EFFICIENT MODEL UPDATES FOR APPROXIMATE UNLEARNING OF GRAPH-STRUCTURED DATA

Eli Chien∗  Chao Pan∗  Olgica Milenkovic
Department of Electrical and Computer Engineering
University of Illinois, Urbana-Champaign
{ichien3,chaopan2,milenkov}@illinois.edu

ABSTRACT

With the adoption of recent laws ensuring the “right to be forgotten”, the problem of machine unlearning has become of significant importance. This is particularly the case for graph-structured data, and learning tools specialized for such data, including graph neural networks (GNNs). This work introduces the first known approach for approximate graph unlearning with provable theoretical guarantees. The challenges in addressing the problem are two-fold. First, there exist multiple different types of unlearning requests that need to be considered, including node feature, edge and node unlearning. Second, to establish provable performance guarantees, one needs to carefully evaluate the process of feature mixing during propagation. We focus on analyzing Simple Graph Convolutions (SGC) and their generalized PageRank (GPR) extensions, thereby laying the theoretical foundations for unlearning GNNs. Empirical evaluations of six benchmark datasets demonstrate excellent performance/complexity/privacy trade-offs of our approach compared to complete retraining and general methods that do not leverage graph information. For example, unlearning 200 out of 1208 training nodes of the Cora dataset only leads to a 0.1% loss in test accuracy, but offers a 4-fold speed-up compared to complete retraining with a $(\epsilon, \delta) = (1, 10^{-4})$ “privacy cost”. We also exhibit a 12% increase in test accuracy for the same dataset when compared to unlearning methods that do not leverage graph information, with comparable time complexity and the same privacy guarantee. Our code is available online.1

1 I N T R O D U C T I O N

Machine learning algorithms are used in many application domains, including biology, computer vision and natural language processing. Relevant models are often trained either on third-party datasets, internal or customized subsets of publicly available user data. For example, many computer vision models are trained on images from Flickr users (Thomee et al., 2016; Guo et al., 2020) while many natural language processing (e.g., sentiment analysis) and recommender systems heavily rely on repositories such as IMDB (Maas et al., 2011). Furthermore, numerous ML classifiers in computational biology are trained on data from the UK Biobank (Sudlow et al., 2015), which represents a collection of genetic and medical records of roughly half a million participants (Ginart et al., 2019). With recent demands for increased data privacy, the above referenced and many other data repositories are facing increasing demands for data removal. Certain laws are already in place guaranteeing the rights of certified data removal, including the European Union’s General Data Protection Regulation (GDPR), the California Consumer Privacy Act (CCPA) and the Canadian Consumer Privacy Protection Act (CPPA) (Sekhari et al., 2021).

Removing user data from a dataset is insufficient to guarantee the desired level of privacy, since models trained on the original data may still contain information about their patterns and features. This consideration gave rise to a new research direction in machine learning, referred to as machine unlearning (Cao & Yang, 2015), in which the goal is to guarantee that the user data information is also removed from the trained model. Naively, one can retrain the model from scratch to meet

∗Equal contribution.

1https://github.com/thupchnsky/sgc_unlearn
We take the first step towards solving the approximate unlearning problem by performing a nontrivial theoretical analysis of some simplified GNN architectures. Inspired by the unstructured data certified removal procedure (Guo et al., 2020), we propose the first known approach for approximate graph unlearning. Our main contributions are as follows. First, we introduce three types of data removal requests for graph unlearning: Node feature unlearning, edge unlearning and node unlearning (see Figure 1). Second, we derive theoretical guarantees for approximate graph unlearning mechanisms for all three removal cases on SGC (Wu et al., 2019) and their GPR generalizations. In particular, we analyze $L_2$-regularized graph models trained with differentiable convex loss functions. The analysis is challenging since propagation on graphs “mixes” node features. Our analysis reveals that the degree of the unlearned node plays an important role in the unlearning process, while the number of propagation steps may or may not be important for different unlearning scenarios. To the best of our knowledge, the theoretical guarantees established in this work are the first provable approximate unlearning studies for graphs. Furthermore, the proposed analysis also encompasses node classification and node regression problems. Third, our empirical investigation on frequently used datasets for GNN learning shows that our method offers an excellent performance-complexity-privacy trade-off. For example, when unlearning 200 out of 1208 training nodes of the Cora dataset, our method offers comparable test accuracy as complete retraining, but offers a 4-fold speed-up with a $(\epsilon, \delta) = (1, 10^{-4})$ “privacy cost”. We also test our model on datasets for which removal requests are most likely to arise, including Amazon co-purchase networks.

Due to space limitations, all proofs and some detailed discussions are relegated to the Appendix.

2 RELATED WORKS

Machine unlearning and certified data removal. Cao & Yang (2015) introduced the concept of machine unlearning and proposed distributed learners for exact unlearning. Bourtoule et al. (2021) introduced sharding-based methods for unlearning, while Ginart et al. (2019) described unlearning approaches for $k$-means clustering. These works focused on exact unlearning: The unlearned model is required to perform identically to a completely retrained model. As an alternative, Guo et al. (2020) introduced a probabilistic definition of unlearning motivated by differential privacy (Dwork, 2011). Sekhari et al. (2021) studied the generalization performance of machine unlearning methods.
Golatkar et al. (2020) proposed heuristic-based selective forgetting in deep networks. These probabilistic approaches naturally allow for “approximate” unlearning. None of these works addressed the machine unlearning problem on graphs. To the best of our knowledge, the only work in this direction is GraphEraser (Chen et al., 2021). However, the strategy proposed therein uses sharding, which only works for exact unlearning and is hence completely different from our approximate approach. Also, the approach in Chen et al. (2021) relies on partitioning the graph using community detection methods. It therefore implicitly makes the assumption that the graph is homophilic which is not warranted in practice (Chien et al., 2021b; Lim et al., 2021). In contrast, our method works for arbitrary graphs and allows for approximate unlearning while ensuring excellent trade-offs among performance, privacy and complexity.

**Differential privacy (DP) and DP-GNNs.** Machine unlearning, especially the approximation version described in Guo et al. (2020), is closely related to differential privacy (Dwork 2011). In fact, differential privacy is a sufficient condition for machine unlearning. If a model is differentially private, then the adversary cannot distinguish whether the model is trained on the original dataset or on a dataset in which one data point is removed. Hence, even without model updating, a DP model will automatically unlearn the removed data point (see also the explanation in Ginart et al., 2019; Sekhari et al., 2021) and Figure 4. Although DP is a sufficient condition for unlearning, it is not a necessary condition. Also, most of the DP models suffer from a significant degradation in performance even when the privacy constraint is loose (Chaudhuri et al., 2011; Abadi et al., 2016). Machine unlearning can therefore be viewed as a means to trade-off between performance and computational cost, with complete retraining and DP on two different ends of the spectrum (Guo et al., 2020). Several recent works proposed DP-GNNs (Daigavane et al., 2021; Olatunji et al., 2021a; Wu et al., 2021; Sajadmanesh et al., 2022) – however, even for unlearning one single node or edge, these methods require a high “privacy cost” ($\epsilon \geq 5$) to learn with sufficient accuracy.

**Graph neural networks.** While GNNs are successfully used for many graph-related problems, accompanying theoretical analyses are usually difficult due to the combination of nonlinear feature transformation and graph propagation. Recently, several simplified GNN models were proposed that can further the theoretical understanding of their performance and scalability. SGCs (Wu et al., 2019) simplify Graph Convolutional Networks (GCNs) (Kipf & Welling, 2017) via linearization (i.e., through the removal of all nonlinearities); although SGC in general underperforms compared to state-of-the-art GNNs, they still offer competitive performance on many datasets. The analysis of SGCs elucidated the relationship between low-pass graph filtering and GCNs which reveals both advantages and potential limitations of GNNs. The GPR generalization of SGC is closely related to many important models that resolve different issues inherent to GNNs. For example, GPRGNN (Chien et al., 2021b) addresses the problem of universal learning on homophilic and heterophilic graph datasets and the issue of over-smoothing. SIGN (Frasca et al., 2020) based graph models and S^2GC (Zhu & Koniusz, 2020) allow for arbitrary sized mini-batch training, which improves the scalability and leads to further performance improvements of methods (Sun et al., 2021; Zhang et al., 2021a; Chien et al., 2022a) on the Open Graph Benchmark leaderboard (Hu et al., 2020).

Hence, developing approximate graph unlearning approaches for SGC and generalizations thereof is not only of theoretical interest, but also of practical importance.

### 3 Preliminaries

**Notation.** We reserve bold-font capital letters such as $\bf S$ for matrices and bold-font lowercase letters such as $\bf s$ for vectors. We use $\bf e_i$ to denote the $i^{th}$ standard basis, so that $\bf e_i^T \bf S$ and $\bf S e_i$ represent the $i^{th}$ row and column vector of $\bf S$, respectively. The absolute value $| \cdot |$ is applied component-wise on both matrices and vectors. We also use the symbols $\bf 1$ for the all-one vector and $\bf I$ for the identity matrix. Furthermore, we let $\bf G = (\bf V, E)$ stand for an undirected graph with node set $\bf V = [n]$ of size $n$ and edge set $\bf E$. The symbols $\bf A$ and $\bf D$ are used to denote the corresponding adjacency and node degree matrix, respectively. The feature matrix is denoted by $\bf X \in \mathbb{R}^{n \times F}$ and the features have dimension $F$; For binary classification, the label are summarized in $\bf Y \in \{-1, 1\}^n$, while the nonbinary case is discussed in Section 5. The relevant norms are $\| \cdot \|$, the $l_2$ norm, and $\| \cdot \|_F$, the Frobenius norm. Note that we use $\| \cdot \|$ for both row and column vectors to simplify the notation. The matrices $\bf A$ and $\bf D$ should not be confused with the symbols for an algorithm $\bf A$ and dataset $\bf D$. 

3
Certified removal. Let $A$ be a (randomized) learning algorithm that trains on $\mathcal{D}$, the set of data points before removal, and outputs a model $h \in \mathcal{H}$, where $\mathcal{H}$ represents a chosen space of models. The removal of a subset of points from $\mathcal{D}$ results in $\mathcal{D}'$. For instance, let $\mathcal{D} = (X, Y)$. Suppose we want to remove a data point, $(e_i^X, e_i^Y)$ from $\mathcal{D}$, resulting in $\mathcal{D}' = (X', Y')$. Here, $X', Y'$ are equal to $X, Y$, respectively, except that the row corresponding to the removed data point is deleted. Given $\epsilon > 0$, an unlearning algorithm $M$ applied to $A(\mathcal{D})$ is said to guarantee an $(\epsilon, \delta)$-certified removal for $A$, where $\epsilon, \delta > 0$ and $X'$ denotes the space of possible datasets, if

$$
\forall \mathcal{T} \subseteq \mathcal{H}, \mathcal{D} \subseteq X, i \in [n] : \mathbb{P}(M(A(\mathcal{D}), \mathcal{D}, \mathcal{D}') \in \mathcal{T}) \leq \exp(\epsilon) \mathbb{P}(A(\mathcal{D}') \in \mathcal{T}) + \delta,
$$

$$
\mathbb{P}(A(\mathcal{D}') \in \mathcal{T}) \leq \exp(\epsilon) \mathbb{P}(M(A(\mathcal{D}), \mathcal{D}, \mathcal{D}') \in \mathcal{T}) + \delta.
$$

(1)

This definition is related to $(\epsilon, \delta)$-DP \cite{dwork2011algorithmic}, except that we are allowed to update the model based on the removed point (see Figure 3). An $(\epsilon, \delta)$-certified removal method guarantees that the updated model $M(A(\mathcal{D}), \mathcal{D}, \mathcal{D}')$ is “approximately” the same as the model $A(\mathcal{D}')$ obtained by retraining from scratch. Thus, any information about the removed data $\mathcal{D} \setminus \mathcal{D}'$ is “approximately” eliminated from the model. Ideally, we would like to design $M$ such that it satisfies equation (1) and has a complexity that is significantly smaller than that of complete retraining.

4 APPROXIMATE GRAPH UNLEARNING WITH THEORETICAL GUARANTEES

Unlike standard machine unlearning, approximate graph unlearning uses datasets that contain not only node features $X$ but also the graph topology $A$, and therefore require different data removal procedures. We focus on node classification, for which the training dataset equals $\mathcal{D} = (X,Y_{T_r}, A)$. Here, $Y_{T_r}$ is identical to $Y$ on rows indexed by points of the training set $T_r$ while the remaining rows are all zeros. Without loss of generality, we assume that the training set comprises the first $m$ nodes (i.e., $T_r = [m]$), where $m \leq n$. An unlearning method $M$ achieves $(\epsilon, \delta)$-approximate graph unlearning with algorithm $A$ if equation (1) is satisfied for $\mathcal{D} = (X, Y_{T_r}, A)$ and $\mathcal{D}'$, which differ based on the type of requests: Node feature unlearning, edge unlearning, and node unlearning.

