機械学習によるグラフモデル OSS 開発コミュニティ構造の特徴量分析方法の提案と評価

M2017SE003　加藤　聖也
指導教員　青山　幹雄

1　研究背景

近年，OSS(Open Source Software)開発におけるコミュニティである開発者間ネットワークは複雑化している．開発者ネットワークを分析する技術として，開発コミュニティを対象としたリポジトリマイニングが行われているが，時間変化を伴う動的な開発者構造に着目した研究は少ない．また，機械学習分野において，グラフのリンク予測やノードラベリングに使用可能な汎用性のある特徴量獲得方法が提案されている．そこで，本稿ではグラフモデルを用いて OSS 開発コミュニティを動的構造変化の表現ができる進化モデルとして定義する．また，定義したモデルからグラフの分散表現を特徴量として獲得する．これらのアプローチから OSS 開発コミュニティの特性の発見，分析を目的とする．

2　研究課題

本稿では，OSS の大規模な開発データに表れる特性と開発者の行動に着目し，それらの特性の獲得する分析方法について以下の 3 点を研究課題(RQ)とする．

RQ1: グラフモデルは OSS 開発コミュニティの進化構造のモデリングに有効か？

RQ2: OSS コミュニティグラフモデルを用いて，OSS 開発コミュニティの進化構造を分析できるか？

RQ3: 機械学習によって OSS コミュニティグラフモデルから特徴は抽出可能か？

3　関連研究

3.1　OSS 開発コミュニティと進化

OSS 開発コミュニティに参画している開発者を 8 つの役割に分類する提案[7]や中核，非中核的開発者にクラスタリングする提案[2]がされている．ソフトウェアプロジェクトコミュニティの進化は，自然生態系と同様の生物進化に従うとされている[8]．さらに，コミュニケーション構造がコミュニティ進化に影響を及ぼす可能性についての発見がある[12]．

3.2　グラフモデルとグラフデータベース

グラフデータベース(グラフ DB)は，グラフデータモデルを基礎とする DB 管理システムである[10]．グラフデータモデルには，プロパティグラフ，ハイパーグラフ，トリプルなどの複数の異なるモデルが存在する．特に，プロパティグラフは，ノード，エッジ，プロパティによって構成されるグラフ型のデータモデルの一種である．エンティティをノードで表現し，エンティティ間の関係をエッジで表現する[11]．

3.3　機械学習

機械学習とは，明示的にプログラムで支持を与えず，データから自動的にパターンを学習し予測を行う技術[3]である．ニューラルネットワーク，深層学習アルゴリズムを利用するためのフレームワークとして，Google の TensorFlow や Preferred Networks の Chainer などがある．

3.4　表現学習

表現学習とは，教師なし学習や半教師あり学習を行う 1 つの方法であり，データの要素を分散表現として抽象化する手法[1]である．表現学習をネットワーク構造へ適用した手法として，DeepWalk[9]や node2vec[4], graph2vec[6]などが提案されている．これらの手法で計算された分散表現を使うことで，既存の複雑なネットワークのラベル推定や分類タスクを高い精度で実施できることが発見されている．graph2vec はオブジェクト集合の特徴を学習することを目的とした表現学習手法であり，任意のタスクに対してグラフの有向，無向，重みに関係なく適用が可能である．

4　アプローチ

OSS 開発コミュニティを理解するためには大規模な開発データを分析する必要がある．また，コミュニティ内での開発者の振る舞いは，OSS 開発コミュニティに影響を与えていると考えられる．このため，OSS の進退を理解するためには OSS 開発コミュニティをマクロとミクロの視点での分析が必要と考える．私たちは OSS 開発コミュニティを定義した新しいクラスのグラフモデルである SCGM(Software Community Graph Model)を提案している[5]．図 1 に定義したグラフモデルを示す．本研究では，このモデルを用いて表現した OSS 開発コミュニティ構造から特徴量を抽出し分析を行う．以上より本稿のアプローチを示す．

