修士論文

Retweet 行為に基づくネットワーク分析と可視化：Twitter でのゲーム会話の例

1950009 Liu Feifan

主指導教員 林 幸雄

北陸先端科学技術大学院大学
先端科学技術研究科
（融合科学共同専攻）

令和 2 年 2 月

目 次

図 目 次

表 目 次

第1章　序論

1.1　研究の背景

Twitter はゲームの会話とつながりの場を提供しており、会話規模は更に大きくなる。2019 年には、ゲームに関する 12 億件を超えるツイートは、前年と比べで約 20%増加した [1]。さらに、2020 年の上半期では、Twitter のゲーム関連ツイート数は 10 億を超え、過去最高を記録した。上半期のゲームに関するツイート数を見ると (図 1.1)、話題数一位になった日本は、ゲームに関する話題の流行に大きく貢献している [2]。

新型コロナの影響により、自宅で過ごす時間が増えるなか、Twitter 上でゲーム関連の会話の勢いが衰える気配はないと言える。2020 年 3 月後半には、会話量は 71%、retweet 利用者の数は 38%が増え、アメリカ国内では、会話が 89%、 retweet 利用者が 50%増加した [3]。多くのゲーム好きが集まっている Twitter では、外出自粛前と比較し、ゲームをする時間が増えたと回答した Twitter 利用者は 37%おり、Twitter 非利用者と比較し 19 ポイントと大きな差がある [4]。

人々がソーシャルネットワーク上における情報消費、共有、拡散などの色々な行為は、ソーシャルネットワークの研究において常に重要である。ゲームを趣味とする人々の交流地として、話題数が増え続けている Twitter はゲーマーの行為を調査する重要な一部分にもなっている。そこで、テキスト情報に頼らず，人々の行為関係からネットワークを抽出して、ネットワークの特性分析により、情報拡

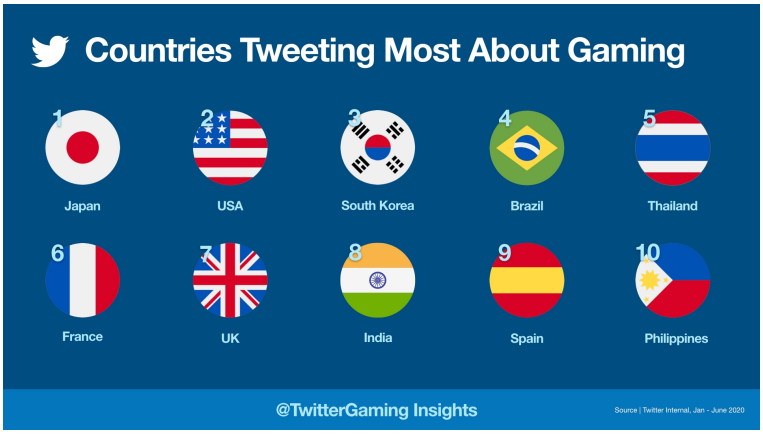


図 1.1: ツイート数が多い国のランキング [1]

図 1.2: 一番話題になっているゲームランキング [1]

散力の強い「インフルエンサー」が見つかれば、ユーザーの好みや政治傾向なども判断できる。例えば、2019 年、Carolina の研究は 2018 年に行ったイタリア大統領選挙前、Twitter 上イタリア語ユーザーの一ヶ月程の話題を収集した。その際、ユーザー間の retweet 関係からネットワークを抽出し、結果的に、政治グループを四つを分けて、ユーザーの政治傾向を予測した [5]。

1.2 研究内容と目的

2019 年に一番話題になっているゲームランキング (図 1.2)[1] を見ると、 上位ゲームの中に日本の会社が開発したゲームは 6 個入り (1 位の fgoproject、3 位の FinalFantasy、5 位の granbluefantasy、6 位の ensemble_stars、7 位の monst_mixi、10 位の Super Smash Bros.)、その中にスマートフォンゲームが多いことも判明している (1 位の fgoproject、5 位の granbluefantasy、6 位の ensemble_stars、7 位の monst_mixi)。また、スマートフォンゲームのランキングは半数を占め (上記の 4 つを含め、4 位の IdentityVJP)、10 年以上歴史あるゲームは 3 つ (3 位の FinalFantasy、9 位の Minecraft、10 位の Super Smash Bros.)、残り 2 つは近年流行りのゲーム種類で (2 位の FortniteGame と 8 位の PUBG)、お互いにファンが競争しているように見受けられる。

本研究では、上記のゲームを seed node(データ収集の始まりの user) として扱い、seed node の 2020 年 1 月から 9 月間発信した tweet を集め、それらの tweet に retweet 行為をしたユーザーを収集して、retweet ネットワークを抽出する。各ゲーム毎月についての発信した tweet に関して、ネットワーク上の変化と、各グループに属しているユーザーのネットワーク特徴を解明するため、retweet ネットワークから tweet 間のネットワークとユーザー間のネットワークを抽出する。

1.3　論文の構成

本論文の構成を以下に示す.

第 2 章はネットワークの基本的な用語と分布指標、Twitter データの収集方法に構成されている。

第 3 章は実験を用いた 3 種のネットワークの抽出法と二部グラフの概念、分析に使った K-core と Louvain 法の解説について説明する。

第 4 章は 8 個の Seednode データを収集して、retweet ネットワークから tweet 間のネットワークを抽出により、各 Seednode の毎月の変化を分析して可視化する。さらに、分類した tweet 間のネットワークからユーザー間のネットワークを抽出して、可視化結果を示す。

第 5 章は本研究の結論をまとめる。

第2章　関連研究

以下、2.1 節は本研究で扱うネットワークの基本的な用語と中心性指標について、2.2 節は Twitter データの収集方法について説明する。その残りは本研究で扱う概念と方法について説明する。

2.1　ネットワークの基本用語と分布指標

2.1.1　基本用語

本研究に対するネットワークは、頂点集合 (ノード)V(G) = {1,2,3,4,..., i, j,...,N-1,N} とノード i,j の間の辺 (リンク) の集合 E(G) = $\{e_{ij}\}$ を用いて、グラフ $G = (V, E)$ で表記される。ノードはすべて同じ種類の対象で構成されているとは限らず、ネットワーク構成対象の変化に応じて変わっていく。本研究の例としては、$User$ ネットワークにおいてはすべでのノードは「$User$」という対象で、$Tweet$ と $User$ のネットワークは一部分のノードは「$Tweet$」という対象で、もう一部分のノードは「$User$」という対象である。リンクはノード間の関係を表し、例えば $Tweet$ と $User$ のネットワークは「$Retweet$」($User$ は特定の $Tweet$ を $Retweet$ する行為) という関係で繋げる。また、リンクの方向性や、重みもある。もしリンクの方向性がない場合は「無向グラフ」と表記され、方向性がある場合は「有向グラフ」と呼ばれる。

2.1.2　分布指標

以下、ネットワークの主な分布指標を説明する。

次数（Degree）

次数とは、1 つのノードが他のノードとつながっているリンクの数である。

無向グラフの場合は、1 つのリンクが両端の 2 つのノードに接続するため、リンクの片側の次数は 1 だけを増加するが、もう一方の次数も 1 だけ増加する。

有向グラフの場合は、もしノード A からノード B を接続すると、ノード B の in-degree は 1 だけ増加し、ノード A の out-degree は 1 だけ増加する。そのため、ネットワークの in-degree と out-degree はリンク総数同じである。

ノードの次数分布から、ネットワーク中のあるノードに次数が集中している範囲が把握できる。各ネットワークの次数分布を比較すると、複数ネットワーク間の区別もできる。

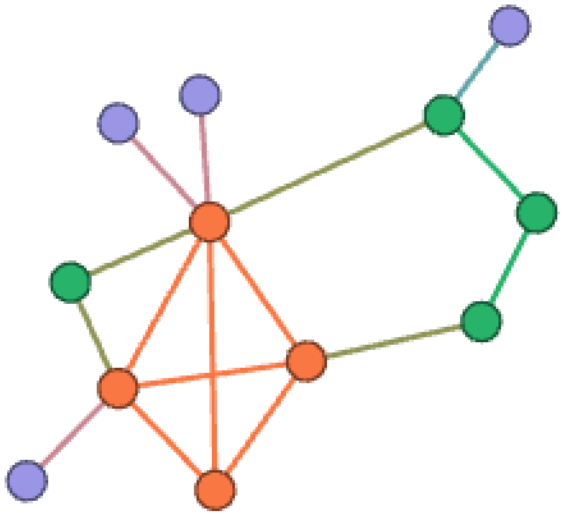


図 2.1: K-shell

K-core/shell

ネットワーク内で密に連結された核として、K-core は少なくとも互いに K 以上のノードに接続されている部分グラフを指す。K-core は階層的に生成される (1-core は 2-core に含まれる)。一方、K-shell の各 shell のノード合がサブセットの関係ではなく、各 shell のノード合は独立している。Kshell の定義は次の式 (2.1) でに従う。

$$Kshell = Kcore - (K-1)core \tag{2.1}$$

図 2.1 の例において、1-shell である次数が 1 のノードを再帰的に削除して、紫色のノードが除去され、ネットワーク内は 2-core である次数 2 以上のノードが残される。次は 2-shell である次数が 2 のノードを再帰的に削除して、緑色のノードを外すと、最後は 3-core である次数が 3 のノードのみが存在する。紫色のノードは 1-shell、緑色のノードは 2-shell、オレンジ色のノードはの 3-shell である。K の値が増加すると、グラフのサイズは小さくなるが、連結度は高まっていく。

ノードの K-core/shell 分布から、各ネットワークの連結度の差がある程度把握できる。連結度の強いノードの数も把握できる。

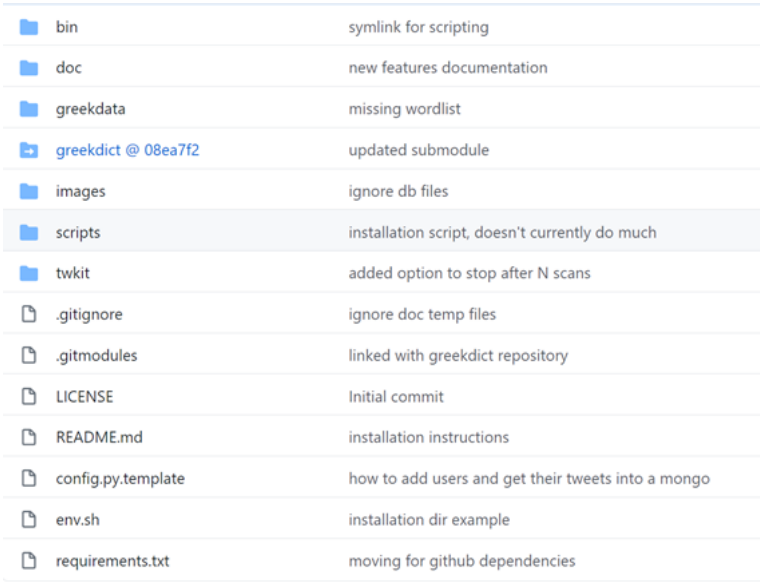


図 2.2: twAwler のスクリプト概要

2.2 Twitter データの収集方法

2.2.1 twAwler の概要

twAwler

twAwler[6] は Polyvios Pratikakis が 2018 年 github に公開した。この「プロジェクト版」では、API を使用して利用可能なすべての tweet を読み込み、tweet を使用してより多くの user を見つけ、それらをフォローして、拡散的な森林火災のサンプリング法でギリシャ語を話す人を探索する。