4.1 UNLEARNING SGC AND COMPARISON WITH UNSTRUCTURED UNLEARNING

SGC is a simplification of GCN obtained by removing all nonlinearities from the latter model. This leads to the following update rule: $P^KXW = ZW$, where $W$ denotes the matrix of learnable weights, $K \geq 0$ equals the number of propagation steps and $P$ denotes the one-step propagation matrix. The standard choice of the propagation matrix is the symmetric normalized adjacency matrix with self-loops, $P = D^{-1/2}AD^{-1/2}$, where $A = A + \mathbf{I}$ and $D$ equals the degree matrix with respect to $A$. We will work with the asymmetric normalized version of $P$, $P = D^{-1}A$. This choice is made purely for analytical purposes and our empirical results confirm that this normalization ensures the competitive performance of our unlearning methods.

The resulting node embedding is used for node classification by choosing an appropriate loss (i.e., logistic loss) and minimizing the $L_2$-regularized empirical risk. For binary classification, $W$ can be replaced by a vector $w$; the loss equals $L(w, \mathcal{D}) = \sum_{i: e_i^Y \neq 0} \left( \ell(e_i^Z w, e_i^Y_{T_r}) + \frac{1}{2} \|w\|^2 \right)$, where $\ell(e_i^Z w, e_i^Y_{T_r})$ is a convex loss function that is differentiable everywhere. We also write $w^* = A(D) = \arg \min_w L(w, \mathcal{D})$, where the optimizer is unique whenever $\lambda > 0$.

We start with a high-level description of the approximate unstructured unlearning approach introduced in \cite{guo2020approximate}. Note that “certified removal” in the context of the former work refers to approximate unlearning that provably satisfies (1). Let us denote the Hessian of $L(\cdot; \mathcal{D}')$ at $w^*$ by $H_{w^*} = \nabla^2 L(w^*; \mathcal{D}')$. The authors of \cite{guo2020approximate} propose the following mechanism for unlearning the $m^{th}$ training point: $w^* = M(w^*, \mathcal{D}, \mathcal{D} \setminus \mathcal{D}') = w^* + H_{w^*} \Delta_{guo}$, where $\Delta_{guo} = \lambda w^* + \nabla \ell(e_m^w Xw^*, e_m^r Y_{T_r})$. When $\nabla L(w^*, \mathcal{D}') = 0$, then $w^*$ is the unique optimizer of $L(\cdot; \mathcal{D}')$. If $\nabla L(w^*, \mathcal{D}') \neq 0$, then information about the removed data point remains present in the model. One can show that the gradient residual norm $\|\nabla L(w^*, \mathcal{D} \setminus \mathcal{D}')\|$ determines the error of $w^*$ when used to approximate the true minimizer of $L(\cdot; \mathcal{D}')$ \cite{guo2020approximate}. Hence, upper bounds on $\|\nabla L(w^*, \mathcal{D} \setminus \mathcal{D}')\|$ can be used to establish approximate unlearning guarantees. More precisely, assume that we have $\|\nabla L(w^*, \mathcal{D}')\| \leq \epsilon'$ for some $\epsilon' > 0$. Furthermore, consider training with the noisy loss $L_n(w, \mathcal{D}) = \sum_{i: e_i^Y \neq 0} \left( \ell(e_i^X w, e_i^Y_{T_r}) + \frac{1}{2} \|w\|^2 \right) + b^T w$, where $b$ is drawn randomly according to some distribution. Then one can leverage the following result.
Theorem 4.1 (Theorem 3 from Guo et al. [2020]). Let \( A \) be the learning algorithm that returns the unique optimum of the loss \( L_0(w, \mathcal{D}) \). Suppose that \( \| \nabla L_b(w^*, \mathcal{D}') \| \leq \epsilon' \) for some computable bound \( \epsilon' > 0 \) independent on \( b \) and achieved by \( M \). If \( b \sim \mathcal{N}(0, (c_0\epsilon')^2) \) with \( c_0 > 0 \), then \( M \) satisfies (1) with parameters \((\epsilon, \delta)\) for algorithm \( A \) applied to \( \mathcal{D}' \), where \( \delta = 1.5e^{-\epsilon_0^2/2} \).

Hence, if we can prove that \( \| \nabla L(w^*, \mathcal{D}) \| \) is appropriately bounded for the graph setting as well, then the unlearning mechanism \( M \) will ensure \((\epsilon, \delta)-\)approximate graph unlearning. One of the main contributions of Guo et al. [2020] is to bound the gradient residual norm \( \| \nabla L(w^*, \mathcal{D}) \| \) of the proposed unlearning mechanism with \( \Delta_{guo} = \lambda w^* + \nabla \ell(e_m^T Xw^*, e_m^T Y_{T'}) \).

Motivated by the unlearning approach from Guo et al. [2020] pertaining to unstructured data, we design an unlearning mechanism for graphs. We generalize their unlearning mechanism by replacing \( \Delta_{guo} \) with \( \Delta = \nabla L(w^*, \mathcal{D}) - \nabla L(w^*, \mathcal{D}') \). As an demonstrative example, for node unlearning we consequently have

\[
\Delta = \lambda w^* + \nabla \ell(e_m^T Xw^*, e_m^T Y_{T'}) + \sum_{i=1}^{m-1} \left[ \nabla \ell(e_i^T Zw^*, e_i^T Y_{T'}) - \nabla \ell(e_i^T Z^*w^*, e_i^T Y_{T'}) \right]. \tag{2}
\]

Note that our generalized unlearning mechanism matches that of Guo et al. [2020] when no graph information is present. This can be seen by setting \( K = 0 \), which leads to \( Z = X \) and \( e_i^T Z = e_i^T Z^* \), \( \forall i \in [m - 1] \). Hence, the third term in equation (2) is zero and thus \( \Delta = \Delta_{guo} \). Note that when graph information is present, the third term in equation (2) is, in general, bounded away from zero. This term captures the impact of unlearning node \( m \) on all remaining training nodes \([m - 1]\), and including effects pertaining to edge and feature removal. This not only highlights the necessity of investigating generalized unlearning mechanisms, but also the main difficulty of extending the analysis of Guo et al. [2020] to graphs. A more detailed discussion regarding the intuition behind our approach can be found in Appendix A.3. The main technical contribution of our work is to establish bounds of the gradient residual norm for all three types of graph unlearning scenarios. For this analysis, we need the loss function \( \ell \) to satisfy the following properties.

Assumption 4.2. For any \( \mathcal{D} \), \( i \in [n] \) and \( w \in \mathbb{R}^F \): (1) \( \| \nabla \ell(e_i^T Zw, e_i^T Y) \| \leq c \) (i.e. the norm of \( \nabla \ell \) is \( c \)-bounded); (2) \( \ell'' \) is \( \gamma_2 \)-Lipschitz; (3) \( \| e_i^T X \| \leq 1 \); (4) \( \ell'' \) is \( \gamma_1 \)-Lipschitz; (5) \( \ell' \) is \( c_1 \)-bounded.

Assumptions (1)-(3) are also needed for unstructured unlearning of linear classifiers (Guo et al., 2020). To account for graph-structured data, we require additional assumptions (4)-(5) to establish worst-case bounds. The additional assumptions may be avoided when working with data-dependent bounds (Section 5). In all subsequent derivations, we assume that the unlearned data point corresponds to the \( m \)th node for node feature and node unlearning; for edge unlearning, we wish to unlearn the edge \((1, m)\). Generalizations for multiple unlearning requests are discussed in Section 5.

4.2 Node Feature Unlearning for SGCs

We start with the simplest type of unlearning – node feature unlearning – for SGCs. In this case, we remove the node feature and label of one node from \( \mathcal{D} \), resulting in \( \mathcal{D}' = \{X', Y_{T'}, A\} \). The matrices \( X', Y_{T'} \) are identical to \( X, Y_{T'} \), respectively, except for the \( m \)th row of the former being set to zero. Note that in this case, the graph structure remains unchanged.

Theorem 4.3. Suppose that Assumption 4.2 holds. For the node feature unlearning scenario and \( Z = P^T X \) and \( P = D^{-1} A \), we have

\[
\| \nabla L(w^*, \mathcal{D}') \| = \| (H_{w^*} - H_{w^-})H_{w}^{-1} \Delta \| \leq \frac{\gamma_2 (2c\lambda + (c\gamma_1 + \lambda c_1)\hat{D}_{mn})^2}{\lambda^4 (m - 1)}, \tag{3}
\]

where \( H_{w} \) denotes the Hessian of \( L(\cdot; \mathcal{D}') \) at \( w = w^* + \eta H_{w}^{-1} \Delta \) for some \( \eta \in [0, 1] \). A similar conclusion holds for the case when we wish to unlearn node features of a node that is not in \( T_r \). In this case we just replace \( D_{mn} \) by the degree of the corresponding node. This result shows that the norm bound is large if the unlearned node has a large degree, since a large-degree node will affect the values of many rows in \( Z \). Our result also demonstrates that the norm bound is independent of \( K \), due to the fact that \( P \) is right stochastic. We provide next a sketch of the proof to illustrate the analytical challenges of graph unlearning compared to those of unstructured data unlearning.
Although for node feature unlearning the graph topology does not change, all rows of $Z = P^K X$ may potentially change due to graph information propagation. Thus, the original analysis from [Guo et al., 2020], which corresponds to the special case $Z = X$, cannot be applied directly. There are two particular challenges. First, in order to ensure that the norm of each row of $Z$ is bounded by 1. We provide Lemma A.1 to guarantee this. It is critical to choose $P = D^{-1} A$ since all other choices of degree normalization lead to worse bounds (see Appendix A.1.1). The second and more difficult challenge is to bound $\| \Delta \|$ when $Z = X$, the third term in equation (2) is exactly zero, in accordance with [Guo et al., 2020]. Due to graph propagation, we have to further bound the norm of the third term, which is highly nontrivial since the upper bound is not allowed to grow with $m$ or $n$. We first focus on one of the $m - 1$ terms in the sum. Using Assumption 4.2, one can bound this term by $\|e^T_i (Z - Z')\|$ (we suppressed the dependency on $\lambda, c, c_1$ and $c_2$ for simplicity). The key analytical novelty is to explore the sparsity of $Z - Z'$ is obtained by replacing the all-zero matrix corresponding to $A$ by $(A - \lambda I)$. The matrix $A'$ is also a probability vector whenever $p$ is a probability vector. Clearly, $e_m$ is also a probability vector. Since all diagonal entries of $D^{-1}$ are nonnegative and upper bounded by 1 given the self-loops for all nodes, $1^T D^{-1} p \leq 1^T p = 1$ for any probability vector $p$. Hence, the term above is bounded by $D_{mm}$. The bound depends on $D_{mm}$ and does not increase with $m$ or $K$. Although node feature unlearning is the simplest case of graph unlearning, our sketch of the proof illustrates the difficulties associated with bounding the third term in $\Delta$. Similar, but more complicated approaches are needed for the analysis of edge unlearning and node unlearning.

### 4.3 Edge and Node Unlearning for SGCs and GPRs Extensions

#### Edge Unlearning for SGC

We describe next the bounds for edge unlearning and highlight the technical issues arising in the analysis of this setting. Here, we remove one edge $(1, m)$ from $D$, resulting in $D' = (X, Y_{T'}, A')$. The matrix $A'$ is identical to $A$ except for $A'_{1m} = A'_{m1} = 0$. Furthermore, $D'$ is the degree matrix corresponding to $A'$. Note that the node features and labels remain unchanged.

**Theorem 4.4.** Suppose that Assumption 4.2 holds. Under the edge unlearning scenario, and for $P = D^{-1} A$ and $Z = P^K X$, we have

$$\| \nabla L(w^-, D') \| = \| (H_{w^\perp} - H_{w^-}) H_{w^-}^{-1} \Delta \| \leq \frac{16\gamma_2 K^2 (c\gamma_1 + c_1 \lambda)^2}{\lambda^4 m}. \tag{4}$$

Similar to what holds for the node feature unlearning case, Theorem 4.4 still holds when neither of the two end nodes of the removed edge belongs to $T_r$.

#### Node Unlearning for SGC

We now discuss the most difficult case, node unlearning. In this case, one node is entirely removed from $D$, including node features, labels and edges. This results in $D' = (X', Y_{T'}, A')$. The matrices $X', Y_{T'}$ are defined similarly to those described for node feature unlearning. The matrix $A'$ is obtained by replacing the $m^{th}$ row and column in $A$ by all-zeros (similar changes are introduced in $A$, with $A'_{mm} = 0$). For simplicity, we let $D'_{mm} = 1$ as this assumption does not affect the propagation results.

