Efficient block contrastive learning via parameter-free meta-node approximation

Gayan K. Kulatilleke, Marius Portmann & Shekhar S. Chandra
University of Queensland, Brisbane, Australia.
{g.kulatilleke@uqconnect, marius@itee.uq, shekhar.chandra@uq}.edu.au

ABSTRACT

Contrastive learning has recently achieved remarkable success in many domains including graphs. However contrastive loss, especially for graphs, requires a large number of negative samples which is unscaleable and computationally prohibitive with a quadratic time complexity. Sub-sampling is not optimal and incorrect negative sampling leads to sampling bias. In this work, we propose a meta-node based approximation technique that can (a) proxy all negative combinations (b) in quadratic cluster size time complexity, (c) at graph level, not node level, and (d) exploit graph sparsity. By replacing node-pairs with additive cluster-pairs, we compute the negatives in cluster-time at graph level. The resulting Proxy approximated meta-node Contrastive (PamC) loss, based on simple optimized GPU operations, captures the full set of negatives, yet is efficient with a linear time complexity. By avoiding sampling, we effectively eliminate sample bias. We meet the criterion for larger number of samples, thus achieving block-contrastiveness, which is proven to outperform pair-wise losses. We use learnt soft cluster assignments for the meta-node constriction, and avoid possible heterophily and noise added during edge creation. Theoretically, we show that real world graphs easily satisfy conditions necessary for our approximation. Empirically, we show promising accuracy gains over state-of-the-art graph clustering on 6 benchmarks. Importantly, we gain substantially in efficiency; up to 3x in training time, 1.8x in inference time and over 5x in GPU memory reduction. Code: https://github.com/gayanku/PAMC

1 Introduction

Discriminative approaches based on contrastive learning has been outstandingly successful in practice [Guo et al., 2017] [Wang and Isola, 2020], achieving state-of-the-art results [Chen et al., 2020a] or at times outperforming even supervised learning [Logeswaran and Lee, 2018] [Chen et al., 2020b]. Specifically in graph clustering, contrastive learning can outperform traditional convolution and attention-based Graph Neural Networks (GNN) on speed and accuracy [Kulatilleke et al., 2022].

While traditional objective functions encourage similar nodes to be closer in embedding space, their penalties do not guarantee separation of unrelated graph nodes [Zhu et al., 2021]. Differently, many modern graph embedding models [Hamilton et al., 2017] [Kulatilleke et al., 2022], use contrastive objectives. These encourage representation of positive pairs to be similar, while making features of the negatives apart in embedding space [Wang and Isola, 2020]. A typical deep model consists of a trainable encoder that generates positive and negative node embedding for the contrastive loss [Zhu et al., 2021]. It has been shown that convolution is computationally expensive and may not be necessary for representation learning [Chen et al., 2020a]. As the requirement for contrastive loss is simply an encoder, recently researchers have been able to produce state-of-the-art results using simpler and more efficient MLP based contrastive loss implementations [Hu et al., 2021] [Kulatilleke et al., 2022]. Thus, there is a rapidly expanding interest and scope for contrastive loss based models.
We consider the following specific but popular form of contrastive loss where $\tau$ is the temperature parameter, $\gamma_{ij}$ is the relationship between nodes $i, j$ and the loss for the $i^{th}$ node is:

$$\ell_i = -\log \frac{\sum_{j=1, j \neq i}^{B} 1_{[j \neq i]} \gamma_{ij} \cdot \exp \left( \frac{\text{sim} (z_i, z_j) \cdot \tau}{\tau} \right)}{\sum_{k=1}^{B} 1_{[k \neq i]} \exp \left( \frac{\text{sim} (z_i, z_k) \cdot \tau}{\tau} \right)},$$

(1)

When no labels are present, sampling of positive and negative nodes plays a crucial role and is a key implementation detail in contrastive methods. Positive samples in graphs are typically connected by edges, similar to words in a sentence in language modelling. Often data augmentation is used to generate positive samples; used crop, colouring, blurring. However, it is harder to obtain negative samples. With no access to labels, negative counterparts are typically obtained via uniform sampling, via synthesizing/augmenting or adding noise. Also, in graphs, adjacency information can be exploited to derive negatives for feature contrastion. However, while graphs particularly suited for contrastive learning, to be effective, a large number of negative samples must be used, along with larger batch sizes and longer training compared to its supervised counterparts.

Unlike other domains, such as vision, negative sample generation brings only limited benefits to graphs. To understand this phenomenon, observe the raw embedding of USPS image dataset, in the top row of Figure 4 which looks already clustered. A direct consequence of this is that graphs are more susceptible to sampling bias. Thus, graph contrastive learning approaches suffer from insufficient negatives and the complex task of sample generation in addition to $O(N^2)$ time complexity required to contrast every negative node.

However, what contrastive loss exactly does remain largely a mystery. For example, ’s analysis based on the assumption of latent classes provides good theoretical insights, yet their explanation on representation quality dropping with large number of negatives is inconsistent with experimental findings. Contrastive loss is seen as maximizing mutual information (MI) between two views. Yet, contradictorily, tighter bound on MI can lead to poor representations.

Motivation: jointly identifies alignment and uniformity as key properties of contrastive loss: alignment encourages encoders to assign similar features to similar samples; uniformity encourages a feature distribution that preserves maximal information. It is fair to assume that latent clusters are dissimilar. Even with the rare possibility of two identical cluster centers initially, one will usually change or drift apart. It is intuitive that cluster centers should be uniformly distributed in the hyperspace, similar to nodes, in order to preserve as much information of the data as possible. Uniformly distributing points on a hyperspace is defined as minimizing the total pairwise potential w.r.t. a certain kernel function and is well-studied.

