Latent Random Steps as Relaxations of Max-Cut, Min-Cut, and More

Sudhanshu Chanpuriya [1] **Cameron Musco** [1]

Abstract

Algorithms for node clustering typically focus on finding homophilous structure in graphs. That is, they find sets of similar nodes with many edges *within*, rather than *across*, the clusters. However, graphs often also exhibit heterophilous structure, as exemplified by (nearly) bipartite and tripartite graphs, where most edges occur across the clusters. Grappling with such structure is typically left to the task of *graph simplification*. We present a probabilistic model based on non-negative matrix factorization which unifies clustering and simplification, and provides a framework for modeling arbitrary graph structure. Our model is based on factorizing the process of taking a random walk on the graph. It permits an unconstrained parametrization, allowing for optimization via simple gradient descent. By relaxing the hard clustering to a soft clustering, our algorithm relaxes potentially hard clustering problems to a tractable ones. We illustrate our algorithm's capabilities on a synthetic graph, as well as simple unsupervised learning tasks involving bipartite and tripartite clustering of orthographic and phonological data.

1. Introduction

A core method of finding structure in networks is mapping nodes to some smaller set of node clusters based on structural similarity. There are various algorithms for this task of node clustering, one of the most well-known being the normalized cuts algorithm (Shi & Malik, 2000), which assigns clusters based on an eigenvector of the normalized graph Laplacian. This algorithm finds a hard clustering, where each node is mapped to exactly one cluster; soft clustering (Yu et al., 2005) relaxes this problem and instead assigns nodes to clusters probabilistically, so that each node is mapped to a categorical distribution over clusters.

[1]University of Massachusetts Amherst. Correspondence to: Sudhanshu Chanpuriya <schanpuriya@umass.edu>.

Published at the Differentiable Almost Everything Workshop of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. July 2023. Copyright 2023 by the author(s).

Clustering algorithms generally try to find groups of nodes that are in line with graph homophily, wherein edges connect nodes with similar attributes, and wedges tend to be closed ("a friend of a friend is a friend"). Only a small number of clustering algorithms can be seen as capturing heterophilous structures, such as (near) bipartiteness: for example, algorithms for the max-cut problem (Grötschel & Pulleyblank, 1981) can find approximately bipartite structure in graphs. Approaches for the closely related task of graph simplification (also called graph compression) are often more amenable than typical clustering approaches to addressing heterophilous structure. Like clustering, simplification is focused on finding structure in graphs, but with the goal of minimizing reconstruction error from a compressed representation. Algorithms for simplification work by merging edges or nodes (Toivonen et al., 2011; Garg & Jaakkola, 2019), or by approximate factorization of the adjacency matrix (Nourbakhsh et al., 2014).

We present a probabilistic framework which unifies node clustering and graph simplification and is applicable to both homophilous and heterophilous structure. In particular, we propose factoring an undirected graph $\boldsymbol{A} \in \mathbb{R}_+^{n \times n}$ into two components: a bipartite graph $\boldsymbol{V} \in \mathbb{R}_+^{n \times m}$, which connects the n original nodes to m latent nodes, where $m < n$, and a smaller undirected graph $\boldsymbol{W} \in \mathbb{R}_+^{m \times m}$, which is a graph on the latent nodes. Intuitively, this factorization approximates taking one step of a random walk on $\boldsymbol{A}$ as a three step procedure: first taking one random step on $\boldsymbol{V}$ from the original nodes to the latent nodes, then one random step within the latent graph $\boldsymbol{W}$, and finally one random step on $\boldsymbol{V}$ back from the latent to the original nodes:

$$\pi(\boldsymbol{A}) \approx \pi(\boldsymbol{V})\,\pi(\boldsymbol{W})\,\pi(\boldsymbol{V}^\top), \tag{1}$$

where π denotes dividing each row of a matrix by its sum, yielding the random walk transition matrix corresponding to the adjacency matrix. Figure 1 illustrates this process. As we discuss in Section 3, this model permits a differentiable parametrization, allowing for fitting via gradient descent on a simple cross-entropy loss. Further, we can ensure that the transition matrix on the right-hand side is reversible, meaning that it corresponds exactly to one step of a random walk on some undirected graph $\boldsymbol{B} \in \mathbb{R}_+^{n \times n}$. Our model allows for retrieval of this $\boldsymbol{B}$ as a rank-m approximation of $\boldsymbol{A}$, connecting this clustering method to graph simplification.