**Theorem 4.5.** Suppose that Assumption 4.2 holds. For the node unlearning scenario and $Z = P^K X$ and $P = D^{-1} A$, we have

$$\| \nabla L(w^-, D') \| = \| (H_{w^\perp} - H_{w^-}) H_{w^-}^{-1} \Delta \| \leq \frac{\gamma_2 \left(2c\lambda + K (c\gamma_1 + c_1 \lambda) \left(2 D_{mm} - 1 \right) \right)^2}{\lambda^4 (m - 1)}. \tag{5}$$

The main challenge arises in the proof of Theorem 4.4 and 4.5 is bounding $\| \Delta \|$ appropriately. Unlike for the node feature unlearning case, now both graph structure and node features can change.
due to the unlearning request. We establish a series of lemmas to characterize the difference between $Z$ and $Z'$, which play important roles in our proofs (see Appendix A.8 and A.9 for complete proofs).

**Approximate graph unlearning in GPR-based model.** Our analysis can be extended to Generalized PageRank (GPR)-based models [Li et al., 2019]. The definition of GPR is $\sum_{k=0}^{K} \theta_k P^k S$, where $S$ denotes a node feature or node embedding. The learnable weights $\theta_k$ are called GPR weights and different choices for the weights lead to different propagation rules (Jeh & Widom, 2003; Chung, 2007). GPR-type propagations include SGC and APPNP rules as special cases (Chien et al., 2021b). If we use linearly transformed features $S = X W$, for some weight matrix $W$, the GPR rule can be rewritten as $Z W = \frac{1}{K+1} \left[ X, P X, P^2 X, \ldots, P^K X \right] W$. This constitutes a concatenation of the steps from 0 up to $K$. The learnable weight matrix $W \in \mathbb{R}^{(K+1)F \times C}$ combines $\theta_k$ and $W$. These represent linearizations of GPR-GNNs (Chien et al., 2021b) and SIGNs (Frasca et al., 2020), simple yet useful models for learning on graphs. For simplicity, we only describe the results for node feature unlearning and delegate the analysis of edge and node unlearning to Appendix A.10.

**Theorem 4.6.** Suppose that Assumption 4.2 holds and considers the node feature unlearning case. For $Z = \frac{1}{K+1} \left[ X, P X, P^2 X, \ldots, P^K X \right]$ and $P = D^{-1} A$, we have

$$
\| \nabla L(w^-, D') \| = \| (H_{w^+} - H_{w^-}) H_{w^-}^{-1} \Delta \| \leq \gamma_2 (2 c_\lambda + (c\gamma_1 + c_1) D_{nn})^2 \lambda^4 / (m - 1).
$$

Note that the resulting bound is the same as the bound in Theorem 4.3. This is due to the fact that we used the normalization factor $1/K+1$ in $Z$. Hence, given the same noise level, the GPR-based models are more sensitive when we trained on the noisy loss $L_w$. Whether the general high-level performance of GPR can overcompensate this drawback depends on the actual datasets considered.

## 5 Empirical Aspects of Approximate Graph Unlearning

**Logistic and least-squares regression on graphs.** For binary logistic regression, the loss equals $\ell(e^T Z w, e^T Y_T) = -\log(\sigma(e^T Y_T, e^T Z w))$, where $\sigma(x) = 1/(1 + \exp(-x))$ denotes the sigmoid function. As shown in Guo et al. (2020), the assumptions (1)-(3) in Assumption 4.2 are satisfied with $c_1 = 1$ and $\gamma_2 = 1/4$. By standard analysis, we show that our loss satisfies (4) and (5) in Assumption 4.2 with $\gamma_1 = 1/4$ and $c_1 = 1$. For multi-class logistic regression, one can adapt the “one-versus-all other-classes” strategy which leads to the same result. For least-square regression, since the hessian is independent of $w$ our approach offers $(0, 0)$-approximate graph unlearning even without loss perturbations. See Appendix A.5 for the complete discussion and derivation.

**Sequential unlearning.** In practice, multiple users may request unlearning. Hence, it is desirable to have a model that supports sequential unlearning of all types of data points. One can leverage the same proof as in Guo et al. (2020) (induction coupled with the triangle inequality) to show that the resulting gradient residual norm bound equals $T e'$ at the $T^{th}$ unlearning request, where $e'$ is the bound for a single instance of approximate graph unlearning.

**Data-dependent bounds.** The gradient residual norm bounds derived for different types of approximate graph unlearning contain a constant factor $1/\lambda^4$, and may be loose in practice. Following Guo et al. (2020), we also examined data dependent bounds.

**Corollary 5.1.** (Application of Corollary 1 in Guo et al. (2020)). For all three graph unlearning scenarios, we have $\| \nabla L(w^-, D') \| \leq \gamma_2 \| Z \|_2 \| H_{w^-}^{-1} \Delta \| \| Z H_{w^-}^{-1} \Delta \|.$

Hence, there are two ways to accomplish approximate graph unlearning. If we do not allow any retraining, we have to leverage the worst case bound in Section 4 based on the expected number of unlearning requests. Importantly, we will also need to constrain the node degree of nodes to be unlearned (i.e., do not allow for unlearning hub nodes), for both node feature and node unlearning. Otherwise, we can select the noise standard deviation $\alpha$, $\epsilon$ and $\delta$ and compute the corresponding “privacy budget” $\alpha \epsilon / \sqrt{2 \log(1.5/\delta)}$. Once the accumulated gradient residual norm exceeds this budget, we retrain the model from scratch. Note that this still greatly reduces the time complexity compare to retraining the model for every unlearning request (see Section 6). We also relegate the pseudo-code of our method leveraging data-dependent bounds for sequential unlearning in Appendix A.6.
6 Experiment

Settings. We test our methods on benchmark datasets for graph learning, including Cora, Citseer, Pubmed (Sen et al. 2008; Yang et al. 2016; Fey & Lenssen 2019) and large-scale dataset ogbn-arxiv (Hu et al. 2020) and Amazon co-purchase networks Computers and Photo (McAuley et al., 2015; Shchur et al., 2018). We either use the public splitting or random splitting based on similar rules as public splitting and focus on node classification. Following Guo et al. (2020), we use LBFGS as the optimizer for all methods due to its high efficiency on strongly convex problems. Unless specified otherwise, we fix $K = 2$, $\delta = 10^{-4}$, $\lambda = 10^{-2}$, $\epsilon = 1$, $\alpha = 0.1$ for all experiments, and average the results over 5 independent trials with random initializations. Our baseline methods include complete retraining with graph information after each unlearning request (SGC Retraining), complete retraining without graph information after each unlearning request (No Graph Retraining), and Unstructured Unlearning (Guo et al. 2020). Additional details can be found in Appendix A.19.

Bounds on the gradient residual norm. The first row of Figure 2 compares the values of both worst-case bounds computed in Section 4 and data-dependent bounds computed from Corollary 5.1 with the true value of the gradient residual norm (True Norm). For simplicity, we set $\alpha = 0$ during training. The observation is that the worst-case bounds are looser than the data-dependent bounds, and both bounds are indeed valid upper bounds for the actual gradient residual norm.

Dependency on node degrees. While an upper bound does not necessarily capture the dependency of each term correctly, we show in Figure 3 (a) and (b) that our Theorem 4.5 and 4.6 indeed do so. Here, each point corresponds to unlearning one node. We test for all nodes in the training set $T_\tau$ and fix $\lambda = 10^{-4}$, $\alpha = 0$. Our results show that unlearning a large-degree node is more expensive in terms of the privacy budget (i.e., it induces a larger gradient residual norm). For other datasets, refer to Appendix A.19.
Figure 3: (a), (b) Simulation verification of the result in Theorem 4.5 and 4.6 pertaining to node degrees. (d), (f) Accumulated unlearning time as a function of the number of removed nodes. The unlearning time of Unstructured Unlearning is often higher than that of our proposed approximate graph unlearning algorithms, because the number of retraining steps needed may be larger. (c), (e) Performance of approximate graph unlearning methods on different datasets. We set $\alpha = 10$, $\lambda = 10^{-4}$ for Computers and $\lambda = 10^{-4}$ for ogbn-arxiv. The number of repeated trials is 3 due to the large amount of removed data. (g) Tradeoff between privacy $\epsilon$ and performance. To achieve similar numbers of retraining, we set $\alpha \epsilon = 0.1$. (h) The number of data points predicted by the membership inference attack model to lie in the training set.

**Performance of approximate graph unlearning methods.** The time complexity of unlearning and test accuracy of our proposed approximate graph unlearning methods after unlearning is shown in Figures 2 and 3. It shows that: (1) Leveraging graph information is necessary when designing unlearning methods for node classification tasks. (2) Our method supports unlearning a large proportion of data points with a small loss in test accuracy. (3) Our method is around $4\times$ faster than completely retraining the model after each unlearning request. (4) Our methods have robust performance regardless of the scale of the datasets. For more results see Appendix A.19.

**Trade-off amongst privacy, performance and time complexity.** As indicated in Theorem 4.1, there is a trade-off amongst privacy, performance and time complexity. Comparing to exact unlearning (i.e. SGC retraining), allowing approximate unlearning gives $4\times$ speedup in time with competitive performance. We further examine this trade-off by fixing $\lambda$ and $\delta$, then the trade-off is controlled by $\epsilon$ and $\alpha$. The results are shown in Figure 3 (g) for Cora, where we set $\alpha \epsilon = 0.1$. The test accuracy increases when we relax our constraints on $\epsilon$, which agrees with our intuition. Remarkably, we can still obtain competitive performance with SGC Retraining when we require $\epsilon$ to be as small as 1. In contrast, one needs at least $\epsilon \geq 5$ to unlearn even one node or edge by leveraging state-of-the-art DP-GNNs [Sajadmanesh et al., 2022; Daigavane et al., 2021] for reasonable performance, albeit our tested datasets are different. This shows the benefit of our approximate graph unlearning method as opposed to both retraining from scratch and DP-GNNs. Unfortunately, the codes of these DP-GNNs are not publicly available, which prevents us from testing them on our datasets in a unified treatment.

**Membership inference attacks on unlearned models.** We performed experiments pertaining to the node unlearning task and applied the membership inference (MI) attack for GNNs reported in [Olatunji et al., 2021b] on our updated model. The experimental details are discussed in Appendix A.19. As shown in Figure 3 (h), even for full SGC retraining the attack can still identify parts of the removed nodes in the training set (for relevant explanations, see Appendix A.2), and the result of SGC node unlearning is slightly worse (w.r.t privacy) than retraining since our algorithm performs approximate unlearning. Note that the performance of the MI attack on the original model is consistent with the results in [Olatunji et al., 2021b] and significantly worse than both our unlearning and complete retraining method. The results also highlight the fact that the privacy definition considered in MI attacks and approximate unlearning are different (see Appendix A.2). Nevertheless, experiments show that our method offers similar privacy-preserving performance (in terms of MI) as complete retraining, and better performance compared to just using the original model without unlearning features.
ACKNOWLEDGMENTS

This work was funded by NSF grants 1816913 and 1956384. The authors thank Wei-Ning Chen and Pan Li for the helpful discussion.

REFERENCES


A.1 CONCLUSION

We introduced the first known framework for approximate graph unlearning. In this setting, new analytical unlearning challenges had to be addressed due to the presence of complex graph feature and topology data. Our analytical contributions pertain to novel proof techniques for approximate graph unlearning, while our empirical studies on six benchmark datasets established fundamental performance-complexity trade-offs between unlearning and complete retraining.
A.2 Future research directions and limitations

Batch unlearning. In practice, it is likely that we not only require sequential unlearning, but also batch unlearning: A number of users may request their data to be unlearned within a certain (short) time frame. The approach in [Guo et al. (2020)] can ensure certified removal even in this scenario. The generalization of our approach for batch unlearning is also possible, but will be discussed elsewhere.

