Universität der Bundeswehr München

Bewertung von Community Detection Algorithmen mit der CDlib-Bibliothek

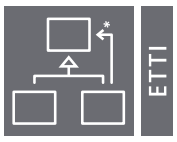
Bachelorarbeit
zur Erlangung des akademischen Grades
Bachelor of Engineering (B.Eng.)

Markus Konietzka

Betreuerin:
Prof. Dr. rer. nat. Andrea Baumann

Tag der Abgabe: 30.06.2023

eingereicht bei
Universität der Bundeswehr München
Fakultät für Elektrotechnik und Technische Informatik



Universität der Bundeswehr München
Institut für
Software Engineering

Neubiberg, Juni 2023

Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst, noch nicht anderweitig für Prüfungszwecke vorgelegt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe, insbesondere keine anderen als die angegebenen Informationen.

Der Speicherung meiner Bachelorarbeit zum Zweck der Plagiatsprüfung stimme ich zu. Ich versichere, dass die elektronische Version mit der gedruckten Version inhaltlich übereinstimmt.

Neubiberg, den 30.06.2023

_____

Markus Konietzka

Zusammenfassung

Dieses Template soll als Formatvorlage für Bachelorarbeiten im Institut Etti6 dienen. Studierend können mithilfe dieser Vorlage, durch Ändern einiger weniger Variablen, eine konsistente Formatierung ihrer Abschlussarbeiten erreichen. Die zu verändernden Variablen sind im LaTeX-Quellcode entsprechend durch Kommentare markiert. Grundlegende LaTeX-Kenntnisse werden dennoch vorausgesetzt.

Inhaltsverzeichnis

1 Anleitung

Diese Anleitung bietet einen Überblick über das Template für Bachelorarbeiten im Institut etti 6, und ist selbst ein Beispiel für die Anwendung Dieser.

1.1 Titelblatt und Erklärung

Das Titelblatt und die Erklärung sind bereits erstellt, und die Quelldateien müssen nicht mehr angepasst werden. Die Belegung der Variablen für Name, Datum, etc. findet im Einstiegspunkt, der Datei thesis.tex statt. Die zu ändernden Variablen sind durch Kommentare markiert.

1.2 Abstract

Standardmäßig wird in diesem Template ein Abstract erzeugt. Sollte dies nicht gewünscht sein, kann die Erzeugung des Abstract in der Datei thesis.tex durch auskommentieren deaktiviert werden. Der Text des Abstract kann in der Datei source/002_abstract.tex angepasst werden.

1.3 Inhaltsverzeichnis

Das Inhaltsverzeichnis wird automatisch erzeugt. Die Tiefe kann in der Datei thesis.tex angepasst werden.

1.4 Hauptteil

Es bietet sich an, für jedes Kapitel eine eigene .tex-Datei im source-Ordner zu erzeugen. Um diese einzubinden, müssen sie in der Datei thesis.tex referenziert werden.

Abbildung 1.1: Beispielabbildung

1.5 Anhang

1.5.1 Abbildungen

Das Abbildungsverzeichnis wird automatisch erstellt, als Beispiel hierfür dient die oben gezeigte Beispielabbildung. Dies kann verhindert werden, indem die entsprechende Anweisung in der Datei 801_appendix.tex auskommentiert wird.

1.5.2 Abkürzungen und Glossar

Abkürzungen wie etwa BA und Erläuterungen wie Latex-Template können in der Datei acronym.tex ergänzt werden. Die entsprechenden Verzeichnisse im Anhang werden automatisch erzeugt.

1.5.3 Quellcode

```
export const writeThesis = (literature, template) => {
    var thesis;
    while(!literature.isUnderstood){
        literature.openAgain();
    }
    template.fill();
    thesis = template.toPDF();
    return thesis;
}
```

Quellcode 1.1: thesis.js

1.5.4 Literaturverzeichnis

Das Literaturverzeichnis wird automatisch erstellt. Dafür wird eine Literaturdatenbank (Dateiformat .bib) benötigt. Empfehlenswert hierfür sind die Programme Zitavi (Windows) oder Zotero (MacOs / Linux). Aus diesen können diese erzeugten Bibliotheken exportiert, und in die oberste Ebene der Ordnerstruktur des Templates unter dem Namen Literatur.bib eingefügt werden. Zitate, die im Text vorkommen, werden automatisch in das Literaturverzeichnis übernommen. Kennzeichnen von indirekten Zitaten [**noauthor_software_nodate**], und "direkten Zitaten"[**noauthor_software_nodate**] erzeugt somit dem IEEE-Zitierstil entsprechenden Zitate.