(1) 大規模開発データを分析することを目的に，表現学習によって開発者の行動の分散表現を獲得することで，大規模データの特徴量抽出を可能にする．

(2) 表現学習によって獲得した特徴量から，開発者をクラスタリングすることで開発者の行動が特徴量として表されているかを確認し，特性に基づき分類を行う．

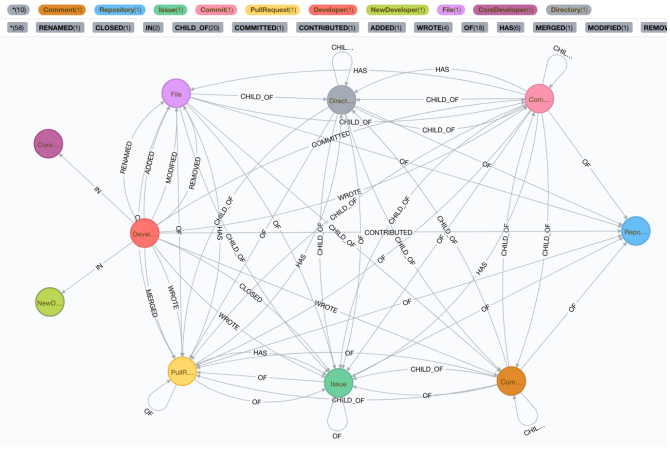

図　1 SCGM(Software Community Graph Model)の定義

4.1 ソフトウェアコミュニティグラフモデルの定義

グラフ DB 上の OSS 開発コミュニティ構造を分析するために，コミュニティ構造を表すグラフモデル SCGM を作成する．本稿では，GitHub から開発データを取得すると仮定している．GitHubから収集したデータを用いて，OSS 開発コミュニティ用の SCGM のインスタンスを作成する．

4.1.1. ノードの定義

ノードは，対象となるオブジェクトを定義する．これは，ノード周りの開発者や関係を分析するために必要となる．SCGM では，Comment, Commit, Developer, Directory, File, Issue, Pull Request, Repository をノードとした．また，開発者ノードは 2 つに分類される．

(a) NewDeveloper
コミュニティへ初めて参画した期間の開発者

(b) CoreDeveloper
中核的な貢献をした期間の開発者．中核的な貢献は，分割期間内で貢献数の上位 10 名までとした．

4.1.2. エッジの定義

エッジは，操作を表す関係とノード間の所属を表す関係の 2 種を定義した．

(a) 操作関係
開発者が行う操作として，ファイルの追加，移動，削除，修正，名前変更や Issue，PullRequest，Comment の作成，終了，Commit の実行を定義している．

(b) 所属関係
IN，OF や HAS など，ノードの親子関係，内包関係を定義している．

4.1.3. プロパティの定義

SCGM の各ノードおよびエッジに対してプロパティを定義し，SCGM 上のプロパティに対するグラフ分析を用いてコミュニティ構造の特性を分析する．任意のノードおよびエッジには，0 個以上のプロパティのセットが関連づけられている．プロパティは name や date などが含まれる．

5 SCGM を用いた機械学習による OSS 開発コミュニティ分析法

5.1 システム構成

以下の図 2 に機械学習を用いた OSS 開発コミュニティ分析システムの構成図を示す．

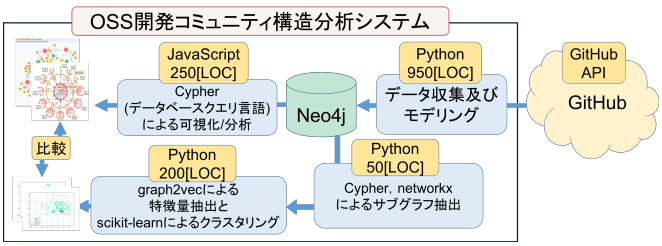

図 2 システム構成図

表 1, 2 にシステムを実現するためのデータ取得，可視化分析の実験環境と機械学習に用いた実験環境を示す．

5.2 OSS 開発コミュニティの特徴量抽出

特徴量抽出プロセスは 2 つのステージからなる(図 3)．

(1) OSS 開発コミュニティグラフモデルの定義
(2) OSS 開発コミュニティ構造特分析プロセス

表 1 データ取得，可視化分析の実験環境

コンポーネント名	名前	Version
OS	OS X High Sierra	10.13.6
グラフデータベース	Neo4j Community Edition	3.5.1
可視化ツール	Neo4j Browser,	3.2.15
可視化，データ取得	JavaScript(Node.js)	v10.15.0
データ取得，変換	Python	v3.6.5
利用した外部 API	GitHub API	v3