Nonlinear models and hypergraph extension. Also akin to what was described in [Guo et al. (2020)], we can leverage pre-trained (nonlinear) feature extractors or special graph feature transforms to further improve the performance of the overall model. For example, [Chien et al. (2022a)] proposed a node feature extraction method termed GIANT-XRT that greatly improves the performance of simple network models such as MLP and SGC. If a public dataset is never subjected to unlearning, one can pre-train GIANT-XRT on that dataset and use it for subsequent approximate graph unlearning. If such a public dataset is unavailable, we have to make the node feature extractor unlearning, one can pre-train GIANT-XRT on that dataset and use it for subsequent approximate graph unlearning automatically and theoretically ensures that one cannot infer information about the unlearned data point from the updated model (if one chooses to set $\epsilon, \delta$ to 0), it remains an open problem whether we can certify this privacy-preserving performance. Note that privacy-based attacks like the membership inference attack [Shokri et al. (2017); Olatunji et al. (2021b)] have completely different design goals and may not work well in unlearning practice. For example, assume that there are two nodes that share similar features and neighborhood structures, come from the same class in the graph and are both included in the training set. This scenario frequently arises in practice, especially for graphs with strong homophily properties. In this case, even if we unlearn one of the nodes, the attack model will still have a high probability of recognizing the unlearned node in the training set due to the presence of the “similar” node. Thus, the viability of using the results returned by the attack models to assess the performance of an unlearner is not clear. This is also verified by some preliminary experiments on node unlearning tasks described in Section A.19 and the main text.

Societal impacts. The authors believe that for medical and biological sciences research, the right to be forgotten may significantly set back potentially life-saving discoveries due to the need to have access to many diverse data samples. But current trends seem to favor privacy over discovery rates and timings. Hence, a compromise between data availability and the right to be forgotten has to be established in the near future.

One current limitation of our work is that the newly proposed proof techniques do not apply to general graph neural networks where nonlinear activation functions are used. Nevertheless, our work is the first step towards developing approximate graph unlearning approaches for general GNNs.

A.3 Intuition behind the model update rule

Our unlearning mechanism proposed in Section 4 is

$$w^− = w^* + \left[∇^2 L(w^*, D')\right]^{-1} \left[∇ L(w^*, D') − ∇ L(w^*, D')\right],$$

and the intuition is stated as follows. Our goal for the updated model is $∇ L(w^−, D') = 0$. By Taylor series we have that

$$∇ L(w^−, D') \approx ∇ L(w^*, D') + ∇^2 L(w^*, D')(w^− − w^*) = 0.$$
Therefore, we have
\[ w^- - w^* = \left[ \nabla^2 L(w^*, D') \right]^{-1} \left[ 0 - \nabla L(w^*, D') \right] \]
\[ w^- = w^* + \left[ \nabla^2 L(w^*, D') \right]^{-1} \left[ \nabla L(w^*, D) - \nabla L(w^*, D') \right]. \]
The last equality holds due to the fact that \( w^* \) should be the unique optimizer for the strongly convex loss \( L(w, D) \) over the entire dataset \( D \).

A.4 ADDITIONAL ILLUSTRATIONS

(a) DP approach
\[ \text{Learning algorithm } A(C) \]
\[ W \sim p_A(C; D) \]
\[ W' \sim p_A(C; D') \]
\[ \text{DP requirement: } p_A(C; D) \approx p_A(C; D') \]

(b) Machine unlearning approach
\[ \text{Learning algorithm } A(C) \]
\[ W \sim p_A(C; D) \]
\[ W' \sim p_A(C; D') \]
\[ \text{Unlearning requirement: } p_A(C; W, D, D/\hat{D}) \approx p_A(C; \hat{D}') \]
\[ \text{Unlearning algorithm } M(W, D, D/\hat{D}') \]
\[ W' \sim p_A(C; D') \]

Figure 4: Difference between machine unlearning (as defined in Guo et al. [2020]) and Differential Privacy (DP).

A.5 ADDITIONAL DISCUSSIONS

Details on on Assumption 4.2. Assumptions (2), (4) and (5) in our model and that of Guo et al. [2020] require Lipschitz conditions with respect to the first argument of \( \ell \), but not the second. We also implicitly assume that the second argument (corresponding to labels) does not effect the norm of \( \gamma_1 \).

Least-squares and logistic regression on graphs. Paralleling once again the results of Guo et al. [2020], it is clear that our approximate graph unlearning mechanism can be used in conjunction with least-squares and logistic regressions. For example, node classification can be performed using a logistic loss. The node regression problem described in Ma et al. (2020); Jia & Benson (2020), it is clear that our approximate graph unlearning mechanism can be used in conjunction with least-squares and logistic regressions.

Using some simple algebra, one can also prove that \( \ell'(e_i^T Z w, e_i^T Y_{T'}) = (e_i^T Z w - e_i^T Y_{T'})^2 \). Note that its Hessian is of the form \( (e_i^T Z)^T e_i^T Z \), which does not depend on \( w \). Thus, based on the same arguments presented in Guo et al. [2020], our proposed unlearning method \( M \) offers \((0,0)\)-approximate graph unlearning even without loss perturbations.

Least-squares and logistic regression on graphs. Paralleling once again the results of Guo et al. [2020], it is clear that our approximate graph unlearning mechanism can be used in conjunction with least-squares and logistic regressions. For example, node classification can be performed using a logistic loss. The node regression problem described in Ma et al. (2020); Jia & Benson (2020), it is clear that our approximate graph unlearning mechanism can be used in conjunction with least-squares and logistic regressions.

A.6 ALGORITHMIC DETAILS

The pseudo-codes for training removal-enabled models and the removal procedure for the case of binary classification are presented below. Note that this procedure is the same for all three types of removal requests (node feature unlearning, edge unlearning and node unlearning). During training, we add a random linear term to the training loss by sampling a Gaussian noise vector \( b \). The choice of standard deviation \( \alpha \) determines the “privacy budget” \( \alpha \epsilon / \sqrt{2 \log(1.5/\delta)} \) as shown in Section 5.
Algorithm 1 Training procedure

1: **input:** Training data $Z \in \mathbb{R}^{m \times d}$, training labels $Y \in \mathbb{R}^m$, loss $\ell$, parameters $\alpha, \lambda > 0$.
2: Sample the noise vector $b \sim N(0, \alpha^2 d)$.
3: $w^* = \arg \min_{w \in \mathbb{R}^d} \sum_{i=1}^m (\ell(z_i^T w, y_i) + \frac{\lambda}{2} \|w\|^2) + b^T w$.
4: **return** $w^*$.

Algorithm 2 Unlearning procedure

1: **input:** Feature matrix $X \in \mathbb{R}^{n \times d}$, labels $Y \in \mathbb{R}^n$, one-step propagation matrix $P$, loss $\ell$, training set indices $T_r = \{i_1, i_2, \ldots\}$, sequence of removal requests $R_m = \{j_1, j_2, \ldots\}$, parameters $K, \epsilon, \delta, \gamma, \alpha, \lambda > 0$.
2: Compute node embedding after propagation $Z = P^K X$.
3: Training set $D = \{z_i, y_i\}_{i \in T_r}$.
4: Compute $w$ using Algorithm 1 ($D, \ell, \alpha, \lambda$).
5: Accumulated gradient residual norm $\beta = 0$.
6: for $j \in R_m$ do
7: Update the feature matrix $X'$ and propagation matrix $P'$ based on the removal.
8: Compute new node embedding after propagation $Z' = P'^K X'$.
9: if $j \in T_r$ then
10: Remove $j$ from the training indices $T_r = T_r \setminus \{j\}$.
11: end if
12: Update the training set $D' = \{z'_i, y_i\}_{i \in T_r}$.
13: Compute $\Delta = \nabla L(w, D) - \nabla L(w, D')$.
14: Compute $H = \nabla^2 L(w; D')$.
15: Update accumulated gradient residual norm $\beta = \beta + \gamma_2 \|Z'\|_{op} ||H^{-1}\Delta|| ||Z'H^{-1}\Delta||$.
16: if $\beta > \alpha \epsilon / \sqrt{2 \log(1.5 / \delta)}$ then
17: Recompute $w$ using Algorithm 1 ($D', \ell, \alpha, \lambda$), $\beta = 0$.
18: else
19: $w = w + H^{-1}\Delta$.
20: end if
21: end for
22: **return** $w$. 
A.7 Proof of Theorem 4.3

**Theorem.** Under the node feature unlearning scenario, \( \mathcal{D} = (X, Y_{T'}, A) \) and \( \mathcal{D}' = (X', Y'_{T'}, A) \). Suppose Assumption 4.2 holds. For \( Z = PKX \) and \( P = D^{-1}A \), we have

\[
\|\nabla L(w^-, D')\| = \| (H_{w_o} - H_{w^*})H_{w^*}^{-1}\Delta \| \leq \frac{\gamma_2(2c\lambda + (c\gamma_1 + \lambda\gamma_1)D_{mm})^2}{\lambda^4(m - 1)}.
\]

We need to ensure that the norm of each row of \( Z \) is bounded by 1. We state the following lemma in support of this claim.

**Lemma A.1.** Assume that \( \|e_i^T S\| \leq 1, \forall i \in [n] \). Then, \( \forall i \in [n], \forall K \geq 0, \|e_i^T PKS\| \leq 1 \), where \( P = D^{-1}A \).

**Proof.** Our proof is a nontrivial generalization and extension of the proof in Guo et al. (2020). For completeness, we outline every step of the proof. We also emphasize novel approaches used to accommodate out approximate graph unlearning scenario.

Let \( G(w) = \nabla L(w, D') \). By the Taylor theorem, \( \exists \eta \in [0, 1] \) such that

\[
G(w^-) = G(w^* + H_{w^*}^{-1}\Delta) = G(w^*) + \nabla G(w^* + \eta H_{w^*}^{-1}\Delta)H_{w^*}^{-1}\Delta
\]

\[
= (a) \quad G(w^*) + H_{w_o}H_{w^*}^{-1}\Delta = (b) \quad 0 + H_{w_o}H_{w^*}^{-1}\Delta - \Delta = (c) \quad H_{w_o}H_{w^*}^{-1}\Delta - H_{w_o}H_{w^*}^{-1}\Delta = (H_{w_o} - H_{w^*})H_{w^*}^{-1}\Delta.
\]

In (a), we wrote \( H_{w_o} \triangleq \nabla G(w^* + \eta H_{w^*}^{-1}\Delta) \), corresponding to the Hessian at \( w_o \triangleq w^* + \eta H_{w^*}^{-1}\Delta \). Equality (b) is due to our choice of \( \Delta = \nabla L(w^*, D) - \nabla L(w^*, D') \) and the fact that \( w^* \) is the minimizer of \( L(\cdot, D) \). We would like to point out that our choice of \( \Delta \) is more general than that of Guo et al. (2020). Since unlearning one node may affect the entire node embedding \( Z \), a generalization of \( \Delta \) is crucial. When \( K = 0 \) (i.e., when no graph topology is included), one recovers \( \Delta \) from Guo et al. (2020) as a special case of our model. In the latter part of the proof, we will see how the graph setting makes the analysis more intricate and complex.

By the Cauchy-Schwartz inequality, we have

\[
\|G(w^-)\| \leq \|H_{w_o} - H_{w^*}\| \|H_{w^*}^{-1}\Delta\|.
\]

Below we bound both norms on the right hand side separately. We start with the term \( \|H_{w_o} - H_{w^*}\| \).

Note that

\[
\|\nabla^2 \ell(e_i^T Z'w_o, e_i^T Y'_{T'}) - \nabla^2 \ell(e_i^T Z'w^*, e_i^T Y'_{T'})\| = \|e_i^T \nabla^2 \ell(e_i^T Z'w_o, e_i^T Y'_{T'}) - e_i^T \nabla^2 \ell(e_i^T Z'w^*, e_i^T Y'_{T'})\| (e_i^T Z')^T e_i^T Z
\]

\[
\leq \gamma_2\|e_i^T Z'w_o - e_i^T Z'w^*\|\|e_i^T Z'\| \leq \gamma_2\|e_i^T Z'\| \leq \gamma_2\|e_i^T Z'\| \leq \gamma_2\|e_i^T Z'\|^3.
\]

(10)

Here, (a) follows from the Cauchy-Schwartz inequality and the Lipschitz condition on \( \ell'' \) in Assumption 4.2. Unlike the analysis in Guo et al. (2020), we are faced with the problem of bounding the term \( \|e_i^T Z'\| \). In Guo et al. (2020) (where \( Z = X \)), a simple bound equals 1, which may be obtained via (3) in Assumption 4.2. However, in our case, due to graph propagation this norm needs more careful examination and a simple application of the Cauchy-Schwartz inequality does not suffice, as it would lead to a term \( \|X\|_{op} \), where \( \|\cdot\|_{op} \) denotes the operator norm. The simple worst case (i.e., when all rows of \( X \) are identical) leads to a meaningless bound \( O(n) \).

By leveraging Lemma A.1, we can further upper bound equation (10) according to

\[
\|\nabla^2 \ell(e_i^T Z'w_o, e_i^T Y'_{T'}) - \nabla^2 \ell(e_i^T Z'w^*, e_i^T Y'_{T'})\| \leq \gamma_2\|H_{w^*}^{-1}\Delta\|\|e_i^T Z'\| \leq \gamma_2\|H_{w^*}^{-1}\Delta\|\|e_i^T Z'\|^3
\]

(11)
where (a) follows from Lemma A.1.