主に python2.7 と mongodb（バージョン 4.0 以降）に依存しているすなわち、python-twitter パッケージを使用して API を呼び出し、twitter データを収集する。2020 年末、作者は python3 を使い、スクリプトを書き直したが、本研究では python2.7 時の「プロジェクト版」(図 2.2) を使用する。

twAwler の統計分析とソフトウェアロボットとして情報収集する crawler 動作するためのコアスクリプトは twkit フォルダーにある。bin フォルダーは twkit フォルダー内のスクリプトを引用する。Unix の shell 言語を使用して、bin フォルダー内のスクリプトファイルを呼び出す。「env.sh」には主に、ファイル名、python バージョンなど、「プロジェクト版」の実行に必要な環境変数を記録する。「config.py」には、api キー、データベース名、言語などの「プロジェクト版」の基本設定が含まれる。「requirements.txt」には、 この「プロジェクト版」が依存する python パッケージを記載する。 「プロジェクト版」全体の最も重要なコンテンツは twkit フォルダーと scripts フォルダーであるため、ここでは主にこれら 2 つを紹介する。

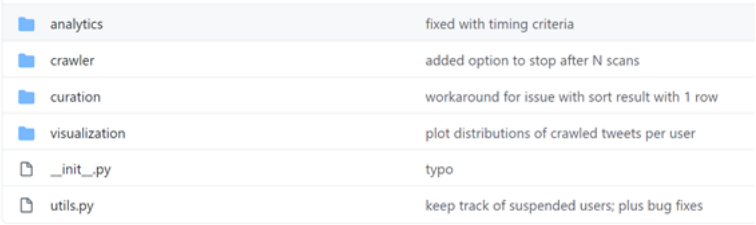


図 2.3: twkit フォルダー

twkit フォルダー

twkit フォルダー (図 2.3) 内の「utils.py」は、「プロジェクト版」全体に必要なものとして、すべての基本的な関数はその内部に保存される。例えば、データの基本処理関数、基本の収集関数、エラー処理など。

1. **Analytics** は、データ分析為のスクリプトで、主にデータベース統計情報を算出して、テキスト分析するための関数もある。

2. **curation** フォルダ内のスクリプトは、主にデータの収集状況を調べる関数を提供し、pymongo を利用してデータベースに検索を実行する。各 user の tweet 数、retweet 数、無効 user 数などの統計を出力する。

3. **visualization** は画像を出力する方法。本研究の可視化ツールは visualization のスクリプトは使用しない。

4. **crawler** フォルダー内のスクリプトはこの「プロジェクト版」のすべてのクロール機能を主に含む、一番重要な部分である。crawler のスクリプトは主に add シリーズ、Count シリーズとコントロールシリーズの 3 種類がある。

- add シリーズのスクリプトは、主に API を利用してデータを収集し、データベースに保存するスクリプトである。

- Count シリーズは、収集状況を確認する為、データベース中の user 数と tweet 数など計算する為に使用される。

- コントロールシリーズは収集をコントロールする為のスクリプト、例えば、クローラーが上限に達したかどうかを確認する「limits.py」、user のクロールを停止する「stop.py」など。

scripts フォルダー

scripts フォルダー内のファイルは、shell 言語を使用し、「プロジェクト版」内の python スクリプトを呼び出す。図 2.4 に示すスクリプトは、さまざまなデータの収集を担当し、start-all.sh はそれらすべてを統合し、操作をログとして出力する。

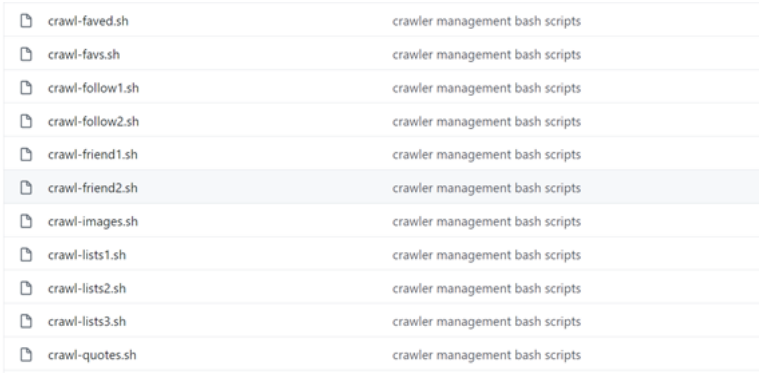


図 2.4: scripts フォルダー

```
if options.ids and len(args) > 100:
  idlist = []
  for idstr in args:
    userid = long(idstr)
    if not can_follow(db, userid, options.refollow): continue
    idlist.append(userid)
    if len(idlist) > 99:
      addedlist = add100_id(db, api, idlist)
      idlist = []
      for u in addedlist:
        add_to_followed(db, u['id'], u['screen_name_lower'], u.get('protected', False))
  if len(idlist):
    addedlist = add100_id(db, api, idlist)
    for u in addedlist:
      add_to_followed(db, u['id'], u['screen_name_lower'], u.get('protected', False))

else:
  for user in userlist:
    if options.ids:
      follow_user(db, api, uid=long(user), wait=True, refollow=options.refollow)
    else:
      follow_user(db, api, uname=user, wait=True, refollow=options.refollow)
```

図 2.5: adduser.py

2.2.2 収集ツールの紹介について

収集ツールについて、クロールツール、データベースツール、分析ツールの 3 つの部分に分けて紹介する。

1. クロールツール

クロールツールの主な目的は、python-twitter を使用してデータ収集用の API を呼び出すことである。 収集されるデータは、主に user データと tweet データの 2 種類に分けられる。そして、「crawler/adduser.py」を使用してユーザー情報を収集する例を次に示す。

図 2.5 のスクリプトは「adduser.py」にインターセプトされ、赤いボックスをマークされた関数はすべて「utils.py」スクリプトから引用するもの、関数は ID または user 名を取得した後に次の操作を実行する。

図 2.6 は userid を add100_id 関数に渡し、次に、api.UsersLookup を介してデー

```
def add100_id(db, api, idlist):
  addedlist = []
  if verbose(): print 'another {}'.format(len(idlist))
  try:
    users = api.UsersLookup(user_id=idlist)
  except twitter.TwitterError as e:
    handle_twitter_error(db, api, e, None, '/users/lookup', None)
    if verbose():
      print 'error, retrying one-by-one'
    map(lambda i: get_if_missing(db, api, i), idlist)
    return []
```

図 2.6: add100_id 関数

タを取得する。api.UsersLookup は twitter-python によって提供された関数である。api.UsersLookup を利用して、指定された userid によって user の情報を取得することができる。

「プロジェクト版」内 tweet の収集と user データ収集の流れは似ており、プロセスは次のとおり：

1. crawler フォルダ内の特定の目的のスクリプトを使用して、収集対象によって、ユーザー ID または tweetID を指定する。

2. utils.py で関数を呼び出す。

3. twitter-python が提供する関数を使用してデータを収集する。

ただし、tweet を収集する GetUserTimeline 関数は「utils.py」にはない。その理由は、フォローや友達など、user を取得するためのチャネルが多数あるが、user の tweet を取得するには、GetUserTimeline を使用するだけで十分である。従って、tweet を収集に関する、「プロジェクト版」で使用したのは「crawler/dumpall.py」および「crawler/load-past.py」のみである。

2. データベースツール

データベースツールの主な機能は、収集したデータを mongodb データベースに保存することである。python パッケージ pymongo は保存する機能だけではなく、データ検索、データ削除など mongodb に関する操作は全部 pymongo に頼る。ここで「adduser.py」の add100_id の例をあげる。

図 2.7 は先に述べた user 収集スクリプトの次の部分を示す。api.UsersLookup を介して user コレクションを取得した後、収集された user 情報は 1 つずつ add_user 関数に渡す。

add_user で収集されたデータをデータに保存できる形にクリニックした後、データが重複しているかどうかを確認し、一部のコンテンツを追加または削減する。

```
for u in users:
  #u1 = user._json
  #u = twitter.User.NewFromJsonDict(u1)
  j = add_user(db, api, u)
  addedlist.append(j)
  idlist.remove(u.id)
for i in idlist:
  if verbose():
    print u'user {} not found, marking dead'.format(i)
  bury_user(db, i)
return addedlist
```

図 2.7: add100_id 関数の次部分

```
db.users.update_one(k, {'$set': j}, upsert=True) #insert if not exact duplicate
if u.protected:
  if verbose(): print(u'protected user {} {}'.format(u.screen_name.lower(), u.id))
  protected(db, u.id)
return j
```

図 2.8: users.update_one 関数

db.users.update_one(図 2.8) を使用して、データベースの users テーブルにデータベースを追加する。

mongodb はデータベースのスキーマ設計に影響を与えることなくデータベースのフィールドを自由に変更できるため、「プロジェクト版」がまだ実行中であっても、必要なカラムを状況に応じて変更できる。例えば、users テーブルに新しく追加された user をマークするために、「addretweeters.py」が変更され、データベースにデータを入るときに action カラムはマークとして、ソースの tweet id を記録する。

tweet は常に大量に収集を行い、収集効率を向上させるため、この「プロジェクト版」では、db.tweets.initialize_unordered_bulk_op(図 2.9) を使用してデータベース挿入インターフェイスを初期化した後、pack_tweet 関数を使用して収集したデータ形式を処理して、最後 insert を使用して大量のデータをデータベースに挿入する。収集したデータの中に、取得した tweet データの規模は user データに対して、1000 倍以上あるので、効率向上の為に上記のインターフェイス方法を使用した。

tweet が retweet されるかどうかを判断するには、retweeted_status が存在するかどうかを確認すれば良く、retweeted_status にはソース tweet の情報が含まれる。