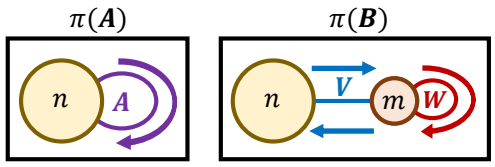

Figure 1. Diagram of the latent random step model. A random step on a graph $\boldsymbol{A}$ with n nodes is approximated by a random step forward on a bipartite graph $\boldsymbol{V}$, then a random step on a smaller graph $\boldsymbol{W}$ with m nodes, then a random step back on $\boldsymbol{V}$.

Demonstrative Toy Graph. We construct a synthetic network that exhibits both homophily and heterophily as a concrete demonstration of how our model can adapt to both. Consider a union of 3 bicliques, each with 10 nodes in either set, for a total of 60 nodes. Figure 2 (left) is a plot of this graph. We can naturally cluster the nodes in at least two ways: either we find 3 homophilous clusters with most edges *within* clusters (as in the min-cut task), or we find 2 heterophilous clusters with most edges *across* clusters (like the max-cut task). As we discuss in Section 4, in our model, this corresponds to fixing one of the following latent graphs $\boldsymbol{W}$, then finding a bipartite graph $\boldsymbol{V}$ that minimizes the approximation error in Equation 1:

$$\boldsymbol{W}_{\text{clique}} = \frac{1}{3}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{W}_{\text{biclique}} = \frac{1}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

In Figure 2 (right), we show the $\pi(\boldsymbol{V})$ matrices, which map each node to a distribution over clusters, that result from both ways of clustering the graph. The one with $\boldsymbol{W}_{\text{clique}}$ indeed assigns each node to three clusters, corresponding to the three bicliques, so that edges occur only within clusters; whereas the one with $\boldsymbol{W}_{\text{biclique}}$ assigns each node to one of two clusters so as to split each biclique, such that edges occur only across clusters. Each method also results in a different simplified graph $\boldsymbol{B}$: the former erases the biclique structure and results in three cliques; whereas the latter erases the distinction between the three disjoint bicliques, resulting in a single large biclique. The latter clustering may be more useful for mining data structure in some applications, but most algorithms only allow for the former.

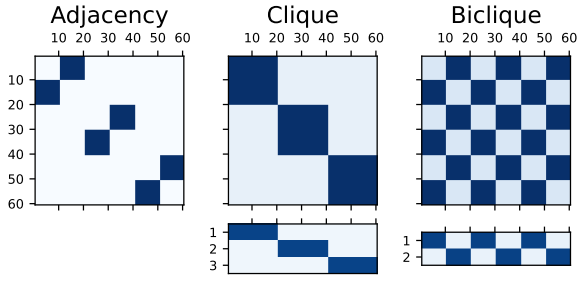

Figure 2. The synthetic graph $\boldsymbol{A}$ (left) and two different clusterings (right), for which we show $\boldsymbol{B}$ (top) and $\pi(\boldsymbol{V})^\top$ (bottom).

Our key contributions can be summarized as follows:

- We present our *latent random step* model. To our knowledge, this is the first probabilistic model for undirected graph simplification that accommodates arbitrary homophilous and heterophilous structure.
- Unlike similar work, our model admits an unconstrained parametrization. Simple gradient descent (and its variants) on a natural probabilistic loss can be used to fit our model, allowing for flexible node clustering and low-rank approximation of a weighted graph. Modifying the latent graph $\boldsymbol{W}$ ranges the type of node clustering task through relaxations of potentially hard problems like k-way max-cut.
- We apply our model and algorithm to real-world data by simplifying weighted graphs constructed from raw orthographic and phonological data. We find that graphs with heterophilous structure naturally arise when considering the sequences of letters and phonemes in words. From these graphs, our unsupervised algorithm finds heterophilous clusters that closely align with ground-truth labels.