1.5.5 Tabellenverzeichnis

Das Tabellenverzeichnis wird automatosch erstellt, wie die Folgende Beispieltablle zeigt:

a	b	c	d
1	6	11	16
2	7	12	17
3	8	13	18
4	9	14	19
5	10	15	20

Tabelle 1.1: Beispieltabelle

2 Einleitung

In der modernen Zeit der Digitalisierung spielt die effiziente und effektive Verarbeitung von Daten eine entscheidende Rolle. Dabei stellt sich auch die Frage, wie Datensätze am besten dargestellt werden können, um die richtigen Informationen leicht erkennbar zu machen. Neben den bekannten Formen wie Balken-, Kreis- oder Liniendiagrammen besteht in der Informatik auch die Möglichkeit, Datensätze in Form eines Graphen zu präsentieren.

2.1 Hintergrund und Motivation

In einer Projektarbeit, welcher dieser Bachelorarbeit vorangegangen ist, habe ich mich mit der Aufgabe beschäftigt kleine Gruppen von Profifussballspielern in einem kompletten Graph zu entdecken, welche häufig zusammen den Verein wechseln. Bei dieser Aufgabe wurden qualitativ leider keine guten Ergebnisse generiert, worauf hin sich folgende Fragestellung entwickelte, mit der ich mich in dieser wissenschaftlichen Arbeit beschäftigen will. Wie können Community Detection Algorithmen effektiv und effizient angewendet werden um optimale Ergebnisse zu erzielen, oder gibt es eine Möglichkeit die gefunden Gruppen in Netzwerken mit einer Eavulation zu Bewerten um eine Aussage über die Qualität und Einteilung der Gruppen zu machen.

2.2 Ziele

Das Hauptziel dieser Bachelorarbeit ist es, eine umfassende Bewertung von Community Detection Algorithmen mithilfe der CDlib-Bibliothek durchzuführen. Dabei liegt der Fokus auf der Analyse und Vergleich der Leistungsfähigkeit verschiedener Algorithmen bei der Erkennung von Gemeinschaftsstrukturen in komplexen Netzwerken. Die Arbeit strebt an, Erkenntnisse über die Stärken und Schwächen der Algorithmen zu gewinnen und ihre Anwendbarkeit in verschiedenen Domänen zu untersuchen. Darüber hinaus sollen konkrete Anwendungsbeispiele der CDlib-Bibliothek vorgestellt werden, um ihre Wirksamkeit in realen Szenarien zu demonstrieren. Das langfristige Ziel ist es, zur Weiterentwicklung und Verbesserung von Community Detection Techniken beizutragen und eine solide Grundlage für zukünftige Forschung und Anwendungen auf diesem Gebiet zu schaffen.
Mit dieser wissenschaftlichen Arbeit verfolge ich das Hauptziel eine qualitative und möglichst umfangreiche Bewertung von Community Detection Algorithmen durchzuführen. Für Arbeit der Evaluation stütze ich mich auf die Funktionalitäten der CDlib-Bibliothek. Die Bewertung soll sich dabei vorallem auf die Leistungsfähigkeit der Techniken zur Gruppenerkennung stützen. Um die Ergebnisse gut miteinander vergleichen zu können werden die Verschiedenen Algorithmen an unterschiedlichen und künstlich als

auch realen Datensätzen getested. Durch diese Bewertungen soll es möglich sein die Einsatzgebiete der Algorithmen noch stärker einzugrenzen.

3 Grundlagen

Im weiteren Verlauf dieser schriftlichen Arbeit, versuche tiefergehende Themen und Strukturen der Netzwerk- bzw. Graphentheorie schrittweise zu erklären. Allerdings setze ich Grundkenntnisse der Informatik und ein gutes Verständnis der Datenstrukturen von Graphen voraus.
Die Analyse von Netzwerken entwickelt sehr schnell zu einer komplexen und unter Umständen zeitaufwendigen Aufgabe. Aufgrund der rasant ansteigenden Menge an Daten die dazu führen, das Netzwerke immer umfangreicher werden ist es nicht nur ein Zeitproblem. Mit der heutigen Zeit und der Möglichkeit auch immer mehr Details in Netzwerkgraphen mit zu berücksichtigen gibt es eine vielzahl an Techniken die angewendet werden können.