3. 分析ツール

作者は多くの分析スクリプトを提供しており、その一部は「graphword.py」などのテキスト分析用で、一部は、データベース内の統計情報を出力するためである。例えば、すべての tweet をカウントし、retweeter id と user id を出力する「listretweet-

```
bulk = db.tweets.initialize_unordered_bulk_op()
for s in posts:
  count += 1
  if s.lang == config.lang: flag = True
  j = pack_tweet(db, s)
  d = j['created_at']
  try:
   bulk.insert(j)
  except:
    print "some issue", j, sys.exc_info()[0]
    sys.stdout.flush()
    pass
```

図 2.9: tweet データ収集

ers.py」など。 また、ユーザーの性別を分析するために、gender.py などが実装され、いくつかの分析アルゴリズムもある。

2.2.3 「プロジェクト版」の使用方法

以下の内容は、github の作者の説明 [7] を要約したものである。

1. 設定

mongodb をダウンロードしてインストールする。ubuntu システムを使用している場合は、次のステートメントを使用して直接インストールできる。

- **sudo apt install mongodb**

最新バージョンの mongodb をインストールしたい場合は、以下の mongodb 公式サイトにアクセスしてインストールできる。

- **https://docs.mongodb.com/manual/tutorial/install-mongodb-on-ubuntu/**

pip を利用し下記のパッケージすべてインストール

- **pip install pymongo sklearn unicodecsv numpy scipy nltk ipython xmltodict emoji progress matplotlib bs4 lxml python-twitter tweepy selenium pyvirtualdisplay python-igraph python-dateutil**

git clone を利用し、「プロジェクト版」をダウンロード

- **git clone git@github.com:polyvios/twAwler.git**

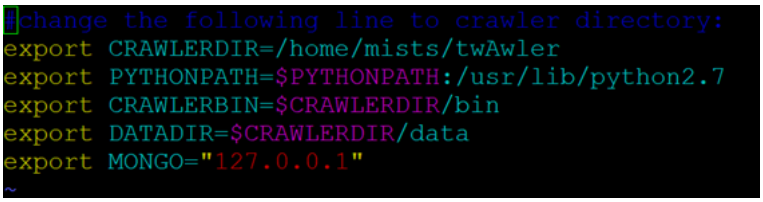


図 2.10: env.sh

図 2.11: config.py

次に、実行する「プロジェクト版」の環境変数 (図 2.10) を設定し、「プロジェクト版」が存在するディレクトリを$CRALWERDIR に追加し、python2.7 の位置と mongodb の IP アドレスを追加する必要がある。

- **vim env.sh**

環境変数起動方法

- **. env.sh**

「install.sh」を実行して data フォルダを作成する。そのフォルダは「プロジェクト版」のプロセス ID、ログと画像など保存するためである。

- **scripts/install.sh**

「config.py」(図 2.11) の内容を修正して、TwitterAPIkey と保存されたデータベース名を追加する。

セキュリティ上の理由から、すべての mongodb データベースとグループユーザーに異なるユーザーとパスワードを設定することが推奨される。そうしないと、他の人がデータベースに簡単にアクセスして操作を実行できるのでを要する。データベースのユーザー名とパスワードを追加された後、「utils.py」の init_state に下記のコードを追加する：

- **db.authenticate(”user”, ”password”,mechanism=’SCRAM-SHA-1’)**

2. 基本の使用方法

次のスクリプトを使用して、クロールするユーザーをデータベースに追加する。ユーザー名の例

- **bin/adduser.py (twitteruser)**

ユーザー ID の例

- **bin/adduser.py –id (twitterid)**

特定のユーザーの tweet データを取得したい場合は、dumpall.py を使用して収集できる。

- **bin/dumpall.py (twitteruser)**

一部の user ーを追加した後、「start-all.sh」を使用して「プロジェクト版」を開始して、データの収集を自動的に開始する。その後、スクリプトはいくつかの tmux プログラムを開始して、追加されたすべてのユーザーのデータを収集する。

収集コンテンツやサンプリング法を変更したい場合は、「start-all.sh」スクリプトを直すことができる。

3.「プロジェクト版」停止

「scripts/stop.py」を使用し、クローラーを停止する。スクリプトは、すべてのクローラープロセスに（ロックファイル）停止して、さらに 15 分間待機するように指示する。 破損した状態を作成するリスクがないため、クローラープロセスを終了することもできる。クローラープロセスを終了するには、data フォルダ内の対応するファイルに保存されている PID を確認する。1 つ以上のクローラープロセスが何らかの理由で停止した場合は、「scripts/stop.py」を再実行して、すべてのクローラープロセスを再起動できる。

第3章　本研究で扱うUserネットワークを抽出する手法

本研究は、ある期間の各Seednodeが公開したTweetを収集し、Tweetとretweet関係を持つUserを集めて、User達のretweet行為による繋がりと考える。データ収集前にまずネットワークの抽出方法を決めなければいけない。以下は本研究で考えたTweetとUserのretweet関係から重き付き有向グラフのUserネットワークの抽出手順について説明する。

3.1　二部グラフ

二部グラフは頂点集合を2つの部分集合に分割して各集合内の頂点同士の間には辺が無いようにできるグラフのことである。本研究に使う二部グラフは、先の収集データから抽出されるTweet-User間のネットワークで、Tweet同士とUser同士の間はリンクが無く、TweetとUserは「Retweet」関係で繋がっている (図 3.1)。

3.2　TweetとUserを繋ぐ無向二部グラフ

まず、研究対象のSeednodeからTweet収集を始める。各Tweetに対し、retweeterを集める (前述のTweetとretweet関係を持つUser)。Retweeterの収集はpythonのpython-twitterパッケージの「GetRetweeters」関数を利用する。その「GetRetweeters」関数は、status_idパラメーターで指定されたTweetをretweetしたUserに属する、最大100のUserIDのコレクションを獲得できる。

最大百人以下のretweeterしか集められない制限があり、特定のTweetは一定時間を回して異なるretweeterが収集可能ではあるが、 retweeter増長スピートと収集制限によって、全体の収集はほぼ不可能である。また、異なるTweetに対して、同じUserが収集される可能性もあって、実際の収集には、1つSeednodeに対して、数千から2万以下のretweeterしか集められない。そこで、現在研究対象としているretweeterは全収集ではなく、Twitter側に一定のサンプリングをしたretweeterを対象と考える。

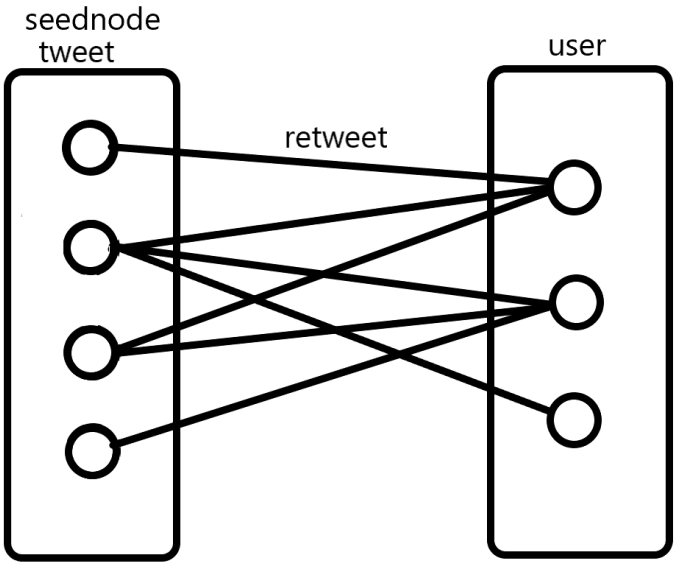


図 3.1: 二部グラフ

次に、Tweet と User の二部グラフから Tweet ネットワークを抽出することについて説明する。ノードは Seednode Tweet と User、リンクは retweet 関係で構成された二部グラフ (図 3.1) で、Retweet 関係には重みが付いていない。

3.3 重き付き無向の Tweet ネットワーク

Tweet と User を繋ぐ無向二部グラフを利用して、Tweet の間重き付き無向グラフを抽出する。

もし 1 人の共通 User があれば、それらの Tweet の間に 1 本のリンクを引く。Tweet の間にもし複数の共通 user があれば、重みとして加算する。図 (3.2) の例を挙げると、Seednode Tweet の１と２は共通 User １があるので、Seednode Tweet の１と２には 1 つリンクが繋がる。Seednode Tweet の２と３は共通 User １と２があるので、Seednode Tweet の１と２には 1 つリンクが繋がり、重みは２になっている。図 (3.2) の例から抽出した Tweet ネットワークが図 (3.3) である。

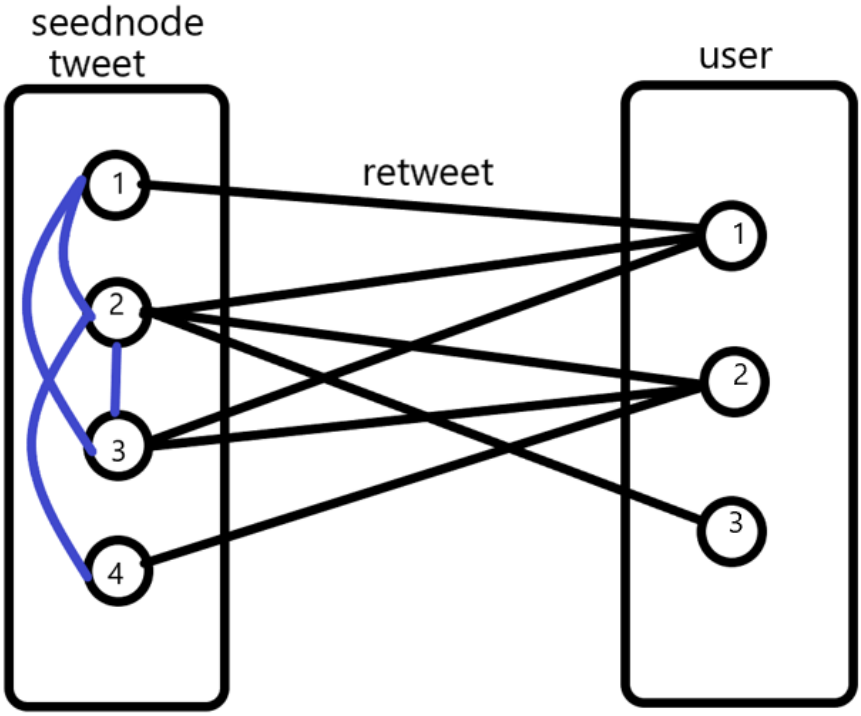


図 3.2: 無向二部グラフから Tweet ネットワーク変換の例

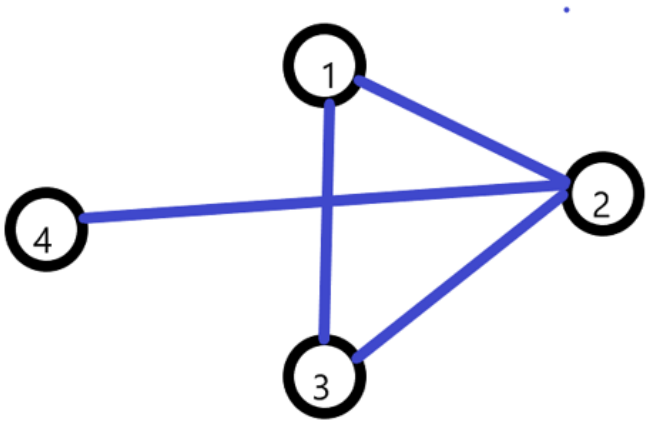


図 3.3: Tweet ネットワークの例

3.3.1　リンクの重み指標

後述する Louvain 法は重みによって、ネットワークの Modularity の計算は変わり、最終的には分類の結果に影響を与える。本研究は以下の 3 種類の重み指標を利用して、Tweet ネットワークのコンミュニティを分類する。