Thus, we are naturally motivated to use the cluster centers as meta-nodes for negative contrastion. By aggregation, all its constituent nodes can be affected. Thus, we avoid sampling, effectively eliminate sample bias, and also meet the criterion of larger number of samples. Learned soft cluster assignments can avoid possible heterophily and add robustness to noise in edge construction.

In this work, we propose a novel contrastive loss, $\text{PamC}$, which uses parameterless proxy meta-nods to approximate negative samples. Our approach indirectly uses the full set of negative samples and yet is efficient with a time complexity of $O(N)$. Not only does $\text{PamCGC}$ based on $\text{PamC}$ outperform or match previous work, but it is also simpler than any prior negative sample generation approach, faster and uses relatively less GPU memory. It can be incorporated to any contrastive learning-based clustering model with minimal modifications, and works with diverse data, as we demonstrate using benchmark datasets from image, text and graph modalities.

To summarize, our main contributions are:

- We introduce an efficient novel parameter-free proxy, $\text{PamC}$ for negative sample approximation that is scalable, computationally efficient and able to include all samples. It works with diverse data, including graphs. We claim $\text{PamC}$ is the first to implicitly use the whole graph with $O(N)$ time complexity, in addition to further 3-fold gains.
- We provide theoretical proof and show that real world graphs always satisfies the necessary conditions, and that $\text{PamCGC}$ is block-contrastive, known to outperform pair-wise losses.
- Extensive experiments on 6 benchmark datasets show $\text{PamCGC}$ using proposed $\text{PamC}$ is on par with or better than state-of-the-art graph clustering methods in accuracy while achieving 3x training time, 1.8 inference time and 5x GPU memory efficiency.
2 Implementation

First we describe PamC, which is our parameter-free proxy to efficiently approximate the negative samples, as shown in Figure 1. Next, we introduce PamCGC, a self-supervised model based on PamC, to simultaneously learn discriminative embeddings and clusters.

2.1 Negative sample approximation by meta-node proxies

Contrastive loss makes positive or connected nodes closer and negative or unconnected nodes further away in the feature space [Kulatilleke et al., 2022]. However, in order to be effective, all negative nodes need to be contrasted with $x_i$, which is computationally expensive. A cluster centre is formed by combining all member nodes, and can be seen as an aggregated representation, or a proxy, of its compositional elements. Motivated by this, we use the cluster centres to enforce negative contrastion. Specifically, we contrast every cluster centre $\hat{\mu}_i$ with every cluster centre $\hat{\mu}_j$ where $i \neq j$.

Following Arora et al. [2019], Chuang et al. [2020], we assume an underlying set of discrete latent classes $C$ which represents semantic content, i.e., similar nodes $x_i, x_j$ are in the same latent class $\hat{\mu}$. Thus, we derive our proxy for negative samples as:

$$\ell_{proxy} = \log \sum_{a=1}^{C} \sum_{b=1}^{C} 1_{a \neq b} \exp (\text{sim} (\hat{\mu}_a, \hat{\mu}_b) \cdot \tau),$$

(2)

Note that, $\ell_{proxy}$ contains no $i$ or $j$ terms! resulting in three fold gains. Firstly, we replace $\sum_{i=1}^{N}$, with a more efficient $\sum_{a=1}^{C}$ where $N \gg C$, typically many magnitudes, in almost all datasets, as evident from Table 1. Secondly, the $\ell_{proxy}$ is at graph level with time complexity of $O(N)$ rather than an instance level $O(N^2)$. Finally, given real world
graphs (especially larger graphs,) are sparse, a sparse implantation for the positives, using edge-lists, will result in a 
third efficiency gain, which is only possible by not having to operate on the negatives explicitly.

Note that a prerequisite of the proxy approximation is the availability of labels to construct the learned cluster centers \( \mu \), which we explain in the next section. Thus, the complete graph level contrastive loss can be expressed as:

\[
\ell_{\text{pcontrast}} = -\frac{1}{N} \sum_{i=1}^{N} \log \sum_{j=1}^{N} 1_{[j \neq i]} \gamma_{ij} \exp \left( \langle z_i, z_j \rangle \cdot \tau \right) + \ell_{\text{proxy}},
\]

(3)

**Theoretical explanation.** The standard contrastive loss uses Jensen-Shannon divergence, which yields \( \log 2 \) constant and vanishing gradients for disjoint distributions of positive and negative sampled pairs [Zhu et al., 2021]. However, in the proposed method, positive pairs are necessarily edge-linked (either explicitly or via influence [Kulatilleke et al., 2022]), and unlikely to be disjoint. Using meta-nodes for negatives, which are compositions of multiple nodes, lowers the possibility of disjointness. An algorithm using the average of the positive and negative samples in blocks as a proxy instead of just one point has a strictly better bound due to Jensen’s inequality getting tighter and is superior compared to their equivalent of element-wise contrastive counterparts [Arora et al., 2019]. The computational and time cost is a direct consequence of node level contrastion. Given, \( N \gg \text{clusters} \), we circumvent the problem of large \( N \) by proposing a proxy-ed negative contrastive objective that operates directly at the cluster level.