表 2 機械学習に用いた実験環境

コンポーネント名	名前	Version
OS	Ubuntu	18.04
実装言語	Python	v3.6.5
フレームワーク	TensorFlow	1.12
データ加工ツール	Pandas	0.20.3
可視化ツール	Matplotlib	2.0.2
計算ライブラリ	scikit-leran	0.20.2

本稿では，(1)は既存研究で作成した SCGM インスタンスを用いて，(2)のグラフの分散表現による特徴量抽出，分析を中心とする．

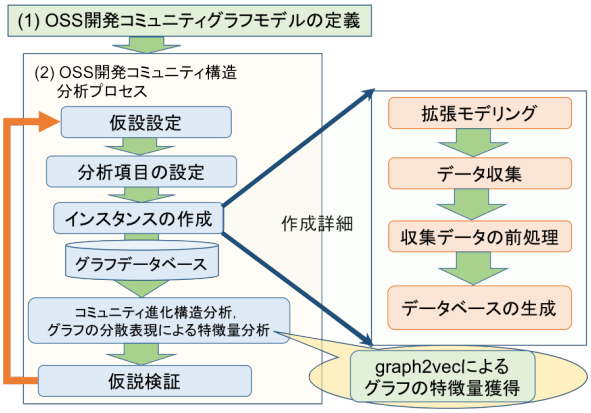

図 3 特徴量分析プロセス

5.2.1. 仮説設定

私たちは OSS 開発コミュニティの特性によってソフトウェア進化が推進される仮説を立てた．設定した仮説は検証プロセスの結果に基づき必要があれば追加，変更する．

5.2.2. 分析項目の設定

設定した仮説に基づいて，仮説ごとに明らかにするべき項目を列挙する．列挙した項目を統合して，分析項目を設定する．

5.2.3. グラフ DB 上で SCGM インスタンスの作成

グラフ DB の作成は，(a)SCGM の特定のコミュニティへの拡張，(b)GitHub からのデータ収集，(c)収集データの前処理，(d)データベースの生成の 4 つのプロセスからなる．

(a) 対象のコミュニティへの SCGM の拡張モデリング
分析対象の OSS 開発コミュニティに SCGM で定義されていない特定のタイプのノードとエッジがある場合は，SCGM を拡張する．

(b) GitHub からのデータ収集
データ収集は主に，OSS 開発のプラットフォームが提供している API を使用する．GitHub 上の OSS リポジトリから GitHub API を利用してデータを収集する．

(c) 分析のための収集データの前処理
収集するデータは，分析項目で設定した内容に沿う必要

がある. 収集したデータには, ノイズデータが存在する可能性がある. そこで, GDB へデータを挿入する前にデータのプルーニングをすることができる.

(d) データベースの生成

GDB Neo4j 上に SCGM のインスタンスを作成し, 前処理されたデータを挿入する.

5.2.4. グラフの分散表現による特徴量抽出

グラフ DB で作成された SCGM のインスタンスに対し, 分散表現獲得手法によって, グラフ上のノードに対する特徴量を抽出する.

5.2.5. 仮説検証

分析で得られた結果から, 設定した仮説を検証する. 必要に応じ, 得られた結果に基づいて, 仮説の追加や変更を行い, はじめからの一連のプロセスを繰り返す.

また, 本稿ではこのプロセスで, 設定された仮説に対する分析が機械学習によって表現できているかを検証する.

以上をふまえ, 本稿での特徴量検証までのプロセスを図 4 に示す. このプロセスをもとに分析, 評価を示す.