1. 次数（Degree）

次数を重みとした場合はリンクの重みは常に「1」であり、ネットワークは単なる無向グラフになっている。

2. 共有 User の数（Weight）

リンクの重みは常に「1」の次数と違って、前述の Seednode Tweet の共通 User の数を加算してネットワークの重みとする。

3.Jaccard

後述の 3.6 節で説明する Jaccard 類似度係数を利用して、重み値を計算する。図 (3.3) の例では、ノード 1 の隣接ノード集合は 2,3、ノード 2 の隣接ノード集合は 1,3,4 である。ノード 1 とノード 2 の集合和は 1,2,3,4 合計 4 つのノードが含まれる。ノード 1 とノード 2 の集合積は 3 合計 1 つのノードが含まれる。ノード 1 とノード 2Jaccard 類似度係数 J は式 (3.1) のようにで計算できる。

$$\frac{\text{集合積のノード数}}{\text{集合和のノード数}} = \frac{1}{4} \tag{3.1}$$

3.4　重き付き有向辺の User ネットワーク

重き付き有向辺を持つの User ネットワークは User と User 間の retweet 関係を示すネットワークで、その抽出手順は以下に従う。

1. 収集したデータから各 Tweet と retweet 関係で繋がる User を記録する。
2. Tweet ネットワークは後述する Louvain 法によって、Tweet はコンミュニティ別に分類する。
3. 分類した Tweet のコンミュニティに対して、retweet 関係で繋がる User のグループを作る。
4. User のグループ内には retweet 関係を利用して、ネットワークを作る。

例として、図 (3.4) 矢印の方向は retweeter(retweet 行為をする User) から tweeter(retweet された User) に向く。もし V1 が V2 に複数回 retweet 行為をすれば、リンクの重みに加算する。

User ネットワークを抽出すると、ネットワーク内ノードの情報拡散は retweet 行為によって、次数や k-core などのネットワーク指標で分析できるようになる。

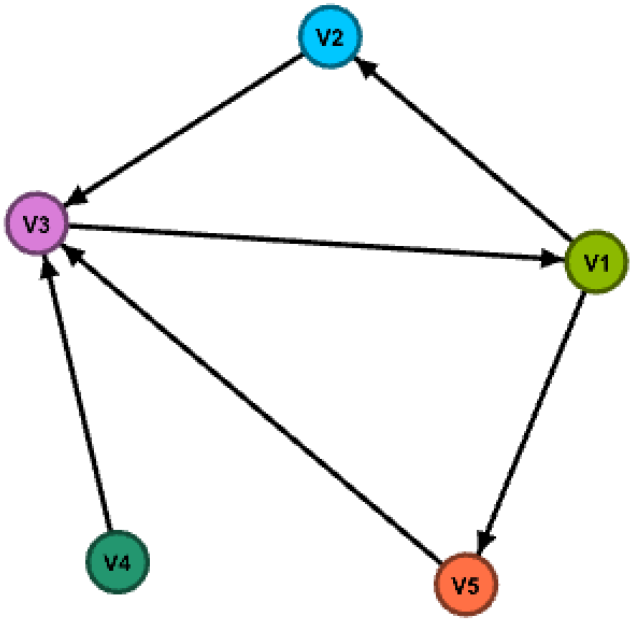


図 3.4: User ネットワークの例

3.5 Louvain 法

Louvain 法 [8] はネットワークモジュール性 (Modularity) の最大化に基づいて、ネットワーク内のコミュニティ（密に結合しているノードの集合）を抽出する手法である。本研究では、Tweet ネットワーク（重き付き無向グラフ）をコミュニティ抽出する為、Louvain 法を利用する。以下の説明では全部無向グラフである場合を考える。

3.5.1 モジュール性 (Modularity)

モジュール性 [9] とはコンミュニティ抽出に関する指標である。Modularity の基本形は以下の式 (3.2) で定義される。

$$Q = \frac{1}{2L} \sum_{i,j=1}^{N} A_{ij} \delta(g_i, g_j) \tag{3.2}$$

ここで、A_{ij}は行 i と列 j の隣接行列 A の要素で、重みがない場合は二値 ($A_{ij} = 1$ または 0)、重みある場合は A_{ij}が重み値である。ここで、$\delta(g_i, g_j)$ は分解の仕方を表して、ノード i と j が同じコンミュニティに属するならば$\delta(g_i, g_j) = 1$、異なるコンミュニティに属するならば$\delta(g_i, g_j) = 0$である。式 (2.2) で定義された $Modularity$ は

、リンクの総数 $L = \frac{1}{2}\sum_{i,j=1}^{N} A_{ij}$ に対する (重みある場合、L はリンクの総数ではなく、リンクの重みを累計する) 割合を表す。

但し、式 (3.2) の形のままでは、ネットワーク全体を 1 つ分割するときに Q は最大 (Q = 1) になる。Q の最大値化する分割が不分割になるのでは式 (2.2) の意味が無くなるので、その問題を解決する為に、以下の手順で「ナル (null) ネットワーク」を考え、Q の最大値を求める。

1. すべてのリンクを切断して、ネットワークを「ナル」(null) 状態に変換する。

2. 各ノードの次数が維持されるようにノード間をランダムにつなぎ直すに従い、ノード i と j の接続確率 P_{ij} でナルネットワークの隣接行列 $\mathbf{P} = P_{ij}$ を定める式 (3.3) 。

$$P_{ij} = \frac{k_j}{\sum_{j'=1}^{N} k_{j'}} \times k_i = \frac{k_i k_j}{2L} \tag{3.3}$$

もし重みがある場合は (式 3.3)k_i と k_j は次数ではなく、重み値を累計する。ナルネットワークではランダム化により元のネットワークの個性が消されている。したがって、ナルネットワークを基準としてそこからの差異を検出することにより、元のネットワークの特徴を明らかにすることができる。そこで、式 (3.2) を式 (3.4) のように修正して、$Modularity$ を定義し直す。

$$Q = \frac{1}{2L}\sum_{i,j=1}(A_{ij} - P_{ij})\delta(g_i, g_j) \tag{3.4}$$

式 (3.4) に従えば、もし i と j の間にリンクがない $(A_{ij} = 0)$ 場合で、i と j(k_i と k_j はゼロより大) を同じグループに分割したら、Q の値は負数になる。このように、ネットワーク全体を 1 つの組になる場合は避けられ、Q の最大値が求められる。

3.5.2 Louvain 法のアルゴリズム

一方、Modularity を最大にするものを見つけ出すという問題は NP 困難である。Louvain 法は貪欲法により、Modularity を最大化を近似的に解くアルゴリズムである。この方法は非常に高速であり、精度も経験的に優れている [10]。そこで、以下の手順をくり返すことでコミュニティを求める。

Step1: ネットワーク内の各ノードを独立したコミュニティと考え、コミュニティの数はノードの数と同じとする。

Step2: 各ノードに対して、順次に各ノードの近隣 (直接リンクと繋げている) コミュニティに分配して、分配前の Modularity 値 Q と分配後の Modularity 値 Q と比べ、ΔQ を計算する。もし $\max\Delta Q > 0$ の場合は、ノードはそのΔQ 最大のコンミュニティに分配する。もし $\max\Delta Q <= 0$ の場合は、分配しない。

Step3: 全ノードは ***Step2*** の手順で繰り返して、コミュニティが変化しないまで続ける。

Step4: 前の手順で分配したコンミュニティは1つ1つのノードとして、コンミュニティ間のリンクはノード間のリンクとして扱い、ネットワークを圧縮する。圧縮したネットワークは普通のネットワークのように ***Step1*** から ***Step3*** までまた繰り返す。

Step5: 最終のネットワークとして Modularity が変化しないとならば、計算は終わる。

前述の手順を見ると、Louvain 法のアルゴリズムにおいて一番計算量が多い所は最初のコンミュニティ分配過程である。ネットワークを圧縮したら、ノードとリンクの数が段々減り、計算量も大幅に減少する。

$$\Delta Q = [\frac{\sum_{in} + k_{i,in}}{2m} - (\frac{\sum_{tot} + k_i}{2m})^2] - [\frac{\sum_{in}}{2m} - (\frac{\sum_{tot}}{2m})^2 - (\frac{k_i}{2m})] \tag{3.5}$$

式 (3.5) は ΔQ の計算式で、$\sum_{in}$ はネットワーク内リンクの重みの総和、$\sum_{tot}$ は、ネットワーク内の全ノードに隣接するリンクの重みの総和、k_iはノード i を端点とするリンクの重みの総和、$k_{i,in}$はネットワーク内にあるノード i を端点とするリンクの重みの総和をそれぞれ表している [8]。

3.6 Jaccard 類似度係数

Jaccard 類似度係数 [11] は Paul Jaccard が 1912 年に提案した、サンプルセットの類似性と多様性を測定する為に使用される統計量である。

Jaccard 類似度係数の計算式は式 (3.6) に示す。Jaccard 類似度係数 J は「グループ A または B」と「グループ A かつ B」の比率である。J の値の範囲は [0,1] で、J のが大きい程、2 つのグループの類似性が強くなると定義している。

$$J = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{3.6}$$

ネットワーク内 2 つのノードの Jaccard 類似度係数 J は、共通の隣接ノードの数を、2 つのノードの少なくとも 1 つの隣接ノード数で割ったものである。

以上の定義で、*Jaccard* 類似度係数 *J* がリンクの重みになる場合は、*2* つノード共通の隣接ノードの比率が大きい程、重みが大きくなり、分類上は同じグループやコンミュニティに分配する可能性が大きくなる。

第4章　分析・可視化・評価

本章では、ネットワーク分析指標を用いて、第 3 章の抽出方法による、Tweet ネットワークと User ネットワークについて分析、かつ可視化した結果について述べる。