Establishing mathematical guarantee: Assume node embeddings \( Z = \{z_1, z_2, z_3, \ldots z_N\} \), clusters \( \mu = \{\mu_1, \mu_2, \ldots \mu_C\} \), a label assignment operator \( \text{label}(z_i) \) such that \( \mu_a = \sum_{i=1}^{N} 1_{[i \in \text{label}(z_i) = a]} \cdot z_i \), a temperature hyper-parameter \( \tau \) and,

\[
\text{similarity}(i,j,z_i,z_j) = \text{sim}(z_i,z_j) \begin{cases} 0, & i = j \\ \frac{z_i \cdot z_j}{\|z_i\| \cdot \|z_j\|}, & i \neq j \end{cases}
\]

(4)

Using \( \text{sim}(z_i,z_j) \) as the shorthand notation for \( \text{similarity}(i,j,z_i,z_j) \), the classic contrastive loss is:

\[
\text{loss}_{NN} = \frac{1}{N} \sum_{i=1}^{N} \log \left( \sum_{j=1}^{N} \exp(\text{sim}(i,j,z_i,z_j) \tau) \right)
\]

(5)

Similarly, we can express the cluster based contrastive loss as:

\[
\text{loss}_{CC} = \frac{1}{C} \sum_{a=1}^{C} \log \left( \sum_{b=1}^{M} \exp(\text{sim}(a,b,\mu_a,\mu_b) \tau) \right)
\]

(6)

As \( 0 \leq \text{sim} \leq 1.0 \), we establish the condition for our inequality as;

\[
\frac{\text{loss}_{NN}}{\text{loss}_{CC}} > \frac{\log(N)}{\log [1 + (C - 1)e^\tau]}
\]

(7)

We provide the full derivation in Appendix A.1.

As \( C > 1 \) (minimum 2 are needed for a cluster), and \( \log(x) : x > 0 \) is strictly increasing, \( N > 1 + (C - 1)e^\tau \) is the necessary condition, which is easily satisfied for nearly all real world datasets and as seen in Figure 2 for multiple \( \tau \) temperatures.

Thus, as \( \text{loss}_{NN} > \text{loss}_{CC} \), \( \text{loss}_{NN} \) upper bounds \( \text{loss}_{CC} \), the more efficient variant. Additionally \( \text{loss}_{CC} \) benefits from block-contrastiveness [Arora et al., 2019], achieves a lower minima and uses the fullest possible negative information. We also show, experimentally, that minimizing \( \text{loss}_{CC} \) results in effective, and sometimes better, representations for downstream tasks.

**2.2 Constructing the meta-node cluster centres (\( \hat{\mu} \))**

In order to derive the real cluster centers \( \hat{\mu} \), which is distinct from the learnt cluster centers \( \mu \), we simply aggregate all the node embedding \( z \) of a cluster using its label. Even with unlabeled data, \( \text{label}() \) can be accomplished using predicted soft labels. The intuition here is that, during back-propagation, the optimization process will update the constituent node embeddings, \( z \), to incorporate negative distancing. Thus,

\[
\hat{\mu}_c = \frac{1}{N} \sum_{i=1}^{N} 1_{[i \in \text{label}(c)]} z_i
\]

(8)
where label(c) is either the ground-truth or learnt soft labels. Accordingly, our proxy can equally be used in supervised and unsupervised scenarios and has a wider general applicability as an improvement of the contrastive loss at large. Finlay, Equation 8 can be realized with $\text{softmax}(\cdot)$ and $\text{mean}(\cdot)$ operations, which are well optimized GPU primitives in any machine learning framework. We provide a reference pytorch implementation.

### 2.3 Obtaining the soft labels

Graph clustering is essentially unsupervised. To this end, following Xie et al. [2016], Guo et al. [2017], Wang et al. [2019], Kulatilleke et al. [2022], we use probability distribution derived soft-labels and a self-supervision mechanism for cluster enhancement. Specifically, we obtain soft cluster assignments probabilities $q_{iu}$ for embedding $z_i$ and cluster centre $\mu_u$. In order to handle differently scaled clusters and be computationally convenient [Wang et al., 2019], we use the student’s t-distribution [Maaten and Hinton, 2008] as a kernel for the similarity measurement between the embedding and centroid:

$$q_{iu} = \frac{1 + ||z_i - \mu_u||^2 / \eta}{\sum_u'(1 + ||z_i - \mu_u'||^2 / \eta)^{-\frac{n+1}{2}}}.$$ (9)

where, $\eta$ is the Student’s t-distribution’s degree of freedom. Cluster centers $\mu$ are initialized by K-means on embeddings from the pre-trained AE. We use $Q = [q_{iu}]$ as the distribution of the cluster assignments of all samples, and $\eta=1$ for all experiments following [Bo et al., 2020], Peng et al. [2021], Kulatilleke et al. [2022].

Nodes closer in embedding space to a cluster center has higher soft assignment probabilities in $Q$. A target distribution $P$ that emphasizes the confident assignments is obtained by squaring and normalizing $Q$, given by :

$$p_{iu} = \frac{q_{iu}^2 / \sum_i q_{iu}}{\sum_k (q_{ik}^2 / \sum_i q_{ik})},$$ (10)

where $\sum_i q_{iu}$ is the soft cluster frequency of centroid $u$.