2. Related Work

Our model can be represented as an approximate graph factorization of the form $\boldsymbol{A} \approx \boldsymbol{U}\boldsymbol{W}\boldsymbol{U}^\top$, which has also been employed in some prior works. Yu et al. (2005) present such a model for use in soft clustering and also discuss its interpretation in terms of random walks, but they focus on the case where $\boldsymbol{W}$ is diagonal, that is, the homophilous case. Perhaps the closest model to ours is that of Nourbakhsh et al. (2014), who also allow their equivalent of $\boldsymbol{W}$ to be non-diagonal. However, their experiments do not explore non-homophilous clustering, and they do not work within a fully probabilistic framework as we do; among other differences, the loss they optimize is based on Frobenius norm rather than cross-entropy. While these are the two models that are closest to ours, all three models fit under the umbrella of non-negative matrix factorization (NMF) for node clustering and graph simplification, for which there is other prior work (Ding et al., 2008; Kuang et al., 2012).

Our model, along with some of the other discussed models, can be seen as a very generalized variant of the well-known stochastic block model (SBM) (Holland et al., 1983). The key differences are that: 1) nodes in the SBM are assigned to exactly one community, as opposed to our model's distributional assignments; and 2) the central probability matrix in the SBM gives probabilities of nodes in two communities being connected, which is slightly different from our model, wherein the central matrix gives the proportion of edges which occur between two communities; and 3) SBMs, while capable of representing heterphilous structure, are

also typically studied in the context of homophilous structure. As suggested by use of the term 'latent' states, our model can also be seen as an instance of the Hidden Markov Model (Baum & Petrie, 1966) to the process of taking a random walk on a graph; unlike in most applications of HMMs, here the analyzed process is explicitly first-order, by construction. Finally, our fitting algorithm joins much prior work as a relaxation of a computationally-hard node partitioning problem; perhaps best-known is the work of Goemans & Williamson (1995), which gives a spectral relaxation of the max-cut problem. Unlike that work, we provide no theoretical guarantees of performance, though we observe good performance in experiments. On the other hand, our framework can go well beyond max-cut to unify min-cut, k-way max-cut, and more, depending on how the latent graph $\boldsymbol{W}$ is set.

3. Methodology

As stated in Section 1, we propose to approximately factorize an undirected graph $\boldsymbol{A} \in \mathbb{R}_+^{n \times n}$ into a bipartite graph $\boldsymbol{V} \in \mathbb{R}_+^{n \times m}$ and a smaller undirected graph $\boldsymbol{W} \in \mathbb{R}_+^{m \times m}$. We seek

$$\pi(\boldsymbol{A}) \approx \pi(\boldsymbol{V})\, \pi(\boldsymbol{W})\, \pi(\boldsymbol{V}^\top) = \pi(\boldsymbol{B}), \tag{2}$$

where again π denotes dividing each row of a matrix by its sum, yielding a random walk transition matrix. $\boldsymbol{B}$, which is a symmetric matrix in $\mathbb{R}_+^{n \times n}$, is a rank-$m$ reconstruction of $\boldsymbol{A}$ (that is, a simplified version of $\boldsymbol{A}$); like $\boldsymbol{A}$, $\boldsymbol{B}$ can be seen as an undirected, weighted graph on the original n nodes.

Reversibility Criterion. We first establish a condition on $\boldsymbol{V}$ and $\boldsymbol{W}$ for the transition matrix $\pi(\boldsymbol{B})$ from Equation 2 to be reversible, that is, to correspond to a random step in an *undirected* graph $\boldsymbol{B}$. This condition is crucial for fitting the model to not only yield a clustering of the nodes (given by $\pi(\boldsymbol{V})$), but also a simplified graph $\boldsymbol{B}$. Reversibility is satisfied iff there exists a diagonal matrix $\boldsymbol{D_B} \in \mathbb{R}_+^{n \times n}$ for which the product $\boldsymbol{D_B}\, \pi(\boldsymbol{B})$ is a symmetric matrix $\boldsymbol{B}$. We assume that $\boldsymbol{V}$ and $\boldsymbol{W}$ are not just non-negative, but strictly positive; we will only parametrize such $\boldsymbol{V}$ and $\boldsymbol{W}$ anyway.