6 GitHub 上の OSS 開発コミュニティへの適用と発見

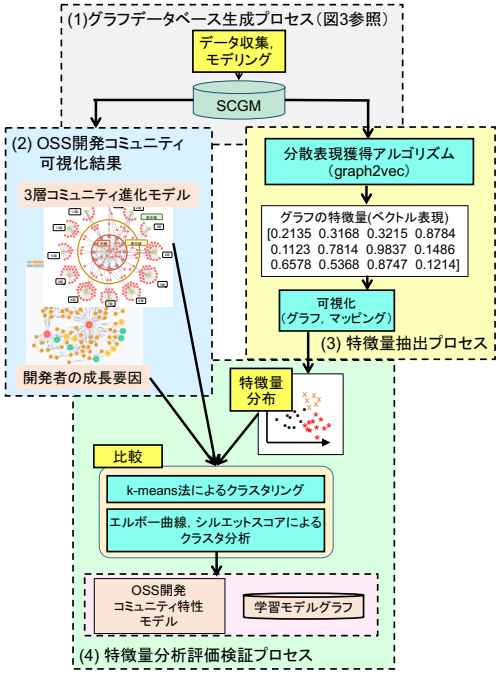

図 4 特徴量分析検証プロセス

6.1 分析対象の OSS 開発コミュニティデータ

OSS 開発コミュニティデータとして, 主要な機械学習 OSS ライブラリを含む 10 種の開発コミュニティ, Chainer, PyTorch, Hadoop などを分析対象とした. 分析対象期間は, 2008 年 1 月 6 日から 2018 年 12 月 1 日までとした. この期間を 3 ヶ月ごとに 1 期, 2 期の順に分割した. また, 3 ヶ月に満たない部分は分析対象外とした.

6.2 コミュニティの代謝構造と開発者の行動特性

コミュニティを期間ごとに区切り, 期間ごとに新規参画した開発者と貢献が多い開発者をサブグラフとして可視化した結果を図 6 に示す. さらに, Chainer 内の開発者 3 人の新規参画時から半年間の行動をサブグラフとして可視化した結果を図7に示す. グラフ上で, 赤ノードは開発者, 薄緑ノードはある期間内に貢献を行った開発者ノードを繋ぐもの, 紫ノードはある期間内で貢献度数の上位 10 名までの開発者ノードを繋ぐもの, 濃緑ノードが Issue, 黄ノードが Pull Request, 橙ノードが Comment となっている.

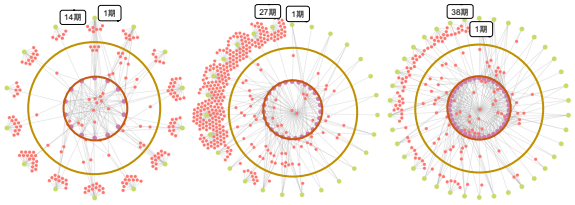

図 5 3 層コミュニティ進化モデル(Chainer, PyTorch, Hadoop)

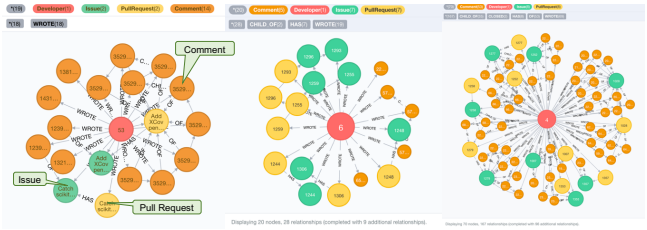

図 6 開発者の行動特性(非中核的, 準中核的, 中核的)

6.2.1. 3 層コミュニティ進化モデル

図 6 より, コミュニティ上の開発者の構造を以下の 3 層で構成されるコミュニティ進化モデルとして定義できる.

(1) 非中核: 短い期間の参画, または貢献が少ない開発者
(2) 準中核: 一定期間のみに高い貢献をする開発者
(3) 中核: 長期間, または高い貢献数を維持している開発者

中核的開発者の構成は, コミュニティ発足時の一部の開発者が主体になっている. この構造はどのコミュニティで見られ, コミュニティが進化しても, 大きな変化はない. 一方, コミュニティ進化に伴い, 一定期間のみ高い貢献を行っている開発者が存在している. 本稿では, これらの開発者は準中核的開発者と呼ぶ. 準中核的開発者は一定期間のみ貢献した後フェードアウトする傾向があることより, 彼らはコミュニティの進化と共に入れ替わっていると考えられる. この現象は本研究で初めて発見された.