Following Kulatilleke et al. [2022], we minimize the KL divergence between $Q$ and $P$ distributions, which forces the current distribution $Q$ to approach the more confident target distribution $P$. KL divergence updates models more gently and lessens severe disturbances on the embeddings [Bo et al., 2020]. Further, it can accommodate both the structural and feature optimization targets of PamCGC. We self-supervise cluster assignments $1$ by using distribution $Q$ to target distribution $P$, which then supervises the distribution $Q$ in turn by minimizing the KL divergence as:

$$\text{loss}_{\text{cluster}} = KL(P||Q) = \sum_i \sum_u p_{iu}log\frac{p_{iu}}{q_{iu}},$$ (11)

The final proposed model, after incorporating PamC contrastive objective with self-supervised clustering, where $\alpha > 0$ controls structure incorporation and $\beta > 0$ controls cluster optimization is:

$$\text{PamCGC} : L_{\text{final}} = \alpha \ell_{\text{Pcontrast}}(K, \tau) + \beta \text{loss}_{\text{cluster}},$$ (12)

1We follow Bo et al. [2020] use of the term ‘self-supervised’ to be consistent with the GCN training method.
Table 1: Statistics of the datasets (left) and PamCGC hyperparameters (right).

| Dataset       | Type    | Nodes | Classes | dimension | α   | β   | K      | τ     | LR     |
|---------------|---------|-------|---------|-----------|-----|-----|--------|-------|--------|
| USPS Image    | Image   | 9298  | 10      | 256       | 2   | 2   | 4      | 0.5   | \(10^{-3}\) |
| HHAR Record   | Record  | 10299 | 6       | 561       | 0.5 | 12.5| 2      | 1.5   | \(10^{-3}\) |
| RET Text      | Text    | 10000 | 4       | 2000      | 0.5 | 1   | 0.2    | 0.25  | \(10^{-4}\) |
| ACM Graph     | Graph   | 3025  | 3       | 1870      | 0.5 | 0.5 | 2      | 1     | \(10^{-3}\) |
| CITE Graph    | Graph   | 3327  | 6       | 3703      | 2   | 2   | 1      | 1     | \(10^{-3}\) |
| DBLP Graph    | Graph   | 4057  | 4       | 334       | 2   | 2.5 | 3      | 0.5   | \(10^{-3}\) |

2.4 Complexity Analysis

Given input data dimension \(d\) and AE layer dimensions of \(d_1, d_2, \ldots, d_L\), following Kulatilleke et al. [2022], \(O_{AE} = O(N d^2 d_1^2 \cdots d_L^2)\) for PamCGC AE. Assuming \(K\) clusters, from Equation 9, the time complexity is \(O_{cluster} = O(NK + N \log N)\) following Xie et al. [2016].

For PamC, we only compute \(\|z\|_2^2\) and \(z_i \cdot z_j\) for the actual positive edges \(E\) using sparse matrix resulting in a time complexity \(O_+ = O(N Ed_z)\), linear with the number of edges \(E\), with \(d_z\) embedding dimension. For the negatives, we use the meta-node based negatives \(O_- = O(CC)\) where \(C\) is the meta-node. Note that, for real graphs, \(N \gg C\) in many magnitudes. Thus, the overall time complexity is linearly related to the number of samples and edges.

3 Experiments

We evaluate PamCGC on transductive node clustering comparing to state-of-the-art self-supervised, contrastive and (semi-)supervised methods.

Datasets. Following Bo et al. [2020], Peng et al. [2021], Kulatilleke et al. [2022], experiments are conducted on six common clustering benchmarks, which includes one image dataset (USPS [Le Cun et al., 1990]), one sensor data dataset (HHAR [Stisen et al., 2015]), one text dataset (REUT [Lewis et al., 2004]) and three citation graphs (ACM\(^2\), CITE\(^4\), and DBLP\(^3\)). For the non-graph data, we use undirected \(k\)-nearest neighbour (KNN [Altman, 1992]) to generate adjacency matrix \(A\) following Bo et al. [2020], Peng et al. [2021]. Table 1 summarizes the datasets.

Baseline Methods. We compare with multiple models. K-means [Hartigan and Wong, 1979] is a classical clustering method using raw data. AE [Hinton and Salakhutdinov, 2006] applies K-means to deep representations learned by an auto-encoder. DEC [Xie et al. 2016] clusters data in a jointly optimized feature space. IDEC [Guo et al., 2017] enhances DEC by adding KL divergence-based reconstruction loss. Following models exploit graph structures during clustering: SVD [Golub and Reinsch, 1971] applies singular value decomposition to the adjacency matrix. DGI [Velickovic et al., 2019] learns embeddings by maximizing node MI with the graph. GAE [Kipf and Welling, 2016] combines convolution with the AE. ARGA [Pan et al., 2018] uses an adversarial regularizer to guide the embeddings learning. Following deep graph clustering jointly optimize embeddings and graph clustering: DAEGC [Wang et al., 2019], uses an attentional neighbor-wise strategy and clustering loss. SDNC [Bo et al., 2020], couples DEC and GCN via a fixed delivery operator and uses feature reconstruction. AGCN [Peng et al., 2021], extends SDNC by adding an attention-based delivery operator and uses multi scale information for cluster prediction. CGC [Park et al., 2022] uses a multi-level, hierarchy based contrastive loss. SCGC and SCGC\(^*\) [Kulatilleke et al., 2022] uses block contrastive loss with an AE and MLP respectively. The only difference between SCGC\(^*\) and PamCGC is the novel PamC loss. Also as SCGC\(^*\) is the current state-of-the-art. Thus, it is used as the benchmark.

Evaluation Metrics. Following Bo et al. [2020], Peng et al. [2021], we use Accuracy (ACC), Normalized Mutual Information (NMI), Average Rand Index (ARI), and macro F1-score (F1) for evaluation. For each, larger values imply better clustering.

