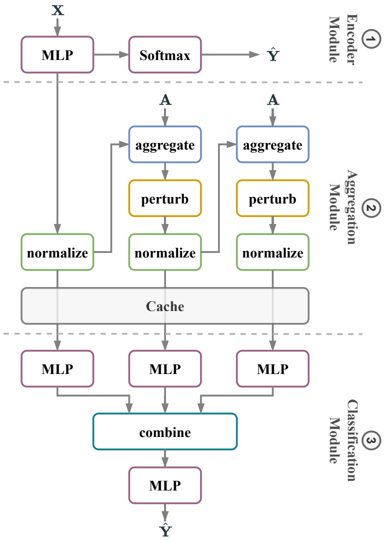


Advantages of GAP Architecture

- ✓ Edge-level DP
- ✓ Node-level DP through combination with DP-SGD
  - For bounded-degree graphs

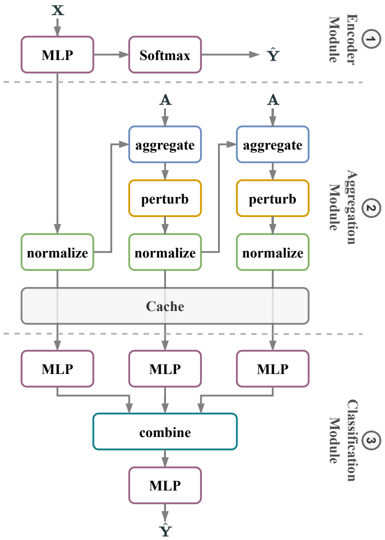


- ✓ Edge-level DP
- ✓ Node-level DP through combination with DP-SGD
  - For bounded-degree graphs
- ✓ Multi-hop aggregations



ADVANTAGES OF GAP ARCHITECTURE

- ✓ Edge-level DP
- ✓ Node-level DP through combination with DP-SGD
  - For bounded-degree graphs
- ✓ Multi-hop aggregations
- ✓ Zero-cost inference privacy



Experiment Settings: Datasets

- **Task:** Node Classification

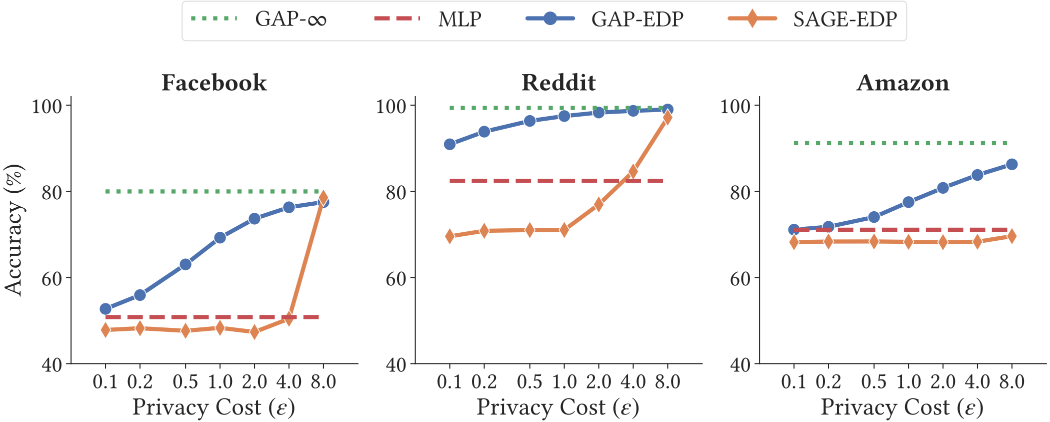
Dataset	Classes	Nodes	Edges	Features	Med. Degree
Facebook	6 Year	26,406 User	2,117,924 Friendship	501	62
Reddit	8 Community	116,713 Post	46,233,380 Mutual User	602	209
Amazon	10 Category	1,790,731 Product	80,966,832 Mutual Purchase	100	22

Comparison of Non-Private Methods

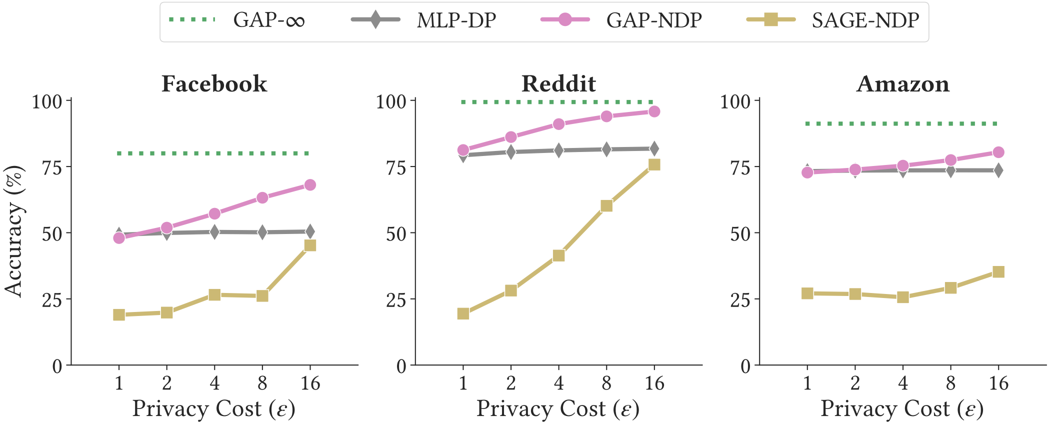
Accuracy of Non-Private Methods

Method	Facebook	Reddit	Amazon
GAP-∞	80.0 ± 0.48	**99.4 ± 0.02**	91.2 ± 0.07
SAGE-∞	**83.2 ± 0.68**	99.1 ± 0.01	**92.7 ± 0.09**

Edge-Level DP Accuracy-Privacy Trade-Off



Node-Level DP Accuracy-Privacy Trade-Off



Resiliency to Membership Inference Attack

Mean AUC of node-level membership inference attack.

Dataset	Method	$\epsilon = 1$	$\epsilon = 2$	$\epsilon = 4$	$\epsilon = 8$	$\epsilon = 16$	$\epsilon = \infty$
Facebook	GAP-NDP	50.16	50.25	50.61	51.11	52.66	81.67
	SAGE-NDP	50.25	50.20	50.23	50.17	50.20	62.49
	MLP-DP	50.32	50.72	52.13	53.44	54.77	81.57

Conclusion

- GNNs leak private information
  - They are vulnerable to privacy attacks
- Implementing DP in GNNs is challenging
  - Exploding sensitivity
  - Inference privacy
- Our Differentially Private GNN: GAP
  - Ensures both edge-level and node-level DP
  - Supports multi-hop aggregations
  - Provides inference privacy

Thank You!

Questions?

sajadmanesh@idiap.ch

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016).
Deep learning with differential privacy.
In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006).
Calibrating noise to sensitivity in private data analysis.
In *Theory of cryptography conference*, pages 265–284. Springer.

He, X., Jia, J., Backes, M., Gong, N. Z., and Zhang, Y. (2021).
Stealing links from graph neural networks.
In *30th {USENIX} Security Symposium ({USENIX} Security 21)*.

Olatunji, I. E., Nejdl, W., and Khosla, M. (2021).
Membership inference attack on graph neural networks.
arXiv preprint arXiv:2101.06570.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014).
Deepwalk: Online learning of social representations.
In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.

Sajadmanesh, S., Shamsabadi, A. S., Bellet, A., and Gatica-Perez, D. (2022).
Gap: Differentially private graph neural networks with aggregation perturbation.
arXiv preprint arXiv:2203.00949.