6.2.2. 開発者の成長要因

図 7 より開発者の行動は, 以下の 3 層構造のコミュニティ進化モデルに対応付けることができる.

(1) 非中核的開発者: バグ報告や機能提案を中心に活動
(2) 準中核的開発者: 機能実装を中心に活動
(3) 中核的開発者: バグ報告, 機能提案, 実装のいずれも積極的に活動

6.3 開発者行動グラフからの特徴量抽出

6.2 から OSS 開発コミュニティは 3 層の開発者による進化構造を持つことが発見された. そこで, issue や pullrequest, commit などを開発者行動グラフとして graph2vec を用いて表現学習することでグラフの特徴量を抽出した. 抽出する特徴量は 1024 次元の値とした.

6.4 特徴量に基づく開発者クラスタリング

図 7 にクラスタリングおよび主成分分析による次元削減後の結果, シルエット図を示す. 初期クラスタ数は 3 とした. クラスタリングでは, 開発者を丸, 四角, 三角に振り分けている.

(1) 開発者の 3 層構造の特定

図 7 左が k-means 法によるクラスタリングであり, 図 7 中央が主成分分析によって次元削減を行なった後の k-means 法

によるクラスタリングの結果である. 次元削減を行わない場合では, グラフ上に丸, 四角が幅広く分布しており, 各開発者のクラスタが判別しにくい. 一方, 次元削減を行なった場合では, 特に Chainer や Keras, Theano などで少数の開発者が密集しているクラスタが顕著に見られる. これが, 中核的開発者であると考える. また, 多数の開発者が幅広く分布しているクラスタと, そこから徐々に離れている開発者が存在する. これらが, 3 層構造における非中核的開発者と準中核的開発者と考える.

(2) クラスタ数の妥当性

図 4 右が各コミュニティのシルエット図である. 全体的に, 開発者がかたまって存在しているクラスタのみがシルエットスコアが高く算出されている. PyTorch 以外のコミュニティでは残り 2 つのクラスタのスコアは低い. また, エルボー曲線を用いてクラスタ数を 1 から 10 まで算出したが, クラスタ数が決定できる有効な値は得られなかった.

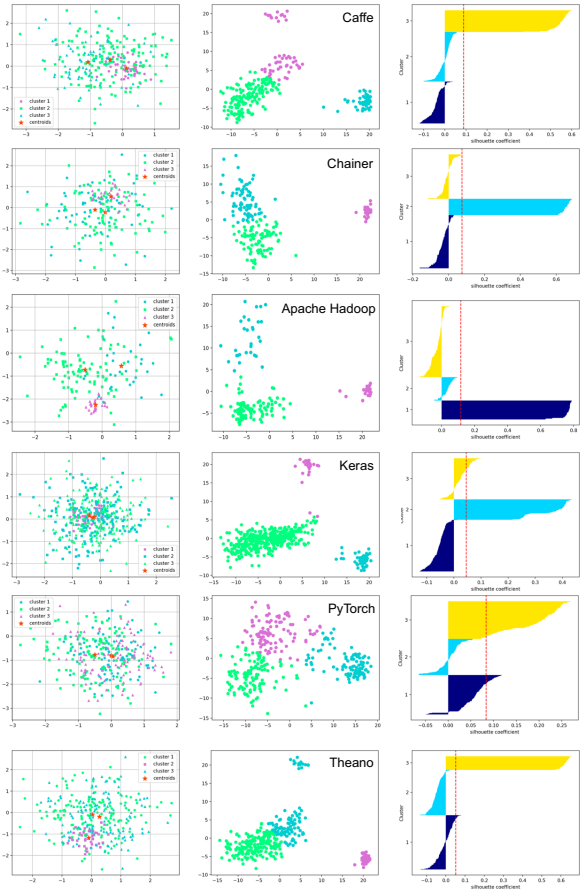

図 7 クラスタリング結果とシルエット図

7 考察

RQ1: グラフモデルは OSS 開発コミュニティの進化構造のモデリングに有効か？

有効である. 私たちは, OSS 開発コミュニティを定義する新しいクラスのグラフモデルである SCGM を提案した. グラフ理論によって裏付けされた SCGM は, OSS 開発コミュニティの新規特性を発見することが可能であった.