3.1 Implementation

The positive component of our loss only requires the actual connections and can be efficiently represented by sparse matrices. Further, the negative component of the loss is graph-based, and not instance based, thus needs to be computed.
Table 2: Clustering performance the three graph datasets (mean±std). Best results are bold. Results reproduced from Bo et al. [2020], Peng et al. [2021], Kulatilleke et al. [2022], Park et al. [2022]. SCGC [Kulatilleke et al., 2022] uses neighbor based contrastive loss with AE while SCGC* variant uses r-hop cumulative Influence contrastive loss with MLP, same as our PamGC.

| Method | DBLP | ACM | CITE |
|--------|------|-----|------|
|        | ACC  | NMI | ARI | F1  | ACC  | NMI | ARI | F1  | ACC  | NMI | ARI | F1  |
| K-means| 38.7±0.4 | 11.5±0.4 | 7.0±0.4 | 31.9±0.3 | 67.3±0.7 | 32.4±0.5 | 30.6±0.7 | 67.6±0.7 | 39.3±0.2 | 16.9±1.2 | 13.4±1.0 | 36.1±3.5 |
| AE     | 51.4±0.4 | 25.4±0.2 | 12.2±0.4 | 35.2±0.4 | 81.8±0.1 | 50.3±0.2 | 54.6±0.2 | 82.0±0.1 | 57.1±0.1 | 27.6±1.1 | 29.3±0.1 | 53.8±0.1 |
| DEC    | 58.3±2.0 | 29.5±0.3 | 23.9±0.4 | 39.0±0.5 | 84.3±0.8 | 54.5±1.5 | 60.6±1.9 | 84.5±0.7 | 55.9±0.2 | 28.3±1.3 | 28.1±0.2 | 52.6±0.2 |
| IDEC   | 63.0±3.0 | 31.2±0.5 | 25.4±0.6 | 61.0±0.5 | 85.6±1.2 | 62.2±1.5 | 85.1±0.5 | 60.5±1.4 | 27.2±2.4 | 25.7±2.7 | 61.6±1.4 |
| SVD    | 29.3±0.4 | 0.1±0.0 | 0.0±0.1 | 13.3±2.2 | 39.9±5.8 | 3.8±4.3 | 3.1±4.2 | 30.1±8.2 | 24.1±1.2 | 5.7±1.5 | 0.3±1.3 | 14.2±1.7 |
| DGI    | 32.5±2.4 | 3.7±1.8 | 1.7±0.9 | 32.3±3.3 | 82.0±1.1 | 63.0±1.9 | 67.7±2.5 | 88.0±1.0 | 61.4±1.3 | 38.8±1.2 | 38.1±1.9 | 60.4±0.9 |
| GAE    | 61.2±1.2 | 30.8±0.9 | 22.0±1.4 | 61.4±5.4 | 15.4±4.1 | 95.5±3.1 | 84.7±1.3 | 61.4±0.8 | 34.6±0.7 | 33.6±1.2 | 57.4±0.8 |
| VGEA   | 58.6±1.0 | 26.9±0.1 | 17.9±0.1 | 58.7±0.1 | 84.1±0.2 | 53.2±0.5 | 57.7±0.7 | 84.2±0.2 | 61.0±0.4 | 32.7±3.3 | 33.1±0.5 | 57.7±0.5 |
| ARGA   | 61.6±1.0 | 26.8±1.0 | 22.7±0.3 | 61.8±0.9 | 86.1±1.2 | 55.7±1.4 | 62.9±2.1 | 86.1±1.2 | 56.9±0.7 | 34.5±0.8 | 33.4±1.5 | 54.8±0.8 |
| DAEGF  | 62.1±0.5 | 32.5±0.5 | 21.0±0.5 | 61.8±0.7 | 76.9±0.2 | 58.3±0.4 | 59.4±3.9 | 87.1±2.8 | 65.3±1.4 | 36.4±0.9 | 37.8±1.2 | 62.2±1.3 |
| SCGC   | 77.6±6.5 | 46.1±0.6 | 49.7±1.1 | 77.2±0.4 | 92.3±0.3 | 72.9±0.7 | 78.4±0.6 | 92.3±0.3 | 69.0±0.6 | 44.6±0.6 | 46.0±0.6 | 65.5±0.7 |
| SCGC*  | 77.7±6.1 | 47.1±0.1 | 50.9±0.1 | 77.5±0.1 | 92.6±0.0 | 73.7±0.1 | 79.4±0.1 | 92.6±0.0 | 73.7±0.0 | 46.9±0.0 | 50.2±1.0 | 63.4±0.0 |

Table 3: Clustering performance the three non-graph datasets (mean±std). Best results are bold; second best is underlined. Results reproduced from Bo et al. [2020], Peng et al. [2021], Kulatilleke et al. [2022]. SCGC [Kulatilleke et al., 2022] uses neighbour based contrastive loss with AE while SCGC* variant uses r-hop cumulative Influence contrastive loss with MLP, same as our PamGC.