As a result, we arrive at a bound for \( \|H_{w_0} - H_w\| \) of the form

\[
\|H_{w_0} - H_w\| \leq \sum_{i=1}^{m-1} \|\nabla^2 \ell(e_i^T Z w_0, e_i^T Y_{T^r}) - \nabla^2 \ell(e_i^T Z w, e_i^T Y_{T^r})\| \\
\leq \gamma_2(m - 1)\|H_{w_0}^1 \Delta\|, \tag{12}
\]

Next, we bound \( \|H_{w_0}^1 \Delta\| \). Since \( L(\cdot, D') \) is \( \lambda(m - 1) \)-strongly convex, we have \( \|H_{w_0}^1 \| \leq \frac{1}{\lambda(m - 1)} \).

For the norm \( \|\Delta\| \), we have

\[
\Delta = \nabla L(w^*, D) - \nabla L(w^*, D') \\
= \lambda w^* + \nabla \ell(e_m^T Z w^*, e_m^T Y_{T^r}) + \sum_{i=1}^{m-1} \left[ \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) - \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) \right]. \tag{13}
\]

The third term does not appear in [Guo et al. 2020], since when \( K = 0 \), \( Z = X \) and \( Z' = X' \) are identical except for the \( m \)-th row. In the approximate graph unlearning scenario, even removing one node feature can make the entire node embedding matrix \( Z \) change in every row, which creates new analytical challenges. For example, consider the case \( X = [x_1, x_2, x_3]^T \), where we have a graph with three nodes, each with a 1-dimensional feature. Consider the fully connected graph (i.e., all entries in \( P \) set to 1/3). Then, unlearning node 1 results in \( Z' = [0, x_2, x_3]^T \) for unstructured unlearning. However, \( Z' = [(x_2 + x_3)/3, (x_2 + x_3)/3, (x_2 + x_3)/3]^T \) for the case of \( L = 1 \), which is completely different from \( Z = [(x_1 + x_2 + x_3)/3, (x_1 + x_2 + x_3)/3, (x_1 + x_2 + x_3)/3]^T \) for \( L = 1 \). Hence, the analysis in [Guo et al. 2020] cannot be directly applied to graphs, as \( Z' \) changes in more than just one row compared to \( Z \) while unlearning a node feature.

By Minkowski’s triangle inequality, we only need to bound the norm of the three individual terms in order to bound the norm of \( \Delta \). For \( \|w^*\| \), since \( w^* \) is the global optimum of \( L(\cdot; D) \), we have

\[
0 = \nabla L(w^*; D) = \sum_{i=1}^{m} \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) + \lambda m w^*. \tag{14}
\]

By (1) in Assumption 4.2, we have

\[
\|w^*\| = \left\| \sum_{i=1}^{m} \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) \right\| \leq \frac{c}{\lambda}. \tag{15}
\]

Once again, by (1) in Assumption 4.2, we have

\[
\|\nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r})\| \leq c. \tag{16}
\]

A bound for the last term is established in the last step, as described below.

\[
\| \sum_{i=1}^{m-1} \left[ \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) - \nabla \ell(e_i^T Z w, e_i^T Y_{T^r}) \right] \| \\
\leq \sum_{i=1}^{m-1} \|\nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r}) - \nabla \ell(e_i^T Z w^*, e_i^T Y_{T^r})\| \\
= \sum_{i=1}^{m-1} \|\ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z) - \ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z')\|. \tag{17}
\]

Observe that

\[
\|\ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z) - \ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z')\| \\
\leq \|\ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z) - \ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z)\| + \|\ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z) - \ell'(e_i^T Z w^*, e_i^T Y_{T^r})(e_i^T Z')\| \tag{18}
\]
The first term can be bounded as
\[
\|\ell'(e_i^T Z w^*, e_i^T Y_{T_i})(e_i^T Z) - \ell'(e_i^T Z' w^*, e_i^T Y_{T_i})(e_i^T Z)\| \\
\leq \|\ell'(e_i^T Z w^*, e_i^T Y_{T_i}) - \ell'(e_i^T Z' w^*, e_i^T Y_{T_i})\| \|(e_i^T Z)\| \\
\overset{(a)}{\leq} \gamma_1 \|e_i^T Z w^* - e_i^T Z' w^*\| \|e_i^T Z\| \\
\overset{(b)}{\leq} \gamma_1 \|e_i^T (Z - e_i^T Z')\| \|w^*\| \\
\overset{(c)}{\leq} \frac{c\gamma_1}{\lambda} \|e_i^T (Z - e_i^T Z')\|. 
\]  
(19)

Here, (a) is due to (4) in Assumption 4.2 while (b) follows from Lemma A.1 and the Cauchy-Schwartz inequality. Inequality (c) is a consequence of the bound for \|w\| that we previously derived.

The second term can be bounded as
\[
\|\ell'(e_i^T Z w^*, e_i^T Y_{T_i})(e_i^T Z) - \ell'(e_i^T Z' w^*, e_i^T Y_{T_i})(e_i^T Z')\| \\
\leq \|\ell'(e_i^T Z' w^*, e_i^T Y_{T_i})\| \|(e_i^T Z) - (e_i^T Z')\| \\
\overset{(a)}{\leq} c_1 \|(e_i^T Z) - (e_i^T Z')\|. 
\]  
(20)

For the inequality in (a), we used (5) from Assumption 4.2. Put together, we have
\[
\| \sum_{i=1}^{m-1} \left[ \nabla \ell(e_i^T Z w^*, e_i^T Y_{T_i}) - \nabla \ell(e_i^T Z' w^*, e_i^T Y_{T_i}) \right] \| \\
\leq \sum_{i=1}^{m-1} \left[ \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \|e_i^T P^K (X - X')\| \right. \\
= \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D}(X - X')\| \\
\overset{(a)}{=} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D} m e_m X\| \\
\overset{(b)}{\leq} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D} m e_m X\| \\
\overset{(c)}{=} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D} m e_m X\| \\
= \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D} m e_m X\| \\
\overset{(d)}{=} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \|e_i^T P^K \hat{D}^{-1} \hat{D} m e_m X\| \\
\overset{(e)}{=\leq} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \|1^T P^K \hat{D}^{-1} e_m D_{mm}\| \\
\overset{(f)}{=} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \|1^T \hat{D}^{-1} p \hat{D}_{mm}\| \\
\overset{(g)}{=} \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \hat{D}_{mm}. 
\]  
(21)
Theorem can be proved as follows. From previous proof we have
\[ P^K \tilde{D}^{-1} = (\tilde{D}^{-1} \tilde{A})^K \tilde{D}^{-1} = \tilde{D}^{-1}(\tilde{A}\tilde{D}^{-1})^K, \tag{22} \]
and \( \tilde{A}\tilde{D}^{-1} \) is a left stochastic matrix. Inequality (g) is a consequence of the observation that the maximum entry in \( \tilde{D}^{-1} \) is at most 1 and that the latter is a diagonal matrix. Hence, \( 1^T \tilde{D}^{-1} p \leq 1^T p \). Also, \( 1^T p = 1 \) by the definition of the probability vector.

Combining the bounds, we obtain
\[ \| \Delta \| \leq c + c + \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \tilde{D}_{mm} = \frac{2c\lambda + (c\gamma_1 + \lambda c_1) \tilde{D}_{mm}}{\lambda}. \tag{23} \]
Including the bound on \( \| H_w^{-1} \| \) and equation (12) we then obtain
\[ \| G(w^-) \| \leq \gamma_2(m - 1)\| H_w^{-1} \Delta \|^2 \leq \gamma_2(m - 1) \left( \frac{2c\lambda + (c\gamma_1 + \lambda c_1) \tilde{D}_{mm}}{\lambda(m - 1)} \right)^2. \]
\[ = \frac{\gamma_2(2c\lambda + (c\gamma_1 + \lambda c_1) \tilde{D}_{mm})^2}{\lambda^2(m - 1)}. \tag{24} \]
This completes the proof. \( \square \)

A.8 Proof of Theorem 4.4

**Theorem.** For the edge unlearning case, we have \( D = (X, Y_{T'}, P) \) and \( D' = (X, Y_{T'}, P') \). If \( P = \tilde{D}^{-1} \tilde{A} \) and \( \tilde{Z} = P^K X \), then we have
\[ \| \nabla\ell(w^-, D') \| = \| (H_{w_n} - H_{w'}) H_{w^*}^{-1} \Delta \| \leq \frac{16\gamma_2 K^2 (c\gamma_1 + c_1\lambda)^2}{\lambda^4 m}. \tag{25} \]
Similar to what holds for the node feature unlearning case, Theorem 4.4 still holds when neither of the two end nodes of the removed edge belongs to \( T' \). Since \( P' \) is a right stochastic matrix, Lemma A.1 still applies. Thus, we only need to describe how to bound \( \| \Delta \| \). Following an approach similar to the previously described one, we have \( \| \Delta \| \leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{k=1}^m \sum_{j=1}^n \| e^T_k (P^K - P'K^k) e_j \| \). We also need the following technical lemmas.

**Lemma A.2.** For both edge and node unlearning, we have \( | e^T_i | [P^K - (P')^K] e_j | \leq \sum_{k=1}^K e^T_i (P')^{k-1} | P - P' | P^{k-1} e_j, \forall i, j \in [n], K \geq 1. \)

**Lemma A.3.** For edge unlearning, we have \( 1^T P^{k-1} | P - P' | P^{k-1} 1 \leq 4, \forall k \in [K] \).

Combining the two lemmas and after some algebraic manipulation, we arrive at the desired result. It is not hard to see that \( | P - P' | \) has only two nonzero rows, which correspond to the unlearned edge. One can again construct a left stochastic matrix \( \tilde{A}/\tilde{D}^{-1} \) and a right stochastic matrix \( P \) which lead to the result of Lemma A.3.

**Proof.** The theorem can be proved as follows. From previous proof we have
\[ \| G(w^-) \| \leq \| H_{w_n} - H_{w'} \| \| H_{w^*}^{-1} \| \| \Delta \| \leq \gamma_2 \frac{\| \Delta \|^2}{\lambda^2 m}. \tag{26} \]
Since the first term \( \| H_{w_n} - H_{w'} \| \) only involved the updated dataset, the upper bound for this term proved for node feature unlearning still holds. The term \( \| H_{w^*}^{-1} \| \) can again be bounded using the fact
that $L(\cdot, D')$ is $\lambda m$-strongly convex. The main difference between node feature and edge unlearning lies in the bound for $\Delta$. By definition,

$$\Delta = \nabla L(w^*, D) - \nabla L(w^*, D')$$

$$= \sum_{i=1}^{m} \left[ \nabla \ell(e_i^T Z w^*, e_i^T Y_{T_i}) - \nabla \ell(e_i^T Z' w^*, e_i^T Y_{T_i}) \right], \text{ and}$$

$$\|\Delta\| \leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \| (Z - Z')^T e_i \|$$

$$= \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \| (P^K X - P'^K X)^T e_i \|$$

$$= \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \| e_i^T (P^K - P'^K) X \|$$

$$= \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \| e_i^T (P^K - P'^K) \sum_{j=1}^{n} e_j e_j^T X \|$$

$$\leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \sum_{j=1}^{n} \| e_i^T (P^K - P'^K) e_j e_j^T X \|$$

$$\leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \sum_{j=1}^{n} \| e_i^T (P^K - P'^K) e_j \| \| e_j^T X \|$$

$$\leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \sum_{j=1}^{n} \| e_i^T (P^K - P'^K) e_j \|$$

(27)

By Lemma A.2, we have

$$\left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \sum_{j=1}^{n} \| e_i^T (P^K - P'^K) e_j \|$$

$$\leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{K} e_i^T P'^{k-1} |P - P'| |P^{K-k} e_j|$$

$$\leq \left( \frac{c\gamma_1}{\lambda} + c_1 \right) \sum_{k=1}^{K} 1^T P'^{k-1} |P - P'| |P^{K-k} 1.$$
Lemma A.4. For node unlearning and $\forall k \in [K]$ and $\forall l \in [n]$, $1^T(P')^{k-1}(\tilde{D}')^{-1}e_l \leq 1$.

Lemma A.5. For node unlearning and $\forall k \in [K]$. $\sum_{i=1}^{n} e_i^T \tilde{D}' |P - P'| P'^K - 1 \leq 2\tilde{D}_{mm} - 1$.

These two lemmas give rise to the term $K(2\tilde{D}_{mm} - 1)$ in the bound of Theorem 4.5 and the rest of the analysis is similar to that of the previous cases. Lemma A.3 is rather technical, and relies on the following proposition that exploits the structure of $|P - P'|$.