RQ2: OSS コミュニティグラフモデルを用いて, OSS 開発コミュニティの進化構造を分析できるか？

可能である. 既存研究に比べ, SCGM によるコミュニティ構造の時間変化のスナップショットを示し, 動的変化を可視化することで, 以下の OSS 開発コミュニティの動的な構造を明らかにした.

(1) 非中核, 準中核, 中核的開発者の 3 層コミュニティ進化モデル

(2) 開発者の行動による成長パターン

(3) 開発段階による開発者間の相互作用の変化

RQ3: 機械学習によって OSS コミュニティグラフモデルから特徴は抽出可能か？

可能である. 表現学習を用いて開発者行動グラフから抽出した特徴量から 3 層構造に分類が可能であることを発見した. この結果は, 表現学習によって SCGM の特徴を抽出できることを示唆している. また, シルエットスコアによって OSS 開発コミュニティ構造の状態判定器としてグラフの特徴量を利用できる可能性があると考える.

8 今後の課題

(1) 機械学習モデルの評価方法と妥当性検証
(2) 分類精度の改善
(3) 開発コミュニティの属性の検証
(4) グラフモデルの改善

9 まとめ

本稿では, OSS 開発コミュニティを定義する新しいクラスのグラフモデルである SCGM(Software Community Graph Model) を提案した. さらに, SCGM に基づく OSS 開発コミュニティの進化構造分析方法と, SCGM の表現学習による OSS 開発コミュニティ構造の特徴量分析方法を提案した. 分析対象として GitHub 上に存在する 10 種の OSS 開発コミュニティ対して適用した. このモデルを構造分析することで, コミュニティ進化の特性を発見した. さらに, 開発者行動グラフから特徴量を抽出し, 開発者の 3 層分類構造を確認した.

提案した SCGM や分析手法を用いることで, OSS コミュニティの進化理解へのサポートや OSS 運用や開発者の成長の支援への活用が期待できる.

参考文献

[1] 浅谷 公威, ネットワークの表現学習, 人工知能学会誌, Vol.31, No.4, pp. 587-593, 2016.
[2] A. Bosu, et al., Impact of Developer Reputation on Code Review Outcomes in OSS Projects: An Empirical Investigation, Proc. of ESEM '14, Article No. 33, ACM, Sep. 2014, 10 pages.
[3] I. Goodfellow, et al., Deep Learning, MIT Press, 2016.
[4] A. Grover, et al., node2vec: Scalable Feature Learning for Networks, Proc. of KDD 2016.
[5] S. Kato, et al., A Structural Analysis Method of OSS Development Community Evolution Based on A Semantic Graph Model, Proc. of COMPSAC 2018, IEEE, Jul. 23-27, 2018, pp. 292-297.
[6] A. Narayanan, et al., graph2vec: Learning Distributed Representations of Graphs, Proc. of MLG 2017.
[7] K. Nakakoji, et al., Evolution Patterns of Open-Source Software Systems and Communities, Proc. of IWPSE '02, ACM, May 2002, pp. 76-85.
[8] T. Mens, et al., Studying Evolving Software Ecosystems based on Ecological Models, T. Mens et al.(eds.), Evolving Software System, Springer, 2014, pp. 297-326.
[9] B. Perozzi, et al., DeepWalk: Online Learning of Social Representations, Proc. of KDD 2014.
[10] I. Robinson, et al., Graph Databases, O'Reilly, 2015.
[11] M. Russell, Mining the Social Web, 2nd ed., O'Reilly, 2014.
[12] M. Zhou, et al., What Make Long Contributors: Willingness Opportunity in OSS Community, Proc. of ICSE 2012, IEEE, Jun. 2012, pp. 518-528.