| Dataset | Metric | K-means | SCGC | SCGC* |
|---------|--------|---------|------|-------|
|         | ACC    | NMI    | ARI  | F1   | ACC   | NMI   | ARI  | F1   | ACC   | NMI   | ARI  | F1   |
| USPS    |        |        |      |      |       |       |      |      |       |       |      |      |
|         | 68.2±0.0 | 63.1±0.3 | 56.1±0.3 | 47.0±0.3 | 23.5±0.4 | 30.8±0.4 | 70.1±0.4 | 70.8±0.1 | 80.9±0.8 | 82.9±0.8 | 84.9±0.6 | 84.7±1.4 |
| HHR     |        |        |      |      |       |       |      |      |       |       |      |      |
|         | 58.6±0.1 | 55.0±0.1 | 40.9±0.1 | 36.7±0.1 | 71.2±0.6 | 74.9±0.6 | 75.1±0.2 | 79.6±0.4 | 82.4±0.8 | 85.8±1.0 | 79.5±0.6 | 77.7±0.6 |
| REUT    |        |        |      |      |       |       |      |      |       |       |      |      |
|         | 58.8±0.1 | 54.9±0.1 | 42.3±0.1 | 38.0±0.1 | 65.1±0.1 | 68.1±0.1 | 71.7±0.5 | 75.4±0.5 | 80.8±0.5 | 85.4±0.6 | 75.3±0.6 | 75.2±0.7 |

only once per epoch. Thus, by decoupling the negatives, our loss is inherently capable of batching and is trivially parallelizable. Computation of the negative proxy, which is only $C \cdot C$ does not even require a GPU!

For fair comparison, we use the same 500 – 500 – 2000 – 10 AE dimensions as in Quo et al. [2017], Bo et al. [2020], Peng et al. [2021], Kulatilleke et al. [2022] and the same pre-training procedure, i.e. 30 epochs; learning rate of $10^{-4}$ for USPS, HHR, ACM, DBLP and $10^{-4}$ for REUT and CITE; batch size of 256. We made use of the publicly available pre-trained AE from Bo et al. [2020]. We use a once computed edge-list for training, which is not needed during inference. For each dataset, we initialize the cluster centers from $K$-means and repeat the experiments 10 times with 200 epochs to prevent extreme cases. We cite published accuracy results from Bo et al. [2020], Peng et al. [2021], Kulatilleke et al. [2022] for other models.

For all timing and memory experiments, we replicate the exact same training loops, including internal evaluation metric calls, when measuring performance for fair comparison. Our code will be made publicly available.

### 3.2 Quantitative Results

We show our hyperparameters in Table 1. Comparison of results with state-of-the-art graph and non-graph datasets are in Table 2 and Table 3 respectively. For the graph data, PamGC is state-of-the-art for DBLP. A paired-t test shows ACM and CITE results to be best for both SCGC* and PamGC in non-graph results, PamGC comes second best in USPS image data. While results for HHR are somewhat lagging, PamGC is the best for REUT. Generally we achieve better results on the natural graph datasets; ACM, DBLP and CITE, while being competitive on other modalities.
3.3 Performance

In Figure 3, we compare the GPU-based training, inference times, and GPU memory. Our model times also include the time taken for the cumulative influence computation. For all the datasets, PamCGC is superior by 3.6x training time, 1.8x inference time, and 5.3x GPU memory savings. Especially, for larger datasets USPS, HHAR, and REUT, PamCGC uses 5.2, 7.7, 8.7x less GPU memory. Note that SCGC only differs from PamCGC by its use of the novel proxy-ed PamC to which we solely attribute the time and memory savings.

3.4 Qualitative Results

We use UMAP [McInnes et al., 2018], in Figure 4, to get a visual understanding of the raw and learnt embedding spaces. Except for USPS, which is a distinct set of 0-9 handwritten digits (raw 1), we see that all other datasets produce quite indistinguishable clusters. Clustering is nearly non-existent in the (last 3) graph datasets. This clearly shows a characteristic difference in graph data, which can lead to high samplings bias. Note that $\hat{\mu} \neq \mu$ for any meta-node.
3.5 Ablation study

Figure 5: Left: GPU training times with PamCGC on SDCN and AGCN is consistently lower, and significant in large datasets (usps, hhar, reut). Right: Accuracy loss of the approximation is very low. For dblp, usps, reut accuracy is actually better.

To investigate PamC’s ability to generalize to other models, we incorporate it to SDCN and AGCN models, modified for contrastive loss. Figure 5 shows the GPU training time and accuracy. As PamC is a loss function, there is no difference in the inference times. As expected, training times are significantly shorter, with (often better) training accuracy due to block contrastiveness. Note that PamC only improves loss computation efficiency. Majority of the SDCN and AGCN computation time is spent in their GNNs convolution operations.

We also carry out extensive experimentation to assess the behavior of hyperparameters. PamC is robust to changes in hyperparameter values and performs best with a learning rate of 0.001, as shown in Appendix A.2. Further, PamC accuracy against all hyperparameter combinations is generally equal or better than the less efficient non proxy-ed contrastive loss variant, as seen in Appendix A.3.

3.6 Future work

Our parameter-free proxy-ed contrastive loss uses the full positive edge information which, as some of our experiments has shown, is redundant. For example, USPS gives similar results with 40% positive edges removed. An algorithm to drop un-informative edges may result in further efficiency improvements, which we leave for future work. While theoretically possible, it would be interesting to see how our proxy-ed contrastive loss works with semi or fully supervised data. Further study is needed to explore how hard cluster centers effect the optimization process. Contrastive loss for vision typically contrasts \(N\) anchor images, each augmented to create two views of the same sample \(x_i, x_j\) with the remaining \(2(N - 1)\) [Chen et al., 2020b, Chuang et al., 2020]. While this approach is different from ours, we have shown a possible use case with USPS for vision data.