Let $\boldsymbol{D_V}$ and $\boldsymbol{D'_V}$ denote the diagonal matrices whose diagonal elements are the row-sums and column-sums of $\boldsymbol{V}$, respectively, and let $\boldsymbol{D_W}$ denote the row-sums of $\boldsymbol{W}$ (which are equivalent to the columns-sums since $\boldsymbol{W}$ is symmetric). We have reversibility if, for some $\boldsymbol{D_B}$, the following matrix is symmetric:

$$\begin{aligned}
\boldsymbol{B} &= \boldsymbol{D_B}\, \pi(\boldsymbol{V})\, \pi(\boldsymbol{W})\, \pi(\boldsymbol{V}^\top) \\
&= \boldsymbol{D_B} \left(\boldsymbol{D_V}^{-1} \boldsymbol{V}\right) \left(\boldsymbol{D_W}^{-1} \boldsymbol{W}\right) \left(\boldsymbol{D'_V}^{-1} \boldsymbol{V}^\top\right) \\
&= \boldsymbol{D_B} \boldsymbol{D_V}^{-1} \left(\boldsymbol{V} \boldsymbol{D_W}^{-1} \boldsymbol{W} \boldsymbol{D'_V}^{-1} \boldsymbol{V}^\top\right).
\end{aligned}$$

The transpose of this matrix is

$$\boldsymbol{B}^\top = \left(\boldsymbol{V} \boldsymbol{D'_V}^{-1} \boldsymbol{W} \boldsymbol{D_W}^{-1} \boldsymbol{V}^\top\right) \boldsymbol{D_V}^{-1} \boldsymbol{D_B}.$$

Note that if $\boldsymbol{D'_V} = \boldsymbol{D_W}$, then the parenthesized parts of the final expressions are equivalent. Further, with $\boldsymbol{D_B} = \boldsymbol{D_V}$, the matrix is fully equal to its transpose and is therefore symmetric. Explicitly, the matrix simplifies to the form

$$\boldsymbol{B} = \boldsymbol{V} \boldsymbol{D_W}^{-1} \boldsymbol{W} \boldsymbol{D_W}^{-1} \boldsymbol{V}^\top.$$

Hence reversibility is satisfied if the column-sums of the bipartite graph $\boldsymbol{V}$ are equal to the degrees of the latent graph $\boldsymbol{W}$. If this condition is satisfied, the degrees $\boldsymbol{D_B}$ of the reconstructed graph $\boldsymbol{B}$, which corresponds to the transition matrix $\pi(\boldsymbol{B})$, are exactly the row-sums of $\boldsymbol{V}$.

Parametrization. We can parametrize our model using two matrices of free parameters, $\boldsymbol{W_p} \in \mathbb{R}^{m \times m}$ and $\boldsymbol{V_p} \in \mathbb{R}^{n \times m}$, to represent the latent graph $\boldsymbol{W}$ and the bipartite graph $\boldsymbol{V}$, respectively. Let σ_{mat} and σ_{col} denote functions which take the softmax of a matrix over all elements and over each column. We first construct $\boldsymbol{W}$ as follows:

$$\boldsymbol{W} = \sigma_{\text{mat}}\left(\boldsymbol{W_p} + \boldsymbol{W_p}^\top\right),$$

which ensures both the positivity and the symmetry of W; the softmax also ensures that all entries of $\boldsymbol{W}$ sum to 1. Let $\boldsymbol{D_W}$ be the diagonal matrix containing the degrees of $\boldsymbol{W}$. We now construct $\boldsymbol{V}$ with:

$$\boldsymbol{V} = \left(\sigma_{\text{col}}(\boldsymbol{V_p})\right) \boldsymbol{D_W},$$

which ensures the positivity of $\boldsymbol{V}$ and the reversibility criterion, that the column-sums of $\boldsymbol{V}$ are equal to the degrees $\boldsymbol{W}$. Note that while we provide the full parametrization for generality, in the experiments in this paper, we fix $\boldsymbol{W}$ and find $\boldsymbol{V}$, so $\boldsymbol{W_p}$ is not used.