Proposition A.6. For node unlearning and $\forall i, j \neq m$, $e_i^T |P - P'| e_j = e_i^T (P' - P) e_j$. For $i = m$ or $j = m$, $e_i^T |P - P'| e_j = e_i^T Pe_j$.

Proof. The proof is similar to the proof of Theorem 4.3 although several parts need modifications. First, the result of Lemma A.1 needs to be replaced by the following claim.

Lemma A.7. Assume that $\|e_i^T S\| \leq 1$, $\forall i \neq m$ and that $e_m^T S = 0^T$. Then $\forall i \in [n]$, $K \geq 0$, we have $\|e_i^T (P')^K S\| \leq 1$, where $P = \tilde{D}^{-1} \tilde{A}$ and $P' = (\tilde{D}')^{-1} \tilde{A}'$.

Next we have to modify the proof regarding the bound of $\|\Delta\|$. Following a proof similar to that of Theorem 4.3 we have

$$\|\Delta\| \leq 2c + \left(\frac{c_1}{\lambda} + c_2\right) \sum_{i=1}^{m-1} \|(Z - Z')^T e_i\|.$$ (30)

Plugging in the expressions for $Z$ and $Z'$ leads to

$$\sum_{i=1}^{m-1} \|(Z - Z')^T e_i\| = \sum_{i=1}^{m-1} \|(P^K X - (P')^K X')^T e_i\|$$

$$= \sum_{i=1}^{m-1} \|(P^K X - (P')^K X')^T e_i\| = \sum_{i=1}^{m-1} \|(P^K - (P')^K) X^T e_i\|$$

$$\leq \sum_{i=1}^{m-1} \sum_{j=1}^{n} \|e_i^T [P^K - (P')^K] e_j^T X\|$$

$$\leq \sum_{i=1}^{m-1} \sum_{j=1}^{n} \|e_i^T [P^K - (P')^K] e_j^T X\|$$

$$\leq \sum_{i=1}^{m-1} \sum_{j=1}^{n} \|e_i^T [P^K - (P')^K] e_j^T e_j^T X\|$$

$$\leq \sum_{i=1}^{m-1} \sum_{j=1}^{n} \|e_i^T [P^K - (P')^K] e_j^T e_j^T X\|.$$ (31)

The equality (a) is due to the fact that $(P')^K X = (P')^K X'$, as the $m^{th}$ row and column of $(P')^K$ are all-zeros. Thus, changing the last row of $X'$ makes no difference of $(P')^K X'$. Equation (b) is a consequence of the fact that $I = \sum_{j=1}^{n} e_j e_j^T$. Inequality (c) follows from Minkowski’s inequality, while (d) follows from the Cauchy-Schwartz inequality. Inequality (e) holds based on (3) in Assumption 4.2.
By Lemma A.2, we can proceed with our analysis as follows:

$$\sum_{i=1}^{m-1} \|(Z - Z')^T e_i\|$$

$$\leq \sum_{i=1}^{m-1} \sum_{j=1}^n \| e_i^T (P^K - (P')^K) e_j\|$$

$$(a) \leq \sum_{i=1}^{m-1} \sum_{j=1}^n \sum_{k=1}^K e_i^T (P')^{k-1} |P - P'| P^{K-k} e_j$$

$$\leq \sum_{k=1}^K 1^T (P')^{k-1} |P - P'| P^{K-k} 1, \quad (32)$$

where (a) is due to Lemma A.2 and the fact that $e_i^T P e_j$ is a scalar, equal to the $i^{th}$ row $j^{th}$ column of the matrix $P$.

Next, we bound each term $1^T (P')^{k-1} |P - P'| P^{K-k} 1$ separately. For $k \in [K]$, we have

$$1^T (P')^{k-1} |P - P'| P^{K-k} 1$$

$$= 1^T (P')^{k-1} (\hat{D}')^{-1} \hat{D}' |P - P'| P^{K-k} 1$$

$$= 1^T (P')^{k-1} (\hat{D}')^{-1} \sum_{l=1}^n e_l e_l^T \hat{D}' |P - P'| P^{K-k} 1$$

$$= \sum_{l=1}^n \left( 1^T (P')^{k-1} (\hat{D}')^{-1} e_l \right) \left( e_l^T \hat{D}' |P - P'| P^{K-k} 1 \right). \quad (33)$$

Note that for each index $l$, the corresponding term in the sum is just a product of two scalars. Let first analyze $1^T (P')^{k-1} (\hat{D}')^{-1} e_l$. This term can be bounded as

$$1^T (P')^{k-1} |P - P'| P^{K-k} 1$$

$$= \sum_{j=1}^n \left( 1^T (P')^{k-1} (\hat{D}')^{-1} e_j \right) \left( e_j^T \hat{D}' |P - P'| P^{K-k} 1 \right)$$

$$(a) \leq \sum_{j=1}^n e_j^T \hat{D}' |P - P'| P^{K-k} 1. \quad (34)$$

where (a) follows from Lemma A.4.

We now turn our attention to the term $e_l^T \hat{D}' |P - P'| P^{K-k} 1$, which can be bounded according to Lemma A.5 as follows

$$1^T (P')^{k-1} |P - P'| P^{K-k} 1$$

$$\leq \sum_{l=1}^n e_l^T \hat{D}' |P - P'| P^{K-k} 1$$

$$\leq 2 \hat{D}_{mm} - 1. \quad (35)$$
Using these two bounds in equation 32 gives
\[
\sum_{i=1}^{m-1} \|(Z - Z')^T e_i\|
\leq \sum_{i=1}^{m-1} \sum_{j=1}^{n} \|e_i^T P^K - (P')^K e_j\|
\leq \sum_{k=1}^{K} 1^T (P')^{k-1} |P - P'| P^{K-k} 1
\leq \sum_{k=1}^{K} (2\tilde{D}_{mm} + 1) = K(2\tilde{D}_{mm} - 1).
\tag{36}
\]

Using this bound in the expression for \(\|\Delta\|\) we obtain
\[
\|\Delta\| \leq 2c + \frac{(c\gamma_1 + c_1)}{\lambda} \sum_{i=1}^{m-1} \|(Z - Z')^T e_i\|
\leq 2c + \frac{(c\gamma_1 + c_1)}{\lambda} K (2\tilde{D}_{mm} - 1)
\Rightarrow \|G(w^-)\| \leq \|H_{w_0} - H_{w^*}\| \|H_{w^*}^{-1}\Delta\|
\leq \gamma_2 (m - 1) \|H_{w^*}^{-1}\Delta\|^2
\leq \gamma_2 (m - 1) \left( \frac{2c + (c\gamma_1 + c_1) K (2\tilde{D}_{mm} - 1)}{\lambda (m - 1)} \right)^2
= \frac{\gamma_2 (2c\lambda + K (c\gamma_1 + c_1) \lambda) (2\tilde{D}_{mm} - 1)^2}{\lambda^3 (m - 1)}.
\tag{37}
\]

This completes the proof.

A.10 Proof of Theorem 4.6

**Theorem.** In the node feature unlearning scenario, we are given \(\mathcal{D} = (X, Y_{T_v}, A)\) and \(\mathcal{D}' = (X', Y'_{T_v}, A')\). Suppose that Assumption 4.2 holds. For \(Z = \frac{1}{K+1} [X, PX, P^2X, \ldots, P^KX]\) and \(P = \tilde{D}^{-1} A\), we have
\[
\|\nabla L(w^-, D')\| = \|(H_{w_0} - H_{w^*})H_{w^*}^{-1}\Delta\| \leq \frac{\gamma_2 (2c\lambda + (c\gamma_1 + c_1) \tilde{D}_{mm})^2}{\lambda^4 (m - 1)}. \tag{38}
\]

**Proof.** The proof is almost identical to the proof of Theorem 4.3. We only need to bound the norms of the terms in \(Z\). We start by modifying Lemma A.1 for the GPR case.

**Lemma A.8.** Assume that \(\|e_i^T S\| \leq 1, \forall i \in [n]\). Then for \(\forall i \in [n], K > 0, we have \(\|\frac{1}{\sqrt{K+1}} e_i^T [S, PS, P^2S, \ldots, P^K S]\| \leq 1\), where \(P = \tilde{D}^{-1} A\).

Another part of the proof that needs to be changed is to establish a bound on
\[
\sum_{i=1}^{m-1} \left[ \nabla \ell(e_i^T Z w^*, e_i^T Y_{T_v}) - \nabla \ell(e_i^T Z' w^*, e_i^T Y_{T_v}) \right]. \tag{39}
\]
Following a proof similar to that of Theorem \ref{thm:4.3}, we have

\[
\| \sum_{i=1}^{m-1} [\nabla \ell(e_i^T Z w^*, e_i^T Y_{T_i}) - \nabla \ell(e_i^T Z' w^*, e_i^T Y_{T_i})] \| \\
\leq \sum_{i=1}^{m-1} \left( \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \| (e_i^T Z)^T - (e_i^T Z')^T \| \right) \\
= \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \| (Z - Z')^T e_i \| \\
= \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \theta_i \| (X - X')^T e_i \| \\
= \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \left( \frac{1}{K+1} \right) \left[ e_m e_m^T X, e_m e_m^T X, \cdots, e_m e_m^T X \right]^T e_i \\
= \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \left[ e_i^T e_m^T X, e_i^T e_m^T X, \cdots, e_i^T e_m^T X \right]^T \\
\leq \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \frac{1}{K+1} \left[ e_i^T e_m^T, e_i^T e_m^T, \cdots, e_i^T e_m^T \right]^T \\
\leq \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \sum_{i=1}^{m-1} \frac{1}{K+1} \sum_{k=1}^{K} e_i^T P^k e_m \\
\leq \frac{c_{\gamma_1}}{K+1} \sum_{k=1}^{K} P^k e_m \\
\leq \frac{c_{\gamma_1}}{K+1} \sum_{k=1}^{K} P^k e_m \\
\leq \frac{c_{\gamma_1}}{K+1} \sum_{k=1}^{K} P^k e_m \\
\leq \frac{c_{\gamma_1}}{K+1} \sum_{k=1}^{K} P^k e_m \\
\leq \left( \frac{c_{\gamma_1}}{\lambda} + c_1 \right) \bar{D}_{mm}. \qquad (40)
\]

where (a) is due to the fact that the $\ell_1$ norm is an upper bound for the $\ell_2$ norm. Also note that $e_i^T e_m = 0, \forall i \neq m$. In (b), $\forall k \in [K], P^{(k)}$ are probability vectors. This completes the proof. \qed

**Remark.** Note that the GPR extension for the edge and node unlearning cases can be derived through a similar analysis. One can also see that the key step is inequality (a), which still holds for the edge and node unlearning cases. The results are similar to Theorem \ref{thm:4.4} and Theorem \ref{thm:4.5}, except that the definition of $Z$ is replaced by one corresponding to the GPR case, as in Theorem \ref{thm:4.6}.

**A.11 Proof of Lemma [A.1]**

**Lemma.** Assume that $\| e_i^T S \| \leq 1, \forall i \in [n]$. Then $\forall i \in [n], K \geq 0$, we have $\| e_i^T P^K S \| \leq 1$, where $P = \bar{D}^{-1} A$.

**Proof.** We prove this lemma by induction. Let $Z^{(k)} = P^K S$. For the base case $k = 0$ it is true by assumption that $\| e_i^T S \| \leq 1 \forall i \in [n]$. Assume next that the claim is true for the case $k = K - 1$. 


Then we have
\[
\|e_i^T P^K S\| = \|e_i^T PZ^{(K-1)}\| = \left\| \frac{1}{D_{ii}} \sum_{j:A_{ij}=1} e_j^T Z^{(K-1)} \right\| \leq \frac{1}{\sqrt{D_{ii}}} \sum_{j:A_{ij}=1} \| e_j^T Z^{(K-1)} \| \leq \frac{1}{D_{ii}} \sum_{j:A_{ij}=1} 1 = \frac{1}{D_{ii}} \times \bar{D}_{ii} = 1,
\]
where (a) is based on the induction hypothesis for \( k = K - 1 \).

