RESUMEN

Aquí va el resumen...

Abstract

Aquí inicia el abstract...

AGRADECIMIENTOS

Aquí van los agradecimientos...

CONTENTS

List of Figures

List of Tables

CHAPTER 1

INTRODUCTION

El capítulo 1 inicia aquí...

1.1 Presentation

1.2 Objectives

1.2.1 General objective

1.2.2 Particular objective

1.3 Justification

1.4 Limitations and delimitations of the project

1.5 Research Problem

1.6 Hypothesis

1.7 Project organization

CHAPTER 2

THEORY AND CONCEPTUAL FRAMEWORK

2.1 Preliminaries

Description of concepts

Definition 2.1.1. *orthogonal complement*

Theorem 2.1.1. *For every $n \times n$ symmetric real matrix, the eigenvalues are real and the eigenvectors can be chosen real and orthonormal.*

Theorem 2.1.2 (Courant-Fisher Formula)**.** *Let A be an $n \times n$ real symmetric matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ and corresponding eigenvectors $v_1, v_2, ..., v_n$. Then*

$$\lambda_1 = \min_{\|x\|=1} x^T Ax = \min_{x \neq 0} \frac{x^T Ax}{x^T x},$$

$$\lambda_2 = \min_{\substack{\|x\|=1 \\ x \perp v_1}} x^T Ax = \min_{x \neq 0} \frac{x^T Ax}{x^T x},$$

$$\lambda_n = \lambda_{max} = \max_{\substack{\|x\|=1 \\ x \perp v_1}} x^T Ax = \max_{\substack{x \neq 0 \\ x \perp v_1}} \frac{x^T Ax}{x^T x}.$$

In general, for $1 \leq k \leq n$, let S_k denote the span of $v_1, v_2, ..., v_k$ (with $S_0 = \{0\}$). Then

$$\lambda_k = \min_{\substack{\|x\|=1 \\ x \in S_{k-1}^{\perp}}} x^T Ax = \min_{\substack{x \neq 0 \\ x \in S_{k-1}^{\perp}}} \frac{x^T Ax}{x^T x}.$$

2.2 Graphs and Laplacian Matrices

For the rest of the chapter, let $G = (V, E)$ be a graph, where $V = \{v_1, v_2, ..., v_n\}$ is the non-empty set of nodes (or vertices) and E is the set of edges, composed by pairs of the form (v_i, v_j), where $v_i, v_j \in V$.

It is assumed that all graphs are undirected, meaning that if $(v_i, v_j) \in E$, then $(v_j, v_i) \in E$, for every $v_i, v_j \in V$. For that reason, the edge (v_i, v_j) will be represented as the unordered set $\{v_i, v_j\}$.

A convenient way to represent a graph is through an *adjacency matrix* $A \in \mathbb{R}^{|V|\times|V|}$. Giving a specific order to the graph nodes, one can represent the edges as binary entries in this matrix:

$$A[v_i, v_j] = \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

Let $\mathcal{N}(v_i)$ denote the neighborhood of node v_i, i.e., the set of the adjacent nodes to it. The quantity that represents the number of nodes in $\mathcal{N}(v_i)$ is called the *degree of the vertex* v_i. This is one of the most obvious and informative feature for the structure of the graph and is denoted by

$$d_i = \sum_{j=1}^{n} A[v_j, v_i].$$

Finally, we can summarize that graph's information in the *degree matrix* D which is defined as the diagonal matrix with the degrees $d_1, d_2, ..., d_n$ on the diagonal.

2.2.1 The Unnormalized Laplacian

The unnormalized graph *Laplacian matrix* L is defined as

$$L = D - A$$

Proposition 2.2.1 (Some properties of L)**.** *The matrix L, as defined above, satisfies the following properties:*

1. *For every vector $x = (x_1, x_2, ..., x_n) \in \mathbb{R}$ we have*

$$x^T L x = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(x_i - x_j\right)^2$$

2. *L is symmetric and positive semi-definite*

3. *L has n non-negative, real-valued, eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$*

4. *The smallest eigenvalue of L is* 0*, the corresponding eigenvector is the constant one vector* $\mathbb{1}$*.*

2.2.2 Normalized Laplacians

The *symmetric normalized Laplacian* matrix L_{sym} is defined as

$$L_{sym} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$$

while the *random walk Laplacian* is defined as

$$L_{rw} = D^{-1} L$$

2.2.3 The graph partitioning problem

In order to introduce the graph partitioning problem in its different settings, the mathematical definition of the concepts involved are presented below.