4.1　実験データ

本研究は User の retweet 行為を調べる為、Twitter 上の 2019 年に最も話題となった上位 10 のゲーム User の 8 個を選択して、seednode として扱い、収集を始める。その 8 個の seednode の Tweet 数と割合を表 4.1 でに示す。

ゲーム名	ゲーム略称	Tweet 数	割合
Fgoproject	fgo	1457	27.00%
IdentityVJP	ivj	1053	19.50%
GranblueFantasy	gbf	968	17.90%
FortniteGame	ftg	872	16.20%
EnsembleStars	ebs	497	9.20%
PUBG	pug	205	3.80%
Minecraft	mcf	193	3.60%
FinalFantasy	fnf	150	2.80%

表 4.1: seednode の Tweet 数と割合

ゲーム名は Twitter での各ゲームの User 名で代用する。また、表現し易いゲームの略称を使用する。Tweet 数では 2020 年 1 月から 10 月の間、各 seednode からオリジナルで公開した、且つ retweet 数が統計できる Tweet の数を対象とする。この retweet 数が統計できる Tweet とは、一般認識上の retweet 数がゼロではない場合である。一方、Tweet 数は前述の「Tweet と User を繋ぐ無向二部グラフ」の「Tweet」部分、表 4.1 の「割合」は各 seednode の Tweet 数は二部グラフ「Tweet」ノートの割合である。

```
_id: ObjectId("5f75962422c99ce35742f988")
created_at: 2013-06-27T06:05:10.000+00:00
description: "「グランブルーファンタジー」公式Twitterアカウントです。運営からのお知らせをツイートします。"
followers_count: 901468
friends_count: 16
id: 1549889018
id_str: "1549889018"
listed_count: 9751
name: "グランブルーファンタジー"
profile_background_color: "131516"
profile_background_image_url: "http://abs.twimg.com/images/themes/theme14/bg.gif"
profile_background_image_url_htt… : "https://abs.twimg.com/images/themes/theme14/bg.gif"
profile_banner_url: "https://pbs.twimg.com/profile_banners/1549889018/1552989165"
profile_image_url: "http://pbs.twimg.com/profile_images/1305705091547029504/pNhHsiuG_norma..."
profile_image_url_https: "https://pbs.twimg.com/profile_images/1305705091547029504/pNhHsiuG_norm..."
profile_link_color: "009999"
profile_sidebar_border_color: "FFFFFF"
profile_sidebar_fill_color: "EFEFEF"
profile_text_color: "333333"
profile_use_background_image: true
screen_name: "granbluefantasy"
statuses_count: 9115
url: "https://t.co/15Gqq9oUyB"
verified: true
screen_name_lower: "granbluefantasy"
timestamp_at: 2020-10-01T08:41:08.585+00:00
```


図 4.1: User データ

count	color	text	time
favourites	background	description	created_at
followers	link	name	timestamp_at
friends	sidebar	screen_name	
listed	sidebar_fill		

表 4.2: User data の例

図 4.2: Tweet データ

4.1.1 データの例

表 4.2 は User データ (図 4.1) のまとめであり、主に数値データ「count」、ホームページ色「color」、テキストデータ「text」、時間データ「time」によるこの 4 部分で構成される。

count	text	time
retweet	hashtags	created_at
favorite	text	retweeted_status.created_at
retweeted_status.favorite_count		

表 4.3: Tweet data の例

表 4.3 は Tweet データ (図 4.2) のまとめであり、主に数値データ「count」、テキストデータ「text」、時間データ「time」の 3 部分で構成される。

4.1.2 Seednode についての紹介

本研究で扱うゲーム公式 Twitter についての基本情報を表 4.4 に示す。選択したゲーム公式 Twitter の半分はスマートフォンゲーム、開発国の半分は日本の会社が開発して、半数以上のゲームの公開時間は 5 年程など一目瞭然な情報以外は省している。同じスマートフォンゲームの中でキャラクター収集育成するゲームが主流になっているが、各ゲームには微妙な差別がある。例を挙げると：

- Fate/Grand Order はキャラクターの収集、ゲームのストーリーを重視して、ゲームをクリアするため、キャラクターの連携に生み出す策略も遊び方の一部分である。
- Ensemble Stars は主に女性向けで、音楽とキャラクター間のストーリーに着目している。

- Granblue Fantasy はキャラクターの収集と冒険ストーリーを中心に展開している。

中でも「Fate/Grand Order」と「Granblue Fantasy」のキャラクターデザインは多くの男性ゲーマーを招き、「Ensemble Stars」は女性ゲーマーを多く引き付けている。

近年人気の「PUBG」と「Fortnite」の「アルタイム対抗」類ゲームの影響で、スマートフォンゲームの中で唯一キャラクターの収集や育成ではないスマートフォンゲーム「Identity V」も流行っている。「対等対抗」のゲーマー皆は同じパラメータで対抗するのと違って、「非対等対抗」とは、ゲーマーは 1 人強い boss キャラ、もしくは数人の人キャラ中を選べ、人キャラは boss キャラに比べると遥かに弱いが、少し異能を持ち、チームを組んで boss キャラと対抗する。

また、リストの中に、公開時間が僅か数年のゲーム以外、「Final Fantasy」は日本のゲームの代表として、世界のゲーマーを魅了している。「Minecraft」は簡単で自由度高い世界に創造力を発揮できる遊び方で人気を集めた。その 2 つの歴史あるゲームは新作の原因で Twitter 上の話題また活躍が続けられている。

ゲーム名	主な platform	開発国	公開時間	遊ぶ方
Fate/Grand Order	スマートフォン	日本	2015 年	キャラクター収集
Fortnite	Pc	米国	2017 年	リアルタイムで対等対抗
Final Fantasy	ゲーム機	日本	1987 年	主は RPG ゲーム
Identity V	スマートフォン	中国	2018 年	リアルタイムで非対等対抗
Granblue Fantasy	スマートフォン	日本	2014 年	キャラクター収集
Ensemble Stars	スマートフォン	日本	2015 年	キャラクター育成
PUBG	Pc	韓国	2016 年	リアルタイムで対等対抗
Minecraft	Pc	スウェーデン	2009 年	サンドボックスゲーム

表 4.4: ゲームの基本情報

表 4.4 は、すべてのゲームを網羅するには難しいことから、ゲームの一部分の情報だけを表しているが、各ゲームの差別はある程度捉えていると考えられる、次章の分析結果と一緒に議論する。

表 4.5 は、各 seednode に対して収集したデータの規模である。「User の数」は seednode の Tweet を利用して、収集できた User の数、「User の Tweet 数」は収集できた各 User 発信した Tweet である。前述の二部グラフのデータを基礎として、User の Tweet を集めることで、User と seednode の retweet 関係が把握できる。図 4.2 の例では、「retweeted_status」ラベルが付いている Tweet データは retweet である。「retweeted_status」ラベルの中の「id」と「user.id」を調べると retweet された Tweet の id と元作者の id が分かる。Twitter 上の User は、自分のデータを公開したくない場合は自分のアカウントをプロテクト状態に入ることができる。その状態に入ると、API は User の retweet 行為が認識できるが、この User に関する

すべてのデータは収集できないとプロテクトが反応する。「プロテクトの数」はそういう状態の User の数である。また、異なる seednode に対して、同じ User を集める可能性があって、データをまとめると User の数は表 4.5 より少し減る。

ゲーム略称	User の数	User の Tweet 数	プロテクトの数
fgo	20,109	48,076,087	750
ivj	15,911	38,795,658	759
gbf	15,115	46,023,605	627
ftg	12,871	15,109,648	136
ebs	8,090	18,550,376	456
mcf	4,527	6,025,666	24
pug	4,461	7,173,606	70
fnf	3,437	8,968,832	38

表 4.5: データ規模の概要

最終的に獲得した Tweet データの数 188,723,478 個 (1.88 億程)、retweet は 109,990,556 個 (1 億程)、大体は Tweet データの半数以上を占める。収集した User 数合計 79,744 人、Twitter API の収集制限によって、1 人の User に対して、1ヶ月内公開にした Tweet データもしくは最大 3200 個程の Tweet データを集められる。これらを踏まえて、データの偏差による分析結果の偏差を避けるため、まず確実に 2020 年 1 月 1 日以後のデータが収集できる User を選出して、データをクリーニングする。その結果、2020 年 1 月から 2020 年 9 月の間すべての tweet データが収集できた User は 29800 人 (seednode も含め) となった。結果的に、クリーニングした後の 2020 年 1 月 1 月から 9 月末までの Tweet データは 115,389,118(1.15 億) 個である。また、29800 人の User が発信した Tweet27,378,400 個である。その中、retweet は 15,552,554 個、seednode の Tweet に対する、retweet 行為をする Tweet 数は 755,692 個、本研究の分析は主にそのクリーニングしたデータに対して行う。

4.2 Tweet ネットワークの分析結果

まずは、Tweet ネットワーク全体の可視化を示して、それらの可視化結果による、Tweet ネットワークの特徴を述べる。次に、2.1.2 節に述べたネットワークの分布指標を利用して、ネットワーク全体の可視化結果と比べて分布指標を議論する。さらに、2020 年 1 月から 9 月間のネットワーク月ことの変化に通じて、各 seednode の特徴について分析する。

4.2.1 Tweet ネットワークの可視化

図 4.3 は Tweet ネットワークを可視化した結果である。本研究のネットワーク可視化は gephi を利用し、グラフレイアウトは OpenOrd[12] を使う。OpenOrd は膨大なデータによるグラフ構造を処理するために特別に設計された、力指向 (Force-directed) のレイアウトアルゴリズムである。

ネットワーク (図 4.3) は 5395 個ノード (表 4.1) で構成され、各ゲームは異なる色で標記される (左上は各ゲームの略称とノードの色)。ネットワーク内のリンク数 (3439498 本) が多いので、1 つ 1 つのリンクを区別するのは難しいが、ノード間の距離と束になったリンクの太さによって、各ゲームの関係がある程度は捉える。例えば、mcf(紺色)、pug(マゼンタ)、ftg(黒色) 間の繋がりが多い、可視化によるその 3 つのゲーム間の距離が短いである。同一ゲームノード間の距離については、リンクの数に関わる。例えば、ebs(オレンジ色) のような互い密に集中している他、fnf(赤色) のような少し分散しているのもある。それは ebs のノード間のリンクが多い、fnf の方ノード間のリンクが少ないの原因である。

4.2.2 Tweet ネットワークの次数分布

図 4.4 は 8 個のゲームの次数分布図で、縦軸はノードの数、横軸は次数である。比較易い為、本文は以下の 4 種類に分ける。

- (a) Fgoproject と (c) GranblueFantany の 2 つの分布は分布が集中している部分と最大次数は少し違うが、次数分布の形は類似している。一方、(a) ノードの一部分は低次数に集中して、その状況は可視化結果から、紫色の (a) は青色の (c) よりノードが分散して、遠く離れるノードが多く存在する。