**Remark:** Note that if we choose another propagation matrix \( P \) compared to the one used in the SGC analysis, the above expression for \( K = 1 \) becomes
\[
\|e_i^T PS\| = \left\| \frac{1}{\sqrt{D_{ii}}} \sum_{j:A_{ij}=1} e_j^T S \right\| \leq \frac{1}{\sqrt{D_{ii}}} \sum_{j:A_{ij}=1} \| e_j^T S \| \leq \frac{1}{\sqrt{D_{ii}}} \sum_{j:A_{ij}=1} 1 = \sqrt{\bar{D}_{ii}}.
\]
We cannot easily simplify the sum \( \sum_{j:A_{ij}=1} \frac{1}{\sqrt{D_{jj}}} \). One way to approach the problem is to simply use the fact that the degree of a node is at least 1 and can thus be further upper bounded by \( \bar{D}_{ii} \). This leads to the bound
\[
\|e_i^T PS\| \leq \sqrt{\bar{D}_{ii}}.
\]
Obviously, this bound is worse than the one in Lemma A.1 even when \( K = 1 \). For general \( K \), there will be an additional exponent \( K/2 \) for the maximal degree, which is undesirable. Nevertheless, our bound is tight since for the worst case of a star graph with a center at node \( i \), so that \( D_{jj} = 2 \) for all \( j \neq i \). The same argument applies for other degree normalizations. Thus it is critical to choose \( P = \bar{D}^{-1} \bar{A} \) to obtain the desired bound in Lemma A.1.

**A.12 Proof of Lemma A.2**

**Lemma.** For either the edge or node unlearning case, and \( \forall i, j \in [n], K \geq 1 \), we have
\[
|e_i^T (P^K - (P')^K) e_j| \leq \sum_{k=1}^{K} e_i^T (P')^{k-1} |P - P'| P^{K-k} e_j.
\]

**Proof.** The proof consists of two parts. We first show that
\[
P^K - (P')^K = \sum_{k=1}^{K} (P')^{k-1} (P - P') P^{K-k}.
\]
Then we proceed to analyze the absolute values of all terms in the sum.

The proof of the first part follows from a telescoping property for the sum,
\[
\sum_{k=1}^{K} (P')^{k-1} (P - P') P^{K-k} = \sum_{k=1}^{K} (P')^{k-1} P^{K-k+1} - (P')^{k} P^{K-k} = (P')^0 P^K - (P') P^{K-1} + (P')^1 P^{K-2} - (P')^2 P^{K-2} + \ldots + (P')^{K-1} P^1 - (P')^K P^0 = P^K - (P')^K.
\]
Next, note that both \( P' \) and \( P \) are nonnegative matrices, and the same is true of their \( k^{th} \) powers, \( k \geq 2 \). Thus,
\[
|e_i^T (P^K - (P')^K) e_j| = \left| \sum_{k=1}^{K} e_i^T (P')^{k-1} (P - P') P^{K-k} e_j \right| \leq \sum_{k=1}^{K} e_i^T (P')^{k-1} |P - P'| P^{K-k} e_j.
\]
This completes the proof.
A.13 Proof of Lemma \[A.3\]

**Lemma.** For the edge unlearning scenario, and \( \forall k \in [K] \), we have
\[
1^T \mathbf{P}^{k-1} (\mathbf{P} - \mathbf{P}') \mathbf{P}^{K-k} \mathbf{1} \leq 4. \tag{47}
\]

**Proof.** Let us start by analyzing the matrix \(|\mathbf{P} - \mathbf{P}'| = |\mathbf{D}^{-1} \mathbf{A} - \mathbf{D}^{-1} \mathbf{A}'|\). Note that all its rows are zeros except for the 1st and \( m \)th row. The first row of the matrix equals
\[
e_1^T (\mathbf{P} - \mathbf{P}') = \left[ \left( \frac{1}{d_1} - \frac{1}{d_1} \right) \tilde{A}_{11}, \ldots, \left( \frac{1}{d_1} - \frac{1}{d_1} \right) \tilde{A}_{1(m-1)}, \frac{1}{d_1} \left( \frac{1}{d_1} - \frac{1}{d_1} \right) \tilde{A}_{1(m+1)}, \ldots \right]
\]
\[
= \left[ \left( \frac{1}{d_1(d_1-1)} \right) \tilde{A}_{11}, \ldots, \left( \frac{1}{d_1(d_1-1)} \right) \tilde{A}_{1(m-1)}, \frac{1}{d_1(d_1-1)} \left( \frac{1}{d_1(d_1-1)} \right) \tilde{A}_{1(m+1)}, \ldots \right]
\]
\[
+ \frac{d_1 - 2}{d_1(d_1-1)} e_m^T = e_m^T \mathbf{D}^{-1} \mathbf{D}^{-1} \mathbf{A} + \frac{d_1 - 2}{d_1(d_1-1)} e_m^T,
\] \tag{48}
where the last equality holds since \( \tilde{A}_{1m} = 1 \). Similar arguments apply for the \( m \)th row, for which we have
\[
e_m^T |\mathbf{P} - \mathbf{P}'| = e_m^T \mathbf{D}^{-1} \mathbf{D}^{-1} \mathbf{A} + \frac{d_m - 2}{d_m(d_m-1)} e_1^T. \tag{49}
\]

For a fixed \( k \in [K] \),
\[
1^T \mathbf{P}^{k-1} |\mathbf{P} - \mathbf{P}'| \mathbf{P}^{K-k} \mathbf{1} = 1^T \mathbf{P}^{k-1} e_1 e_1^T \mathbf{D}^{-1} \mathbf{D}^{-1} \mathbf{A} \mathbf{P}^{K-k} \mathbf{1}
\]
\[
= 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 e_1^T \mathbf{A} \mathbf{P}^{K-k} \mathbf{1}
\]
\[
= 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 e_1^T \mathbf{P}^{K-k+1} \mathbf{1}
\] \tag{50}
We analyze these four terms separately. For the first term, we have
\[
1^T \mathbf{P}^{k-1} e_1 e_1^T \mathbf{D}^{-1} \mathbf{D}^{-1} \mathbf{A} \mathbf{P}^{K-k} \mathbf{1}
\]
\[
= 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 e_1^T \mathbf{A} \mathbf{P}^{K-k} \mathbf{1}
\]
\[
= 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 e_1^T \mathbf{P}^{K-k+1} \mathbf{1}
\]
\tag{51}
By the same argument as used in the proof for node feature unlearning, \( 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 = 1^T \mathbf{D}^{-1} \mathbf{p} \leq 1 \), for some probability vector \( \mathbf{p} \). Also, \( e_1^T \mathbf{P}^{K-k+1} \mathbf{1} \leq 1 \), which holds due to the fact that \( \mathbf{P} \) is a right-stochastic matrix. We have hence shown that the first term in equation \[51\] is bounded by 1. For the second term, note that \( \frac{d_m - 2}{d_m(d_m-1)} \leq \frac{1}{d_m(d_m-1)} \). Hence,
\[
1^T \mathbf{P}^{k-1} \frac{d_1 - 2}{d_1(d_1-1)} e_1 e_m^T \mathbf{P}^{K-k} \mathbf{1}
\]
\[
\leq 1^T \mathbf{P}^{k-1} \frac{1}{d_1(d_1-1)} e_1 e_m^T \mathbf{P}^{K-k} \mathbf{1}
\]
\[
= 1^T \mathbf{P}^{k-1} \mathbf{D}^{-1} e_1 e_m^T \mathbf{P}^{K-k} \mathbf{1} \leq 1, \tag{52}
\]
where the final inequality follows the same argument as the one used for bounding the first term. For the third and fourth term, the analysis is similar to these two cases and both terms can be shown to be bounded by 1. Hence, we have
\[
1^T \mathbf{P}^{k-1} |\mathbf{P} - \mathbf{P}'| \mathbf{P}^{K-k} \mathbf{1} \leq 4. \tag{53}
\]
This completes the proof. \(\square\)
A.14 Proof of Lemma A.4

Lemma. For all \( k \in [K] \) and \( l \in [n] \),
\[
1^T(P')^{k-1}(\tilde{D}')^{-1}e_l \leq 1.
\]

Proof. For \( k = 1 \), the claim is obviously true for all \( l \in [n] \), as the largest entry in \( \tilde{D}^{-1} \) is upper bounded by 1. For \( k \geq 2 \) and \( l \neq m \) we have
\[
1^T(P')^{k-1}(\tilde{D}')^{-1}e_l = 1^T((\tilde{D}')^{-1}\tilde{A}')^{k-1}(\tilde{D}')^{-1}e_l \\
= 1^T(\tilde{D}')^{-1}(\tilde{A}'(\tilde{D}')^{-1})^{k-1}e_l \\
\overset{(a)}{=} 1^T(\tilde{D}')^{-1}p \leq 1.
\]

In (a), \( p \) stands for a probability vector and the result follows since \( \tilde{A}'(\tilde{D}')^{-1} \) is a left-stochastic matrix if one ignores the node \( m \). For \( l = m \), it is easy to see that \( \tilde{A}' \tilde{D} e_m = 0 \) by the fact that the \( m^{th} \) row and column of \( \tilde{A}' \) are all-zeros. This completes the proof. \( \square \)

A.15 Proof of Lemma A.5

Lemma. For node unlearning, and \( \forall k \in [K] \), \( \sum_{l=1}^n e_l^T \tilde{D}' |P - P'| \sum_{r=1}^n e_r e_r^T \sum_{l=1}^n = 2 \tilde{D}_{mm} - 1 \).

Proof. First, note that
\[
\sum_{l=1}^n e_l^T \tilde{D}' |P - P'| \sum_{r=1}^n e_r e_r^T \sum_{l=1}^n = \sum_{l=1}^n e_l^T \tilde{D}' |P - P'| e_l e_l^T. \tag{54}
\]

Then for \( i, j \neq m \), by Proposition A.6 we have
\[
e_l^T \tilde{D}' |P - P'| e_l e_l^T \\
\overset{(a)}{=} e_l^T \tilde{D}' (P' - P) e_l e_l^T \\
= e_l^T (\tilde{A}' - \tilde{D}' \tilde{A}^{-1} \tilde{A}) e_l e_l^T \\
= \left( \tilde{A}'_{lr} - \frac{\tilde{D}_{lr}'}{\tilde{D}_{ll}'} \tilde{A}_{lr} \right) e_l e_l^T \\
\overset{(b)}{=} \left( \tilde{A}'_{lr} - \frac{\tilde{D}_{lr}'}{\tilde{D}_{ll}'} \tilde{A}_{lr} \right) e_l e_l^T. \tag{55}
\]

We used Proposition A.6 in (a) since \( e_l^T \tilde{D}' = \tilde{D}_{ll}' e_l^T \). The equality (b) is due to the fact that for \( i, j \neq m \), \( \tilde{A}'_{lr} = \tilde{A}_{lr} \). Recall that \( \tilde{A}' \) and \( \tilde{A} \) only differ in the \( m^{th} \) row and column.

We consider next the only two possible scenarios, (1) \( l \) is a neighbor of \( m \); (2) \( l \) is not a neighbor of \( m \). For (1), we know that \( \tilde{D}'_{ll} = \tilde{D}_{ll} - 1 \geq 1 \). This leads to
\[
\tilde{A}'_{lr} - \frac{\tilde{D}_{lr}'}{\tilde{D}_{ll}'} \tilde{A}_{lr} = \tilde{A}_{lr} \left( 1 - \frac{\tilde{D}_{lr}'}{\tilde{D}_{ll}'} \right) = \tilde{A}_{lr} \left( 1 - \frac{\tilde{D}_{ll} - 1}{\tilde{D}_{ll}} \right) = \tilde{A}_{lr} \frac{\tilde{D}_{ll} - 1}{\tilde{D}_{ll}}. \tag{58}
\]

For (2), we know that \( \tilde{D}'_{ll} = \tilde{D}_{rr} \). Thus, \( \tilde{A}'_{lr} - \frac{\tilde{D}_{lr}'}{\tilde{D}_{ll}'} \tilde{A}_{lr} = 0 \).

Next, we consider the case \( l \neq m, r = m \). Again, by Proposition A.6 we have
\[
e_l^T \tilde{D}' |P - P'| e_m e_m^T = e_l^T \tilde{D}' \tilde{P} e_m e_m^T = \frac{\tilde{D}_{ll}'}{\tilde{D}_{ll}} \tilde{A}'_{lm} e_m e_m^T. \tag{59}
\]

Now, for case (1), we have \( \tilde{A}'_{lm} = 1 \) and \( \tilde{D}_{ll}' = \tilde{D}_{ll} - 1 \geq 1 \). This leads to
\[
e_l^T \tilde{D}' |P - P'| e_m e_m^T = \frac{\tilde{D}_{ll}'}{\tilde{D}_{ll}} \tilde{A}'_{lm} e_m e_m^T = \frac{\tilde{D}_{ll} - 1}{\tilde{D}_{ll}} e_m e_m^T. \tag{60}
\]
For case (2), we clearly have $e_j^T \tilde{D}' | P - P' | e_m e_m^T = 0$ as $\hat{A}_{im} = 0$.