Fitting. We can fit this model via gradient descent on a simple and natural cross-entropy loss:

$$L = -\sum\nolimits_{i,j \in [n]} \left(\bar{\boldsymbol{A}}_{ij} \log\left(\bar{\boldsymbol{B}}_{ij}\right)\right), \tag{3}$$

where an overline denotes dividing a matrix by the sum of all of its elements. This loss views the adjacency matrix of the original graph $\boldsymbol{A}$ and that of the reconstructed graph $\boldsymbol{B}$ as probability distributions over pairs of nodes. Minimizing it tries to place more mass in $\boldsymbol{B}$ among node pairs which correspond to edges in $\boldsymbol{A}$. We additionally use an L_2 regularization penalty on the parameters.

Our implementation uses PyTorch (Paszke et al., 2019) for automatic differentiation and minimizes the loss using the SciPy (Jones et al., 2001) implementation of the L-BFGS (Liu & Nocedal, 1989; Zhu et al., 1997) algorithm with default hyperparameters. The free parameters are initialized uniformly at random on $(-10^{-2}, +10^{-2})$. The regularization term for the loss is set to 10^{-1} times the mean squared norm of the free parameters.

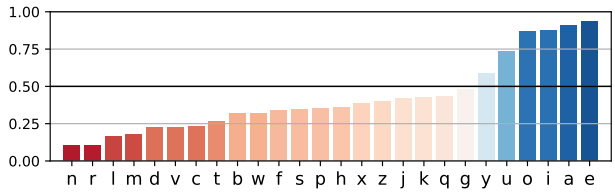

Figure 3. Results of bipartite clustering of the orthographic adjacency graph. For each letter, we plot the probability of assignment to the first of the two clusters, that is, the first column of $\pi(\boldsymbol{V})$. Letters are sorted in ascending order of this probability.

4. Experiments on Real-World Networks

To illustrate the power of our model and algorithm, we perform some experiments on real-world datasets made from English-language orthographic (spelling) and phonological (pronunciation) data. In particular, to find structure in this data in an unsupervised manner, we construct graphs from it and factorize them using the following two latent graphs to find (soft) bipartite and tripartite clusterings:

$$\boldsymbol{W}_{\text{bi}} = \frac{1}{2}\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \qquad \boldsymbol{W}_{\text{tri}} = \frac{1}{6}\begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix}.$$

Factorizing with both latent graphs corresponds to finding clusters such that intra-cluster edges are minimized: with $\boldsymbol{W}_{\text{bi}}$, we find two clusters, and with $\boldsymbol{W}_{\text{tri}}$, we find three clusters such that roughly one-third of the total edge weight is assigned to each of the three pairs of clusters. These clustering tasks can be seen as soft relaxations of the standard and 3-way max-cut problems.

Orthographic Adjacency. We construct a graph based on spellings of common English language words. The nodes of the graph correspond to the 26 letters, and the edge weights correspond to the number of times, across all common words, that two letters are directly adjacent. For the list of words, we use the 20K most frequently-used English words as determined by the Google Books Ngram Viewer [1].

We perform a bipartite clustering of this graph using $\boldsymbol{W}_{\text{bi}}$ based on the intuition that the spelling of words very roughly tends to alternate between vowels and consonants. See Figure 3 for the results. Indeed, the resulting clustering reflects the intuition: if we convert the soft clustering $\pi(\boldsymbol{V})$ into a hard clustering by assigning each letter to the cluster for which it has higher probability, this clustering cleanly divides the letters into vowels and consonants. (The letter 'y', which can act as both, is placed into the vowel cluster, but with the least probability among the vowels.)

Phonological Adjacency. We now construct a graph based on pronunciations of English language words. Using the same list of common words as for the orthographic data, we convert these words to sequences of phonemes as determined by the CMU Pronouncing Dictionary [2]. We use the NLTK API (Bird et al., 2009) to access the dictionary. The graph is similar to the orthographic one: the nodes of this graph correspond to 39 English language phonemes, and the edge weights correspond to the number of times, across all common words in the dictionary, that two phonemes are directly adjacent.