- (f) PUBG のノード数 (205 個) は (d) FortniteGame(872 個) の 4 分の 1 である。もし、(f) のノード数が四倍増え、ノードの分布は密になり、(f) と (d) 分布図はもっと類似と考えられる。(f) と (d) が同じく高次数の所に 1 つピークがあるが、(a) と (c) と比べるとピークが緩やかに見える。また、低次数ノード数にも多く、図 4.3 のピンク色 (f) と黒色 (d) は紫色の (a) と青色の (c) より分散している。

- (e) EnsembleStar と (g) Minecraft の次数分布は、ほぼ高い次数の所に集中している。(e) の分布図の形は一見 (a) と (c) に近いが、横軸の最小次数は 460 から始まって、ノードはほぼ次数 500 程のピンクが存在する。(g) の場合は次数 50 以下のノードは 3 つだけあって、残りはほぼ次数 170 以上の所に集中する。また、(e) のノードは 497 個、(g) のノード数は 193 個、図 4.3 を見ると、その二つゲームのノードが各自集中する。そこで、ネットワーク内の次数はほぼ同じ、Tweet 数が同じ規模のゲームと比べて、次数にも大きいと考えられる。

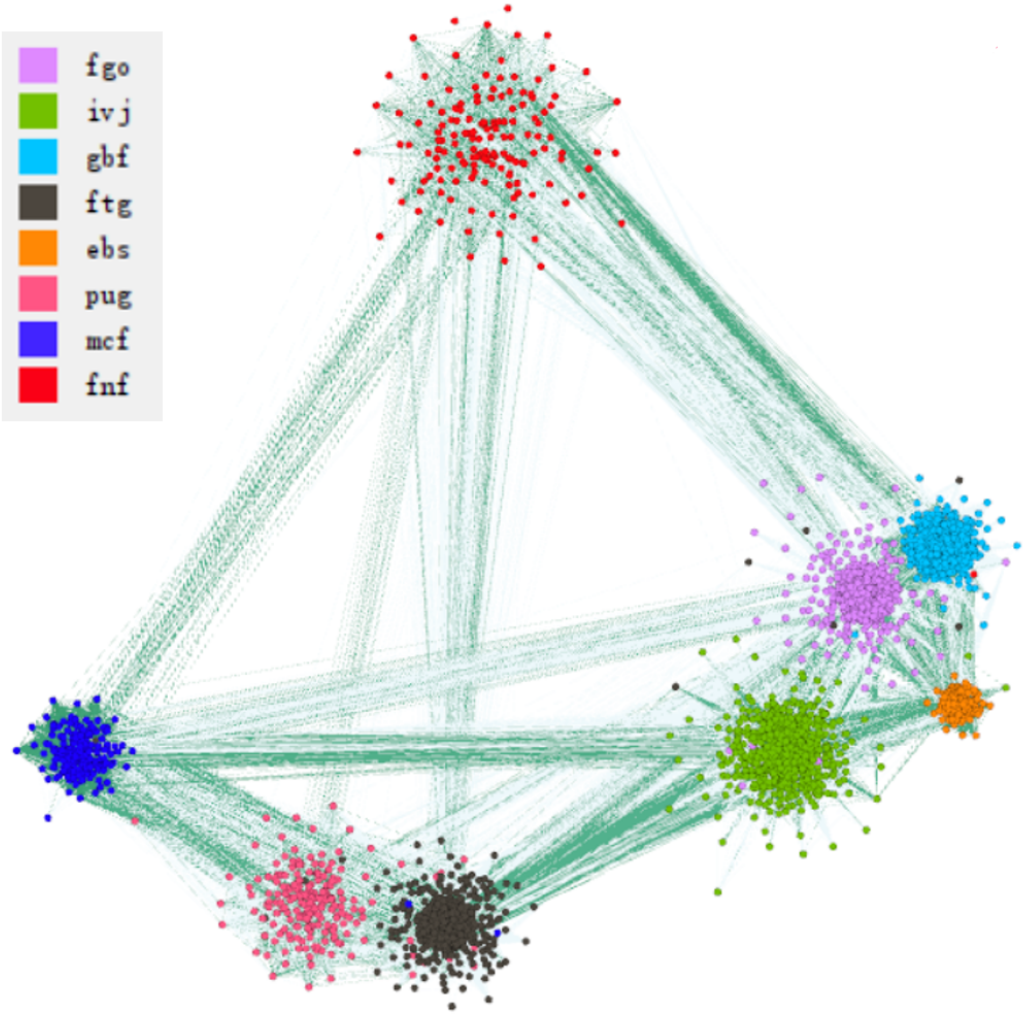


図 4.3: Tweet ネットワークの可視化

- (b) IdentityVJP と (h) FinalFantasy の分布図を見ると、ノード達は分散している。(b) はノードが多い、可視化は困難であるが、実際ノード間はバラバラしであることが次数分布から認識できる。

4.2.3 Tweet ネットワークの K-core 分布

図 4.5 の K-core 分布の中、(b) IdentityVJP、(f) PUBG、(h) FinalFantasy の core 数が少し分散する以外、各ゲームの K-core は最大 core 数の部分に集中する。図 4.4 の次数分布と比べ、K-core 分布図のノード分布がより極端になり、最大 core 以外の core のノード数は全部 1 桁以下になっている。ネットワーク構造の角度を考えると、中心から離れたノードと中心集団がより明白に分けられる。その原因で、次数分布のようにピークや分布変化の視点をから分析するには難しいと考え、中心集団を代表する最大 core 数と他ノードの割合を比較する為表 4.6 を作る。表 4.6 は各ゲームの最大 core 数の部分に含まれるノードの割合、(b)、(f) 、(h) の割合は 70%以下であり、図 4.5 を見ると、(b)、(f) 、(h) の中心集団以外のノードが多く存在する。(b) のノードが横軸を充満して、core 数が分散する。一方、(f) と (h) は他のゲームより分布が分散することが見える。

fgo	ivj	gbf	ftg	ebs	pug	mcf	fnf
91%	59%	96%	80%	97%	66%	98%	48%

表 4.6: 最大 core 数の部分に含まれるノードの割合

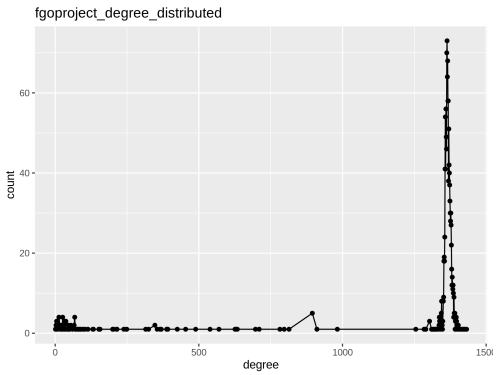
4.2.4 2020 年 1 月から 9 月間の次数変化

図 4.6 の (a) と (b) は、各ゲームの 1 月から 9 月間の次数変化を示す。fnf,mcf,pug の次数は他ゲームより少ないので、その 3 つを単独に (c) と (d) の変化図を作る。

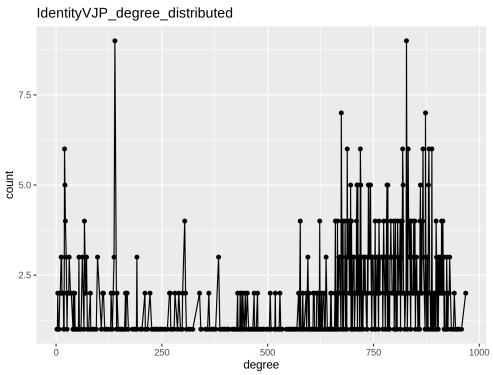
次数の合計値と平均値の変化を見ると、各ゲーム間の顕著な上昇や下降傾向など特にいない。

- 次数の合計値は、gbf は 3 月に突然大きく増えたけど、4 月にはすぐ 2 月と同じ程度に戻った。fnf,mcf,pug の合計値は激しく変化するが、表 4.1 に示した tweet 数の割合は全体の 10%、表 4.5 に示した User の数と User の Tweet 数にも 10%である。もしその 3 つのゲームはある月は普段より少し tweet の発信数が増えでも、データのスケールは他ゲームより小さく、変化の幅はノード数によって影響されやすい。

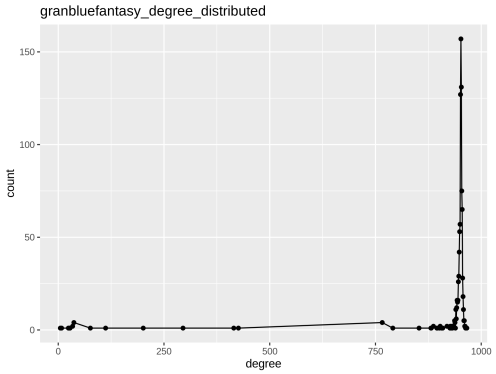
- 次数の平均値は、fgo(1 月以外),gbf,ebs,mcf に大きな変化はないである。特に ebs は 500 程 (ebs のノード数は 497 個)、ほぼ変化していないと見える。一