Definition 2.2.1. *Given a graph $G = (V, E)$ and an integer K, a partition of G is a collection of K subsets $P_1, P_2, ..., P_K \subset V$ such that:*

1. $P_i \cap P_j = \emptyset$ *for* $i \neq j$*, where* $i, j \in \{1, 2, ..., K\}$
2. $\cup_{k=1}^{K} P_k = V$

A partition of a graph can be seen as simply removing edges from the original graph in such way the obtained partitions are subgraphs. There are many ways a graph can be partitioned into subgraphs and the way it gets done depends completely on the application of interest. However, independently of the problem to solve, the objective relies on minimizing the connections between the partitions in the original graph. The following concepts provides a useful notation to turn the problem into an optimization one.

For a collection $S \subset V$ of vertices, we define the *edge boundary* $\partial(S)$ to consist of all edges in E with exactly one endpoint in S, that is,

$$\partial(S) := \{\{u, v\} \in E \mid u \notin S \text{ and } v \in S\}$$

Now the problem turns into finding a partition $P_1, P_2, ..., P_k$ such that minimizes the *cut value* of the partition, usually called just *cut*, which is defined as

$$\text{CUT}(P_1, P_2, ..., P_K) := \frac{1}{2} \sum_{k=1}^{K} |\partial(P_k)| \tag{2.1}$$

The notion of cut allows to measure the quality of any partition, nevertheless solving the min cut problem ...

For a collection of vertices $S \subset V$ consider the following quantities related to the edges of

$$\text{VOL}(S) := \sum_{v_i \in S} d_i$$

we want to agroupe by similarity so its natural to Cut value of that partition

solve the mincut problem

The next consider two different ways of measuring the size of the partitions

$$\begin{aligned}\text{RATIOCUT}(P_1, P_2, ..., P_K) &:= \frac{1}{2}\sum_{k=1}^{K}\frac{|\partial(P_k)|}{|P_k|} \\ &= \sum_{k=1}^{K}\frac{\text{CUT}(P_k, \overline{P_k})}{|P_k|}\end{aligned}$$

$$\begin{aligned}\text{NORMCUT}(P_1, P_2, ..., P_K) &:= \frac{1}{2}\sum_{k=1}^{K}\frac{|\partial(P_k)|}{\text{VOL}(P_k)} \\ &= \sum_{k=1}^{K}\frac{\text{CUT}(P_k, \overline{P_k})}{\text{VOL}(P_k)}\end{aligned}$$

2.2.4 Spectral partitioning and Normalized Cut

Here will be presented the derivation of the normalized cut as relaxation of the problem to solve the spectral partitioning problem

2.3 Literature review

Mention all the approaches that use Graph Neural Networks and its importance in solving graph related tasks

2.3.1 Graph Convolutional Neural Networks and GraphSAGE

Start with Grpah Convolutional Neural Networks (GCN) and its limitations. Emphasize in how GraphSAGE solve those limitations and how it extends the GCN capabilities

2.3.2 Generalizable Approximate Graph Partitioning (GAP) Framework

From all the Deep Learning approaches that solve the Graph Partitioning problem, one of the most notorious, not only for its simplicity but for the exceptional results, is the *Generalizable Approximate Graph Partitioning* framework better known as GAP [2].

GAP is a Graph Neural Network approach that propose a continuous relaxation of the problem using a differentiable loss function that is based on the normalized cut. According to Nazi et al. [2], it is an unsupervised learning algorithm that is capable of generalization, meaning that it can be trained in small graphs and its capable of generalize into unseen much larger ones. This section describes the model described by them in the original paper which consists of two modules: the Graph Embedding Module and the Graph Partitioning Module.

The framework takes a graph $G = (V, E)$, where $V = \{v_1, v_2, ..., v_n\}$, as input and outputs the probabilities tensor $Y \in \mathbb{R}^{n \times K}$, where Y_{ik} represents the probability that node v_i belongs to partition P_k.

2.3.2.1 Expected Normalized Cut Loss Function

Recall the normalized cut given by

$$\text{NORMCUT}(P_1, P_2, ..., P_K) = \sum_{k=1}^{K} \frac{\text{CUT}(P_k, \overline{P_k})}{\text{VOL}(P_k)} \tag{2.2}$$

In order to calculate the normalized cut expected value, one need compute the expected value of $\text{CUT}(P_k, \overline{P_k})$ and $\text{VOL}(P_k)$ from Equation 2.2. For the deduction of those quantities, it will be followed an approach similar to the one presented in [1].