3.1 Community Detection

Die Erkennung von Gruppen in Netzwerken wird dazu verwendet Graphen zu analysieren oder zusammenhängende Information zu extrahieren. Je nach Zielsetzung kann es auch als eine Art der Klassifiezierung verwendet werden. Doch wie bei vielen anderen Themen gibt es auch bei der Community Detection ein Problem das bewältigt werden muss. Es kann nicht einfach eine deterministische Vorgehensweise angewendet werden, denn es gibt keine einheitliche Definition wie Gruppen erkannt werden können. Für Communityies dagegen gibt es zumindest klarere Regeln wie diese definiert werden können. So zeichnet sich eine Gruppe durch die kompakte und gegenseitige Bindung, sowie eine klare Abgrenzung nach außen aus. Hierfür lassen sich einige Methriken gut verwenden, welche im entsprechenden Nachfolgendem Kapitel näher erläutert werden
//TODO Definition Community, Verweis auf Methriken.

3.1.1 Definition

DELETE ?? Inhalte können in den Absatz "Community Detection"geschrieben werden.

3.1.2 Methriken

3.1.2.1 Modularity

//TODO read Fortunato 2010

3.1.2.2 Density

//TODO read Zhenping Li et al 2008 Die Dichte (engl. density) ist ein Maß, womit angegeben wird wie stark die Verzweigung in einem Graph oder Teilgraph ist. Hiermit lassen sich schon relativ einfach und

schnell kleine Gruppen definieren. Berechnet wir sie im allgemeinen wie folgt.

$$d = \frac{2m}{n(n-1)}$$

Berechnung der Dichte des Graphen mit m Kanten und n Knoten.
Liegen schon Informationen vor, aus welchen Kanten und Konten eine Community besteht kann natürlich auch hiervon die Dichte dieser bestimmt werden.

$$d_{com} = \frac{2m_{com}}{n_{com}(n_{com}-1)}$$

Auch hier entspricht m den Kanten und n den Knoten.
Darüber hinaus kann auch die Dichte berechnet werden, wie stark eine Gruppe nach außen zum restlichen Graphen verbunden ist.

$$d_{ext} = \frac{2m_{ext}}{n_{com}(n-n_{com})}$$

3.1.2.3 Edge Betweeness

Edge Betweeness ist ein Maß dafür wie wichtig eine Kante ist. Es wird die Anzahl der Durchläufe gemessen.
Edge Betweenness gibt das Verhältnis an, wie oft eine Kante auf einem Kürzesten Weg zwischen zwei Knoten durchlaufen wird.

$$E_B(e) = \sum_{s \neq t \in V} \frac{\sigma_{st}(e)}{\sigma_{st}}$$

Es gilt somit als verlässliche Ausage darüber wie wichtig eine Kante im Graphen ist. Je höher der Wert desto öfter wir die Kante auf der Suche nach dem kürzesten Pfad durch einen Graphen passiert.

3.1.2.4 Degree Centrality

Die Degree Centrality gibt an wie gut ein einzelner Konten innerhalb des Graphes oder seiner Community verzweigt ist.

$$c(n) = \frac{deg(n)}{N-1}$$

3.1.3 Algorithmen

Liste der Algorithmen: Girvan-Newman Random Walk Louvain Label Propagation k-Clique

3.2 NetworkX

NetworkX ist eine Bibliothek für die Scriptsprache Python und bietet umfassende Möglichkeiten Datensätze in Form von Graphen zu visuallisieren. Es werden darüber hinaus auch Funktionen angeboten um künstliche bzw. zufällige Netzwerke zu generieren.

3.3 CDlib - Biblitohek

Die CDlib Bibliothek wird verwendet für die Bearbeitung der Hauptaufgabe. Die Bewertung der Algorithmen. Auch wird sie dazu herangezogen, verschiedene künstliche Datensätze für Graphen zu generieren. Hierbei stüzt sie sich aber hauptsächlich auf bereits fertig implementierte Funktionen aus der vorherig genannten Bibliothek NetworkX. Dennoch unterscheiden sich die grundsätzlichen Funktionen und der Umfang dieser beiden Bibliotheken stark voneinander. Zum Vergleich, sind in CDlib jedoch mehr Algorithmen zur Erkennung von Netzwerken implementiert und darüber hinaus sind auch die einzelnen Bewertungsfunktionen, die für Analyse der Ergebnisse essentiell sind.

4 Durchführung der Bewertung

4.1 Zusammenstellung der Testdaten

4.1.1 Synthetische Daten

4.1.2 Reale Daten

4.2 Bewertungsfunktionen

4.3 Experimente

5 Analyse und Bewertung

5.1 Ergebnisse der Algorithemen und Bewertungsfunktionen

5.2 Vergleich der Ergebnisse

6 Zusammenfassung

6.1 Diskussion der Ergebnisse

6.2 Grenzen der Studie

6.3 Ausblick

Anhang

Abbildungsverzeichnis

Tabellenverzeichnis

Quellcodeverzeichnis