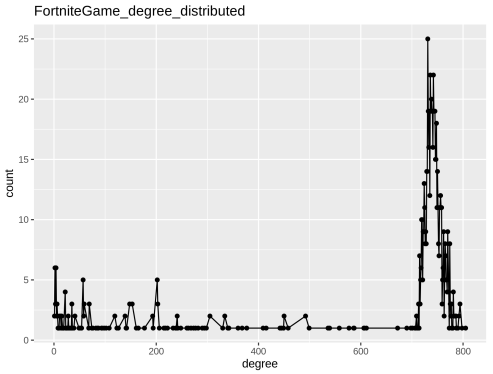
(a) Fgoproject



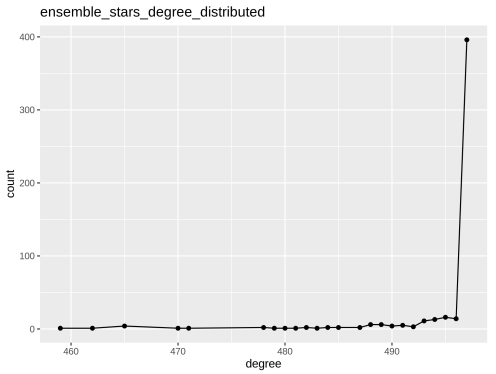
(b) IdentityVJP



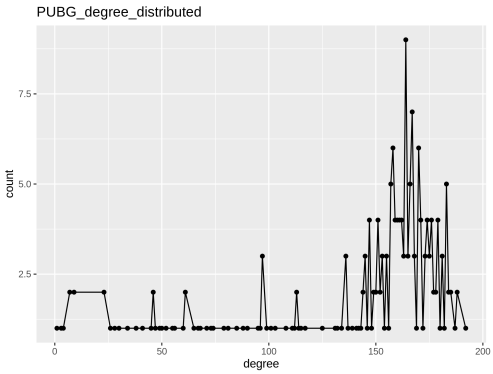
(c) GranblueFantasy



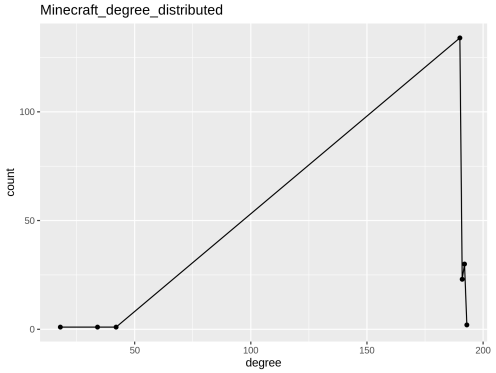
(d) FortniteGame



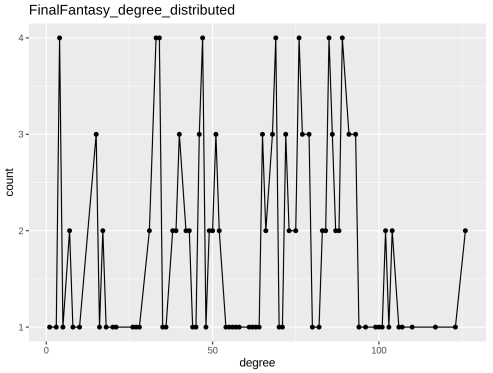
(e) EnsembleStars



(f) PUBG

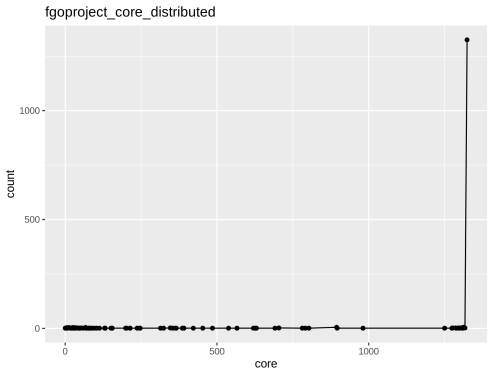


(g) Minecraft

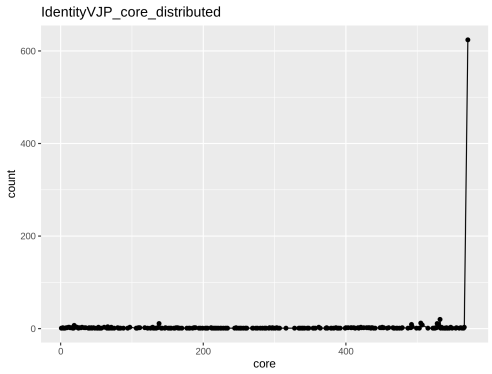


(h) FinalFantasy

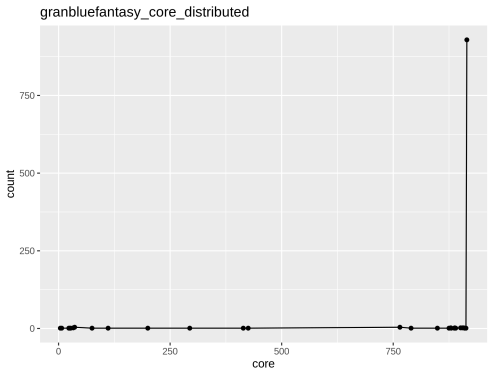
図 4.4: Seednode の次数分布



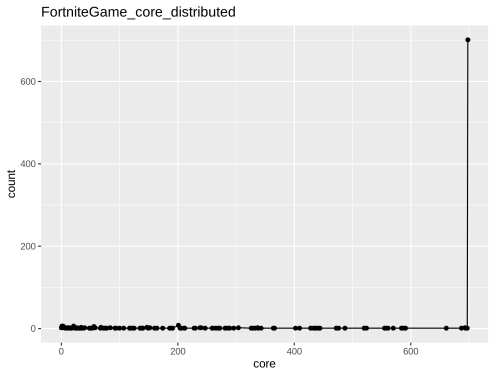
(a) Fgoproject



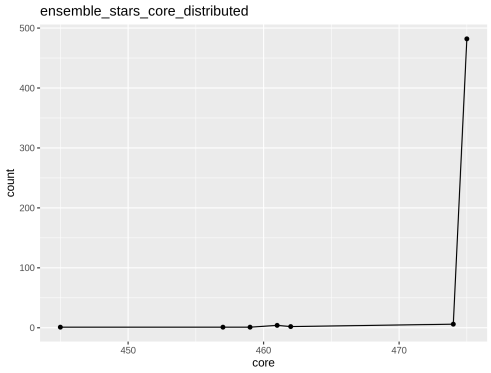
(b) IdentityVJP



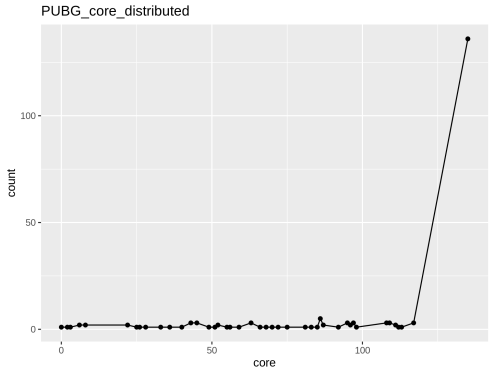
(c) GranblueFantasy



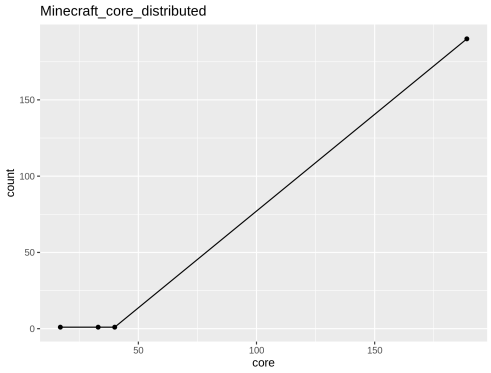
(d) FortniteGame



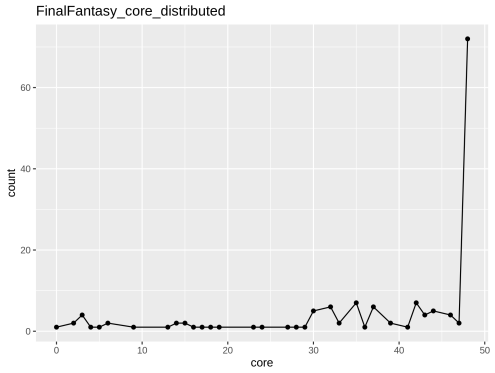
(e) EnsembleStars



(f) PUBG

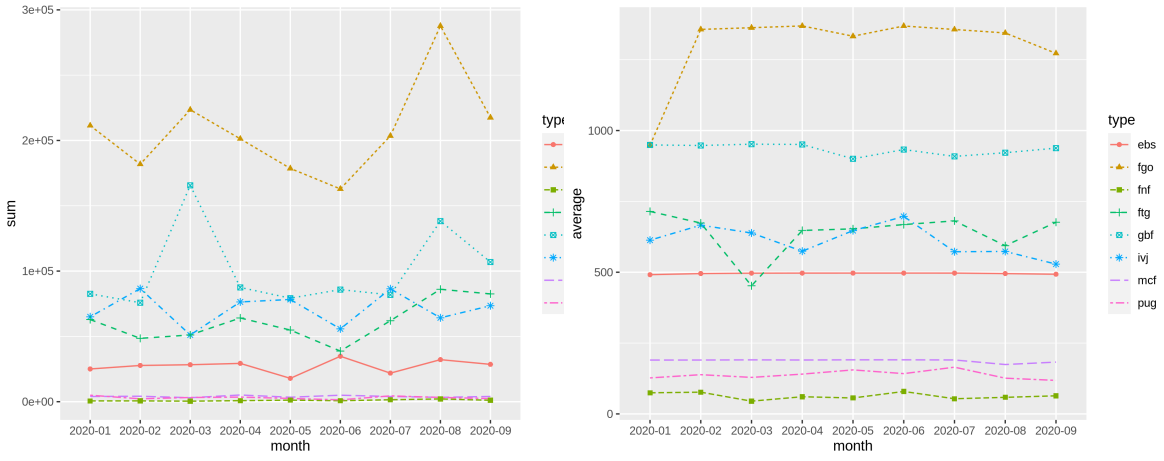


(g) Minecraft

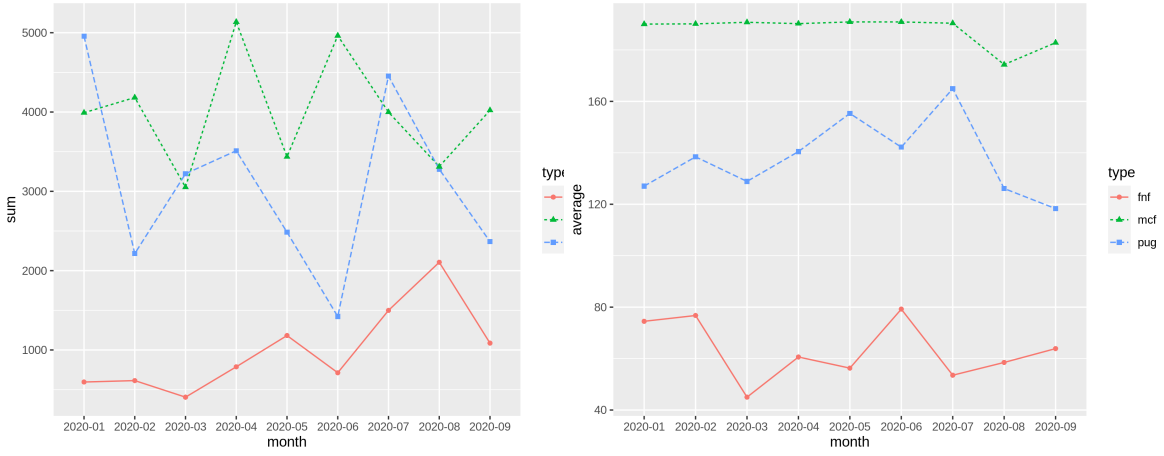


(h) FinalFantasy

図 4.5: Seednode の K-core 分布



(a) 次数の合計値　　(b) 次数の平均値

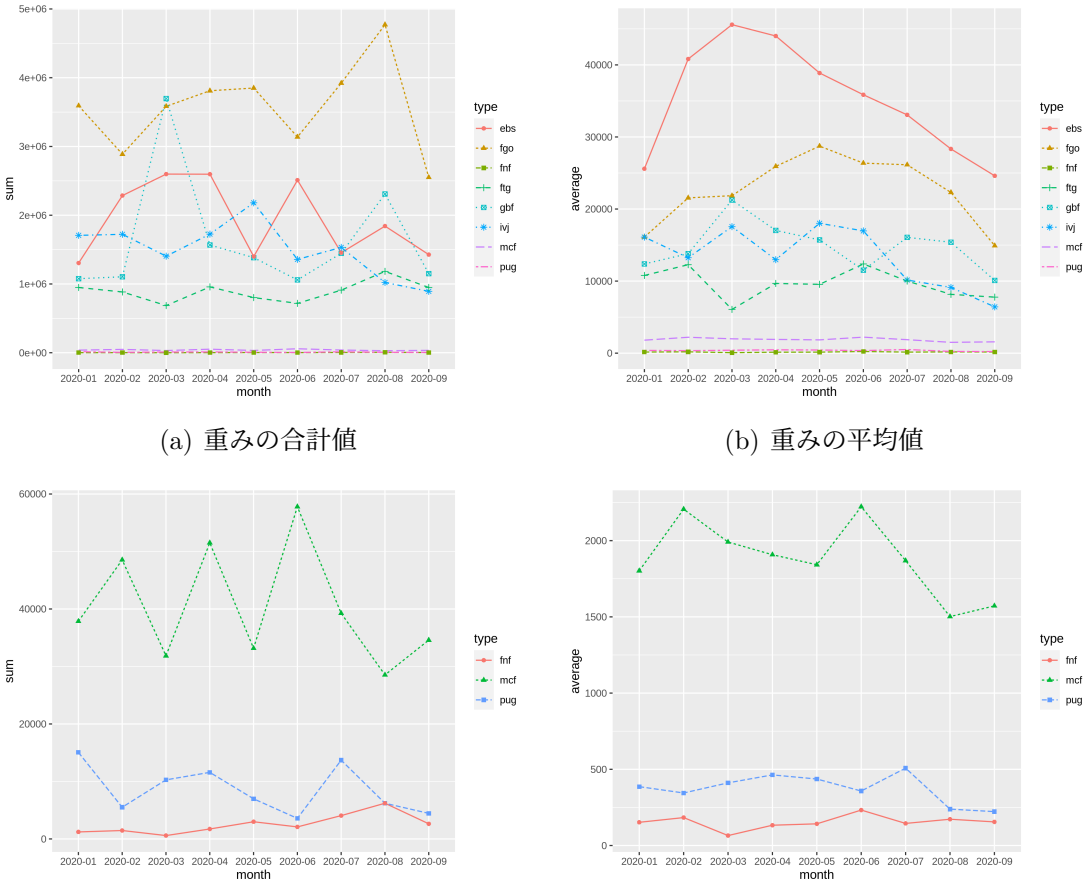


(c) 次数の合計値 (fnf,mcf,pug の部分)　　(d) 次数の平均値 (fnf,mcf,pug の部分)

図 4.6: 2020 年 1 月から 9 月間の次数変化

方、図 4.6 の (b) と (d) 折り線により、ivj,ftg,fnf,pug の変化の幅は前述の 4 つより大きい、月ことの変化は明白と見られる。これらの結果を表 4.6 と比較すると、最大 core 数の割合が 90%を境として、平均値変化の平穏と不穏の 2 グループ、fgo,gbf,ebs,mcf と ivj,ftg,fnf,pug に分けられる。

4.2.5 2020 年 1 月から 9 月間重みの変化



(a) 重みの合計値

(b) 重みの平均値

(c) 重みの合計値 (fnf,mcf,pug の部分)

(d) 重みの平均値 (fnf,mcf,pug の部分)

図 4.7: 2020 年 1 月から 9 月間の重み変化