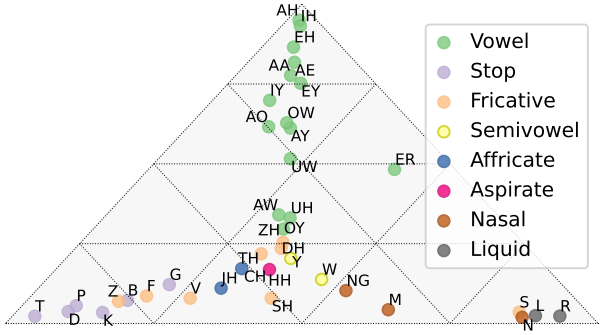

Figure 4. Results of tripartite clustering of the phonological adjacency graph. This ternary plot projects the 3D categorical distributions given by the cluster affinities $\pi(\boldsymbol{V})$ onto a 2D space. Each corner corresponds to a different cluster.

While applying a bipartite node clustering to phonological data also separates vowel sounds as with the orthographic data, we find that significantly more interesting structure can be extracted with a tripartite clustering, using the latent graph $\boldsymbol{W}_{\text{tri}}$. See Figure 4 for a plot of the resulting cluster affinities $\pi(\boldsymbol{V})$. The clustering strongly reflects ground-truth categorizations of the phonemes. Most striking is that one of the clusters is dominated by vowel sounds, and that vowel, stop, and nasal/liquid sounds have high affinities for three distinct clusters.

5. Conclusion

We propose our latent random step model and perform node clustering on a synthetic graph and real-world orthographic and phonological graphs, finding structure in the graphs that goes beyond typical homophilous clusterings. The simplicity and flexibility of the model suggests several directions for extension of this work. We focus here on the setting where the latent graph $\boldsymbol{W}$ is fixed and the bipartite graph $\boldsymbol{V}$ is fit. We could instead attempt to fit both at once, yielding the full graph simplification setting. Besides this, there may be a straightforward extension to data and graphs of greater scale than we consider here: the cross-entropy loss (Equation 3) could be approximated by sampling of node pairs based on the weights of $\bar{\boldsymbol{A}}$, allowing for an SGD fitting algorithm. More broadly, we hope that considering latent graphs beyond homophilous clusters can expand the applicability of node clustering, out to new problems and fields.

[1] Specifically, we use the list on the google-10000-english repo.

[2] This resource is hosted online at the speech.cs.cmu website.

References

Baum, L. E. and Petrie, T. Statistical inference for probabilistic functions of finite state markov chains. *The annals of mathematical statistics*, 37(6):1554–1563, 1966.

Bird, S., Klein, E., and Loper, E. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

Ding, C., Li, T., and Jordan, M. I. Nonnegative matrix factorization for combinatorial optimization: Spectral clustering, graph matching, and clique finding. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 183–192. IEEE, 2008.

Garg, V. and Jaakkola, T. Solving graph compression via optimal transport. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Goemans, M. X. and Williamson, D. P. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

Grötschel, M. and Pulleyblank, W. R. Weakly bipartite graphs and the max-cut problem. *Operations research letters*, 1(1):23–27, 1981.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

Jones, E., Oliphant, T., Peterson, P., et al. SciPy: Open source scientific tools for Python, 2001. URL http://www.scipy.org/.

Kuang, D., Ding, C., and Park, H. Symmetric nonnegative matrix factorization for graph clustering. In *Proceedings of the 2012 SIAM international conference on data mining*, pp. 106–117. SIAM, 2012.

Liu, D. C. and Nocedal, J. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, 1989.

Nourbakhsh, F., Bulo, S. R., and Pelillo, M. A matrix factorization approach to graph compression. In *2014 22nd International Conference on Pattern Recognition*, pp. 76–81. IEEE, 2014.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In *2019*, pp. 8024–8035. 2019.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Toivonen, H., Zhou, F., Hartikainen, A., and Hinkka, A. Compression of weighted graphs. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 965–973, 2011.

Yu, K., Yu, S., and Tresp, V. Soft clustering on graphs. In *Nips*, pp. 1553–1560, 2005.

Zhu, C., Byrd, R. H., Lu, P., and Nocedal, J. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)*, 23(4):550–560, 1997.