Since Y_{ik} represents the probability that node $v_i \in P_k$, $1 - Y_{ik}$ is the probability that $v_i \notin P_k$, and hence

$$\mathbb{E}[\text{CUT}(P_k, \overline{P_k})] = \sum_{i=1}^{|V|} \sum_{v_j \in \mathcal{N}(v_i)} Y_{ik}(1 - Y_{jk}) \tag{2.3}$$

Due to the fact that for a given node, the adjacent nodes can be retrieved from the adjacency matrix A, Equation 2.3 can be rewritten as follows:

$$\mathbb{E}[\text{CUT}(P_k, \overline{P_k})] = \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} Y_{ik}(1 - Y_{kj}^T) A_{ij} \tag{2.4}$$

Keeping in mind that $\text{VOL}(P_k)$ is the sum over the degrees, define Δ to be a column tensor where Δ_i is the degree of the node $v_i \in V$. Then, given Y one can calculate the expected

value of $\text{VOL}(P_k)$ as follows:

$$\begin{aligned} \Gamma &= Y^T \Delta \\ \mathbb{E}[\text{VOL}(P_k)] &= \Gamma_k \end{aligned} \tag{2.5}$$

From the results obtained in Equation 2.4 and Equation 2.5, it is obtained a way to calculate the expected value of $\text{NORMCUT}(P_1, P_2, ..., P_K)$, obtaining

$$\mathbb{E}[\text{NORMCUT}(P_1, P_2, ..., P_K)] = \sum_{k=1}^{K} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \frac{Y_{ik}(1 - Y_{kj}^T) A_{ij}}{\Gamma_k} \tag{2.6}$$

Nazi et al. [2], also showed that given the probability tensor Y one can evaluate how balanced those partitions. Note that the sum of the columns in Y is the expected number of nodes in each partition, i.e., $\mathbb{E}[|P_k|] = \sum_{i=1}^{|V|} Y_{ik}$. In the other hand, in order to have balanced partitions, the number of nodes in each one should be $\frac{|V|}{K}$. As a consequence, the quantity $\left| \sum_{i=1}^{|V|} Y_{ik} - \frac{|V|}{K} \right|$ measures how balanced the partition P_k is.

Using the last result, replacing the absolute value by the squared function, together with Equation 2.6, one can derive the loss function used in GAP that minimizes the expected value of the normalized cut at the same time that intends to balance the cardinalities of the partitions:

$$\mathscr{L} = \sum_{k=1}^{K} \sum_{i=1}^{|V|} \sum_{j=1}^{|V|} \frac{Y_{ik}(1 - Y_{kj}^T) A_{ij}}{\Gamma_k} + \sum_{k=1}^{K} \left(\sum_{i=1}^{|V|} Y_{ik} - \frac{|V|}{K} \right)^2 \tag{2.7}$$

2.3.2.2 Embedding Module

2.3.2.3 Partitioning Module

2.3.3 PinSAGE and Markov Chain Negative Sampling (MCNS)

CHAPTER 3

PROPOSED SOLUTION (GRAPH PARTITIONING FOR LARGE GRAPHS)

CHAPTER 4

EXPERIMENTAL RESULTS

The number of partitions was set to three

For the dataset used to train and test the algorithm "The Graph Partitioning Archive" [3]

CHAPTER 5

CONCLUSION

Las conclusiones y el trabajo a futuro inicia aquí...

5.1 Contributions

5.2 Recommendations and future work

APPENDIX

ANALISIS

El apéndice inicia aquí.

BIBLIOGRAPHY

[1] Alice Gatti, Zhixiong Hu, Tess Smidt, Esmond G. Ng, and Pieter Ghysels.
Deep Learning and Spectral Embedding for Graph Partitioning, pages 25–36.

[2] Azade Nazi, Will Hang, Anna Goldie, Sujith Ravi, and Azalia Mirhoseini.
Gap: Generalizable approximate graph partitioning framework.
ArXiv, abs/1903.00614, 2019.

[3] Alan Soper, Chris Walshaw, and Mark Cross.
A combined evolutionary search and multilevel optimisation approach to graph-partitioning.
Journal of Global Optimization, 29:225–241, 06 2004.