4 Conclusion

In this work, we present an efficient parameter-free proxy approximation to incorporate negative samples in contrastive loss for joint clustering and representation learning. We eliminate sample bias, achieve block contrastiveness and \(O(N)\). Our work is supported by theoretical proof and empirical results. We improve considerably over previous methods accuracy, speed and memory usage. Our approach differs from prior self-supervised clustering by the proxy mechanism we use to incorporate all negative samples efficiently. The strength of this simple approach indicates that, despite the increased interest in graphs, effective contrastive learning remains relatively unexplored.

Acknowledgments

Dedicated to Sugandi.
References

Naomi S Altman. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185, 1992.

Sanjeev Arora, Hrishikesh Khandeparkar, Mikhail Khodak, Orestis Plevrakis, and Nikunj Saunshi. A theoretical analysis of contrastive unsupervised representation learning. In *36th International Conference on Machine Learning, ICML 2019*, pages 9904–9923. International Machine Learning Society (IMLS), 2019.

Deyu Bo, Xiao Wang, Chuan Shi, Meiqi Zhu, Emiao Lu, and Peng Cui. Structural deep clustering network. In *Proceedings of The Web Conference 2020*, pages 1400–1410, 2020.

Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *International Conference on Machine Learning*, pages 1725–1735. PMLR, 2020a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020b.

Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debias contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Linear algebra*, pages 134–151. Springer, 1971.

Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: node classification without message passing in graph. *arXiv preprint arXiv:2106.04051*, 2021.

Thomas N Kipf and Max Welling. Variational graph auto-encoders. *arXiv preprint arXiv:1611.07308*, 2016.

Gayan K Kulatilleke, Marius Portmann, Ryan Ko, and Shekhar S Chandra. Fdgatii: Fast dynamic graph attention with initial residual and identity mapping. *arXiv preprint arXiv:2110.11464*, 2021.

Gayan K Kulatilleke, Marius Portmann, and Shekhar S Chandra. Scgc: Self-supervised contrastive graph clustering. *arXiv preprint arXiv:2204.12656*, 2022.

Yann Le Cun, Ofer Matan, Bernhard Boser, John S Denker, Don Henderson, Richard E Howard, Wayne Hubbard, LD Jacket, and Henry S Baird. Handwritten zip code recognition with multilayer networks. In *ICPR*, volume 2, pages 35–40. IEEE, 1990.

David D Lewis, Yiming Yang, Tony G Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of machine learning research*, 5(Apr):361–397, 2004.

Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018.

Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, Lina Yao, and Chengqi Zhang. Adversarially regularized graph autoencoder for graph embedding. *arXiv preprint arXiv:1802.04407*, 2018.

Namyoung Park, Ryan Rossi, Eunyee Koh, Iftikhar Ahamath Burhanuddin, Sungchul Kim, Fan Du, Nesreen Ahmed, and Christos Faloutsos. Cgc: Contrastive graph clustering for community detection and tracking. In *Proceedings of the ACM Web Conference 2022*, pages 1115–1126, 2022.

Zhihao Peng, Hui Liu, Yuheng Jia, and Junhui Hou. Attention-driven graph clustering network. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 935–943, 2021.
Allan Stisen, Henrik Blunck, Sourav Bhattacharya, Thor Siiger Prentow, Mikkel Baun Kjærgaard, Anind Dey, Tobias Sonne, and Mads Møller Jensen. Smart devices are different: Assessing and mitigating mobile sensing heterogeneities for activity recognition. In SenSys, pages 127–140, 2015.

Petar Velickovic, William Fedus, William L Hamilton, Pietro Liò, Yoshua Bengio, and R Devon Hjelm. Deep graph infomax. ICLR (Poster), 2(3):4, 2019.

Chun Wang, Shirui Pan, Ruiqi Hu, Guodong Long, Jing Jiang, and Chengqi Zhang. Attributed graph clustering: A deep attentional embedding approach. arXiv preprint arXiv:1906.06532, 2019.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In International Conference on Machine Learning, pages 9929–9939. PMLR, 2020.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In ICML, pages 478–487, 2016.

Hao Zhu, Ke Sun, and Peter Koniusz. Contrastive laplacian eigenmaps. Advances in Neural Information Processing Systems, 34, 2021.
A Appendix

A.1 Proofs of Theoretical Results - Derivation of Equation 7

Assume node embeddings \( Z = \{z_1, z_2, z_3 \ldots z_N\} \), clusters \( \mu = \{\mu_1, \mu_2 \ldots \mu_C\} \), a label assignment operator \( \text{label}(z_i) \) such that \( \mu_a = \sum_{i=1}^{N} 1_{[i \in \text{label}(z_i) = a]} \cdot z_i \), a hyperparameter \( \tau \) related to the temperature in contrastive loss and

\[
\text{similarity}(i, j, z_i, z_j) = \sim(z_i, z_j) \begin{cases} 
0, & i = j \\
\frac{z_i \cdot z_j}{\|z_i\| \cdot \|z_j\|}, & i \neq j
\end{cases}
\]  

(13)

We use \( \sim(z_i, z_j) \) as the shorthand notation for \( \text{similarity}(i, j, z_i, z_j) \) interchangeably for brevity.