Hence, for each $j \neq m$ and under the setting in case (1), $e_j^T \tilde{D}' | P - P' |$ equals the row vector

$$\begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} & \cdots & \hat{A}_{l(m-1)} & 0 & \hat{A}_{l(m+1)} & \cdots \\ D_{ll} & D_{ll} & \cdots & D_{ll} & 0 & D_{ll} & \cdots \\ \end{bmatrix} + \begin{bmatrix} 0, \ldots, 0, \frac{D_{ll} - 1}{D_{ll}}, 0, \ldots \\ \end{bmatrix},$$

where the $n^{th}$ entry of the first row vector equals 0 and the second row vector is all-zeros except for the $m^{th}$ entry. Note that the first row vector times $\frac{D_{ll}}{D_{ll} - 1} > 1$ is a probability vector. Hence, by the property of $P^{K-k}$ being a right-stochastic matrix, we have

$$\begin{bmatrix} \hat{A}_{11} & \hat{A}_{12} & \cdots & \hat{A}_{l(m-1)} & 0 & \hat{A}_{l(m+1)} & \cdots \\ D_{ll} & D_{ll} & \cdots & D_{ll} & 0 & D_{ll} & \cdots \\ \end{bmatrix} P^{K-k} 1 \leq 1. \quad (62)$$

Since $\frac{D_{ll}}{D_{ll} - 1} < 1$, we also have

$$\begin{bmatrix} 0, \ldots, 0, \frac{D_{jj} - 1}{D_{jj}}, 0, \ldots \\ \end{bmatrix} P^{K-k} 1 \leq 1. \quad (63)$$

Together, this shows that for each $j \neq m$ and for the case (1), one has

$$e_j^T \tilde{D}' | P - P' | P^{K-k} 1 \leq 2. \quad (64)$$

For case (2), note that $e_j^T \tilde{D}' | P - P' |$ is an all-zero row vector. Note also that, excluding self-loops, there are at most $D_{mm} - 1$ neighbors $l$ of $m$ (case (1)). Thus,

$$\sum_{l \neq m} e_l^T \tilde{D}' | P - P' | P^{K-k} 1 \leq 2D_{mm} - 2. \quad (65)$$

To conclude the proof, we analyze the term $l = m$. For any $i \in [n]$, by Proposition A.6, we have

$$e_m^T \tilde{D}' | P - P' | e_i e_i^T = e_m^T D' Pe_i e_i^T = \frac{D_{mm}}{D_{mm}} \hat{A}_{mr} e_r e_r^T = \frac{D_{mm}}{D_{mm}} e_m D^{-1} \hat{A} e_r e_r^T = e_m^T Pe_r e_r^T \quad (66)$$

where (a) holds by definition, and since $\tilde{D}'_{mm} = 1$. Thus,

$$e_m^T \tilde{D}' | P - P' | P^{K-k} 1 = e_m^T P P^{K-k} P^{T} 1 = e_m^T \hat{A} e_r e_r^T = e_m^T Pe_r e_r^T \quad (67)$$

for some probability vector $p$. We have hence shown that for any $k \in [K]$,

$$\sum_{j=1}^n e_j^T \tilde{D}' | P - P' | P^{K-k} 1 \leq 2D_{mm} - 2 + 1 = 2D_{mm} - 1. \quad (68)$$

This completes the proof. \(\square\)

A.16 PROOF OF LEMMA A.7

**Lemma.** Assume that $\|e_i^T S\| \leq 1$, $\forall i \neq m$ and that $e_m^T S = 0^T$. Then $\forall i \in [n]$, $K \geq 0$, we have $\|e_i^T (P')^K S\| \leq 1$, where $P = \tilde{D}^{-1} \hat{A}$ and $P' = (\tilde{D}')^{-1} \hat{A}'$.

**Proof.** The proof is similar to the proof of Lemma A.1 and based on induction. The base case $k = 0$ is obviously true by assumption. Now, assume that the claim is true for $k = K - 1$ and let
\[ Z^{(K-1)} = (P')^{(K-1)}S. \] Then, \( \forall i \neq m, \)
\[
\|e_i^T (P')^K S\| = \|e_i^T P' Z^{(K-1)}\| = \| \frac{1}{D_{ii}} \sum_{j: \hat{A}_{ij} = 1} e_j^T Z^{(K-1)} \| 
\leq \frac{1}{D_{ii}} \sum_{j: \hat{A}_{ij} = 1} \|e_j^T Z^{(K-1)}\| 
\leq \frac{1}{D_{ii}} \sum_{j: \hat{A}_{ij} = 1} 1 = \frac{1}{D_{ii}} \hat{d}_{ii}' = 1. \tag{69}
\]

Here, (a) is due to our hypothesis for \( k = K - 1 \). For \( i = m \), note that \( \hat{A}_{mj} = 0, \forall j \in [n] \). Thus, \( \|e_i^T (P')^K S\| = 0 \leq 1 \). This completes the proof. \( \square \)

A.17 Proof of Lemma A.8

**Lemma.** Assume that \( \|e_i^T S\| \leq 1, \forall i \in [n] \). Then, \( \forall i \in [n], K \geq 0 \), we have \( \|\frac{1}{\sqrt{K+1}} e_i^T [S, PS, P^2 S, \ldots, P^K S]\| \leq 1 \), where \( P = \hat{D}^{-1} \hat{A} \).

**Proof.** By Lemma A.1 we have \( \|e_i^T P^k S\| \leq 1, \forall k \in [K] \). Thus,
\[
\|\frac{1}{\sqrt{K+1}} e_i^T [S, PS, P^2 S, \ldots, P^K S]\|^2 = \frac{1}{K+1} \left( \sum_{k=0}^{K} \|e_i^T P^k S\|^2 \right) \leq 1, \tag{70}
\]
which complete the proof. \( \square \)

**Remark.** Using the normalization \( \frac{1}{\sqrt{K+1}} \) also leads to a norm bounded by 1. Hence, the norm of each row of \( Z \) is bounded by 1. We need the normalization \( \frac{1}{\sqrt{K+1}} \) instead of \( \frac{1}{\sqrt{K+1}} \) to accommodate another claim in the proof.

A.18 Proof of Proposition A.6

**Proposition.** We have \( e_i^T |P - P'| e_j = e_i^T (P' - P) e_j, \forall i, j \neq m \). For \( i = m \) or \( j = m \), \( e_i^T |P - P'| e_j = e_i^T P e_j \).

**Proof.** For the first case when \( \forall i, j \neq m \),
\[
e_i^T (P - P') e_j = \frac{\hat{A}_{ij}}{\hat{D}_{ii}} - \frac{\hat{A}'_{ij}}{\hat{D}'_{ii}}. \tag{71}\]
Recall that by definition, in this case we have \( \hat{A}_{ij} = \hat{A}'_{ij} \). Now, there are two cases to consider: (1) \( i \) is a neighbor of \( m \); (2) \( i \) is not a neighbor of \( m \). For (1), we know that \( \hat{D}'_{ii} = \hat{D}_{ii} - 1 \geq 1 \). As a result,
\[
\frac{\hat{A}_{ij}}{\hat{D}_{ii}} - \frac{\hat{A}'_{ij}}{\hat{D}'_{ii}} = \frac{\hat{A}_{ij}}{\hat{D}_{ii}} - \frac{\hat{A}_{ij}}{\hat{D}_{ii}} - 1 < 0. \tag{72}\]
This directly implies \( e_i^T |P - P'| e_j = e_i^T (P' - P) e_j \). For (2), we know that \( \hat{D}'_{ii} = \hat{D}_{ii} \) and thus \( e_i^T |P - P'| e_j = 0 = e_i^T (P' - P) e_j \). These claims complete the proof for the first part. For the case that \( i = m \) or \( j = m \), note that since both the \( m \)th row and column are all-zeros for \( P' \), we simply have \( e_i^T |P - P'| e_j = e_i^T P e_j \). Note that in establishing the claim we also used the fact that \( P \) is nonnegative. This completes the proof. \( \square \)
Table 1: Properties of benchmarking datasets.

<table>
<thead>
<tr>
<th>Name</th>
<th>#nodes</th>
<th>#edges</th>
<th>#features</th>
<th>#classes</th>
<th>train/val/test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cora</td>
<td>2,708</td>
<td>10,556</td>
<td>1,433</td>
<td>7</td>
<td>1,208/500/1,000</td>
</tr>
<tr>
<td>Citeeseer</td>
<td>3,327</td>
<td>9,104</td>
<td>3,703</td>
<td>6</td>
<td>1,827/500/1,000</td>
</tr>
<tr>
<td>PubMed</td>
<td>19,717</td>
<td>88,648</td>
<td>500</td>
<td>3</td>
<td>18,217/500/1,000</td>
</tr>
<tr>
<td>Computers</td>
<td>13,752</td>
<td>491,722</td>
<td>767</td>
<td>10</td>
<td>12,252/500/1,000</td>
</tr>
<tr>
<td>Photo</td>
<td>7,650</td>
<td>238,162</td>
<td>745</td>
<td>8</td>
<td>6,150/500/1,000</td>
</tr>
<tr>
<td>ogbn-arxiv</td>
<td>169,343</td>
<td>1,166,243</td>
<td>128</td>
<td>40</td>
<td>90,941/29,799/48,603</td>
</tr>
</tbody>
</table>

A.19 ADDITIONAL EXPERIMENTAL DETAILS

All our experiments were executed on a Linux machine with 48 cores, 376GB of system memory, and two NVIDIA Tesla P100 GPUs with 12GB of GPU memory each. Information about all datasets can be found in Table 1. The data split is public and obtained from PyTorch Geometric [Pey & Lenssen (2019)]. We used the “full” split option for Cora, Citeeseer and Pubmed. Since there is no public split for Computers and Photo, we adopted a similar setting as for the citation networks via random splits (i.e., 500 nodes in the validation set and 1,000 nodes in the test set). The data split for ogbn-arxiv is the public split provided by the Open Graph Benchmark [Hu et al. (2020)].

Dependency on the node degree. We verified our Theorem 4.5 and Theorem 4.6 for node degree dependencies on Photo, Cora, Citeeseer and Pubmed. The results are presented in Figure 5.

Figure 5: Additional examination of the degree dependency result from Theorem 4.5 (top) and Theorem 4.6 (bottom).

Nonaccumulative time for each of the unlearning procedures. Figure 6 shows the average time complexity for each unlearning step on the Cora dataset. The spikes for approximate graph unlearning methods and Unstructured Unlearning [Guo et al. (2020)] corresponds to retraining after a removal.

Membership inference attacks for unlearned models. We performed experiments for node unlearning tasks and applied the membership inference attack for GNNs reported in Olatunji et al. (2021b) to our obtained updated models. For simplicity, we used the Cora dataset and removed up to 100 nodes. After each removal, we applied an MI attack on the updated model. We compare the results of our SGC node unlearning approach with that of the original SGC model without updates, which is the model trained on the full dataset, and with SGC retraining, which corresponds to the model obtained after retraining upon each removal request. We repeated the experiments with 10 different trials and random splits and averaged the results. As shown in Figure 7, even for full SGC retraining the attack model can still identify parts of the removed nodes in the training set, and the result of SGC node unlearning is slightly worse (w.r.t privacy) than retraining since our method is concerned with approximate unlearning. Note that the performance of the MI attack on the original model is consistent with the results from Olatunji et al. (2021b) and significantly worse than both our unlearning as well as the complete retraining method. This, from the experimental side, shows that
our method offers similar privacy-preserving performance as full retraining, and better performance when compared to the original model without unlearning. Nevertheless, the results also motivate the search for alternatives to MI attacks for unlearning schemes.

Additional experiments. The performance of our proposed approximate graph unlearning methods on three datasets, including Citeseer, Pubmed and Amazon Photo, is shown in Figure 8. It is worth pointing out that our bound on the gradient residual norm in Section 4 does not guarantee the generalization ability of the updated model. Therefore, It could happen that the test accuracy increases as we remove information from the training set, as shown in the second row of Figure 8, or that the performance is not very stable, as seen in the third row of Figure 8.

We also performed additional experiments on the Cora dataset, with results shown in Figure 9. The first row shows the average performance over 10 repeated trails with random splitting, and the conclusion is the same as the one stated in Section 6. The second row shows the performance on GPR-based models. Note that when the number of removal requests becomes large, the performance of GPR-based models degrades much faster than that of SGC-based models. This observation is consistent with our discussion of GPR-based models. None of the retraining methods involves noise.
Figure 8: Performance of approximate graph unlearning methods on different datasets. First Row: We removed up to 55% of the training data in Citeseer. Second Row: We removed up to 50% of the training data in Pubmed. Third Row: We set $\alpha = 10, \lambda = 10^{-4}$, and removed up to 30% of the training data in Amazon Photo.
Figure 9: Performance of approximate graph unlearning methods on Cora. First Row: The reported statistics are based on averaging over 10 repeated trails with random splitting. Second Row: GPR-based models are used to obtain node embedding. All other settings are the same as in Figure 2.