We begin with Equation 4 which is the popular form of contrastive loss \cite{hu2021learning, kulatilake2022self}. With \( \tau \) as the temperature parameter, \( \gamma_{ij} \) the relationship between nodes \( i, j \), the loss for the \( i^{th} \) can be expanded as:

\[
\ell_i = + \log \sum_{j=1}^{B} 1_{[j \neq i]} \exp (\sim(z_i, z_j) \tau) - \log \sum_{j=1}^{B} 1_{[j \neq i]} \gamma_{ij} \exp (\sim(z_i, z_j) \tau),
\]  

(14)

where, the first part on the right corresponds to the negative node contrasting portion and the second portion contrasts the positives for node \( i \). From Equation 14, for all nodes \( N \), we take to negative node contrasting portion, by averaging over \( N \) nodes to obtain:

\[
\text{loss}_{NN} = \frac{1}{N} \sum_{i=1}^{N} \log \left[ \sum_{j=1}^{N} e^{\text{sim}(i, j, z_i) \tau} \right],
\]  

(15)

Note our use of the more concise \( \sim() \) and the compact \( e \) notation over \( \exp() \) interchangeably for compactness reasons.

We expand Equation 15 together with \( e^0 = 1 \) in cases where \( i = j \), as:

\[
\text{loss}_{NN} = \frac{1}{N} \log \left[ 1 + e^{\sim(z_1, z_2) \tau} + e^{\sim(z_1, z_3) \tau} + e^{\sim(z_1, z_4) \tau} \ldots + e^{\sim(z_1, z_N) \tau} \right] + \log \left[ e^{\sim(z_2, z_3) \tau} + 1 + e^{\sim(z_2, z_3) \tau} + e^{\sim(z_2, z_4) \tau} \ldots + e^{\sim(z_2, z_N) \tau} \right] + \log \left[ e^{\sim(z_3, z_1) \tau} + e^{\sim(z_3, z_2) \tau} + 1 + e^{\sim(z_3, z_2) \tau} \ldots + e^{\sim(z_3, z_N) \tau} \right] + \ldots
\]  

(16)

Similarly, we can express the cluster based contrastive loss as:

\[
\text{loss}_{CC} = \frac{1}{C} \sum_{a=1}^{C} \log \left[ \sum_{b=1}^{M} e^{\text{sim}(a, b, \mu_a, \mu_b) \tau} \right]
\]  

(17)

with the following expansion:

\[
\text{loss}_{CC} = \frac{1}{C} \log \left[ 1 + e^{\sim(\mu_1, \mu_2) \tau} + e^{\sim(\mu_1, \mu_3) \tau} + e^{\sim(\mu_1, \mu_4) \tau} \ldots + e^{\sim(\mu_1, \mu_C) \tau} \right] + \log \left[ e^{\sim(\mu_2, \mu_1) \tau} + 1 + e^{\sim(\mu_2, \mu_3) \tau} + e^{\sim(\mu_2, \mu_4) \tau} \ldots + e^{\sim(\mu_2, \mu_C) \tau} \right] + \log \left[ e^{\sim(\mu_3, \mu_1) \tau} + e^{\sim(\mu_3, \mu_2) \tau} + 1 + e^{\sim(\mu_3, \mu_2) \tau} \ldots + e^{\sim(\mu_3, \mu_C) \tau} \right] + \ldots
\]  

(18)
If, \( \text{loss}_{\text{NN}}^{\text{min}} > \text{loss}_{\text{CC}}^{\text{max}} \), we have \( \frac{\text{loss}_{\text{NN}}}{\text{loss}_{\text{CC}}} > 1 \). Next we show the conditions necessary for establishing this inequality.

As \( 0 \leq \text{sim} \leq 1.0 \), we obtain the \( \text{min} \) using \( \text{sim}_{\text{min}} = 0 \):

\[
\text{loss}_{\text{NN}}^{\text{min}} = \frac{1}{N} \left[ \log (1 + e^0 + e^0 + \ldots + e^0) + \cdots + \log (1 + e^0 + e^0 + \ldots + e^0) \right] = \log [1 + (N - 1)e^0] = \log(N)
\] (19)

Similarly, we can obtain the \( \text{max} \), using \( \text{sim}_{\text{max}} = 1.0 \):

\[
\text{loss}_{\text{CC}}^{\text{max}} = \frac{1}{C} \left[ \log (1 + e^{1.\tau} + e^{1.\tau} + \ldots + e^{1.\tau}) + \cdots + \log (1 + e^{1.\tau} + e^{1.\tau} + \ldots + e^{1.\tau}) \right] = \log [1 + (C - 1)e^\tau]
\] (20)

Combining Equation 19 and Equation 20, we establish the necessary condition for our inequality, Equation 7 as:

\[
\frac{\text{loss}_{\text{NN}}}{\text{loss}_{\text{CC}}} > \frac{\log(N)}{\log [1 + (C - 1)e^\tau]}
\]

This derivation is used in Section 2.1, where we show how the condition is almost always satisfied in real graphs. As a result, \( \text{loss}_{\text{NN}} \) upper bounds \( \text{loss}_{\text{CC}} \). Note that a lower loss is better.
A.2 Hyperparameters vs Accuracy

Figure 6: Ablation study on the hyperparameters. TAU=τ, ALPHA=α, ORDER=R and LR denotes learning rate. A hyperparameter with higher and more condensed distribution represents its superiority over its counterpart. PamCGC is robust to τ, α, R and best with a learning rate of 0.001. Best viewed in colour.

A.3 Hyperparameter behaviour with and without PamC
Figure 7: Comparison of hyperparameters with and without PamC. TAU=τ, ALPHA=α, ORDER=R and LR denotes learning rate. A hyperparameter with higher and more condensed distribution represents its superiority over its counterpart. PamC is generally better in accuracy for majority of the hyperparameter combinations. Best viewed in colour.