辺の重みの違いによる、2020 年 1 月から 9 月の間重みの合計値の変化は前節と類似し、ゲーム間の顕著な上昇や下降傾向など特にいないが、重みの平均値については、ebs,fgo,gbf,ivj,mcf は前半年にはある程度は上って、その後下がった。但し、世界各国で自粛よりの影響がどんどん緩和になっている原因かどうかは、2019 年と比べないとわからない。

また、ebs は重み平均値で 1 位になって、前節 ebs の次数平均値と図 4.3 一緒に見ると、ebs のノードはほぼお互いリンクが繋がている (完全グラフ) ことが認識

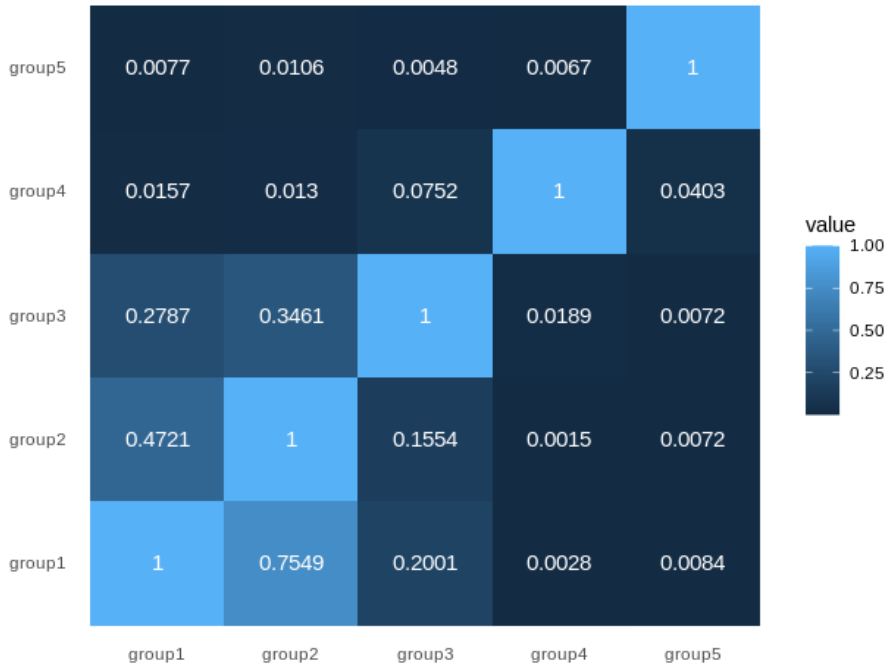


図 4.8: degree 分割した heatmap 図

でき、retweeter の角度を見ると、ebs の Tweet を retweet する人はほぼ必ず ebs の他 Tweet を retweet すると理解できる。

4.3 Tweet ネットワークのコンミュニティ分割

本節は 2.4 節に示した Louvain 法を使い、R 言語の「igraph」パッケージの「cluster_louvain」関数を利用して、次数のみ、重み付けと Jaccard の 3 種のリンクの重み指標で Tweet ネットワークを分割する。コンミュニティ抽出の結果は主要分割だけ収集し、分類数が 1 桁の分割結果は捨てる。分割した各グループ間の共通 User の比率を算出して、heatmap 図を作る。比率の計算方法は縦軸のグループと横軸のグループの共同 user 数を割る縦軸のグループの user 数を用いて、計算式は式 (4.1) に示す。

$$\text{比率} = \frac{\text{座標の縦横と対応する共同 } user \text{ 数}}{\text{縦軸のグループの } user \text{ 数}} \tag{4.1}$$

4.3.1 節は次数 (degree)、重み (weight) と Jaccard の順番で分割結果と heatmap 図を示す。

次数 (degree) のみ				
分割グループ	ゲーム名	tweet 数	user 数	認証数
Group1	fgo	1412	8119	2
	ftg	3		
	gbf	1		
Group2	ebs	497	12983	3
	fgo	36		
	ftg	1		
	ivj	1053		
Group3	fgo	9	5831	4
	fnf	3		
	ftg	1		
	gbf	966		
Group4	fnf	147	1463	5
	ftg	1		
Group5	ftg	866	8795	44
	mcf	193		
	pug	204		

表 4.7: 次数の分割結果

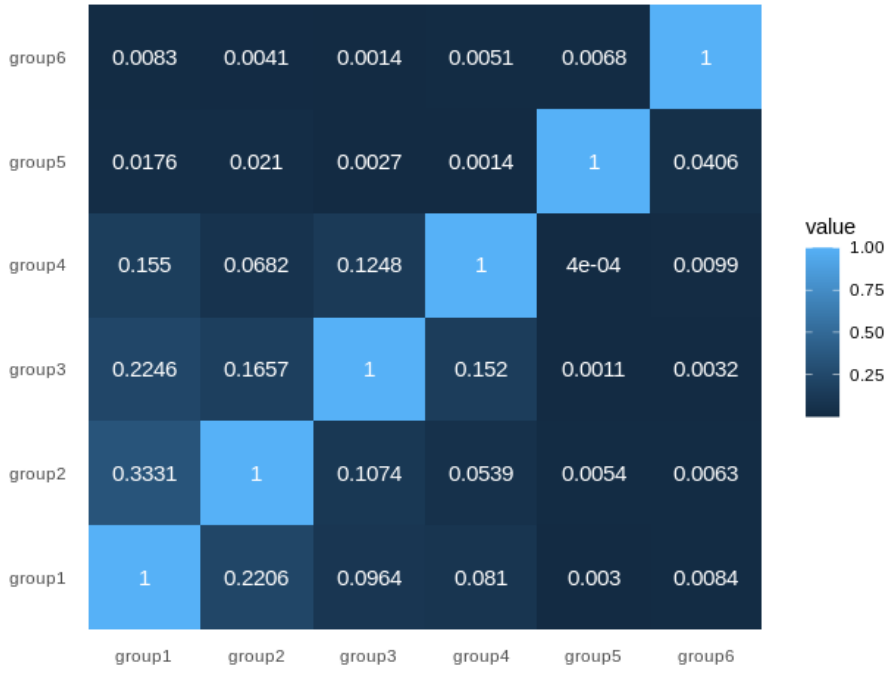


図 4.9: weight 分割した heatmap 図

重み (weight) 付き				
分割グループ	ゲーム名	tweet 数	user 数	認証数
Group1	fgo	1448	8662	2
	ftg	3		
	gbf	1		
Group2	fgo	5	5737	3
	ftg	1		
	fnf	3		
	gbf	966		
Group3	ebs	497	3717	0
	ivj	1		
Group4	fgo	4	4529	2
	ftg	1		
	ivj	1052		
Group6	fnf	147	1477	5
	ftg	1		
Group5	ftg	866	8785	44
	mcf	193		
	pug	204		

表 4.8: 重みの分割結果

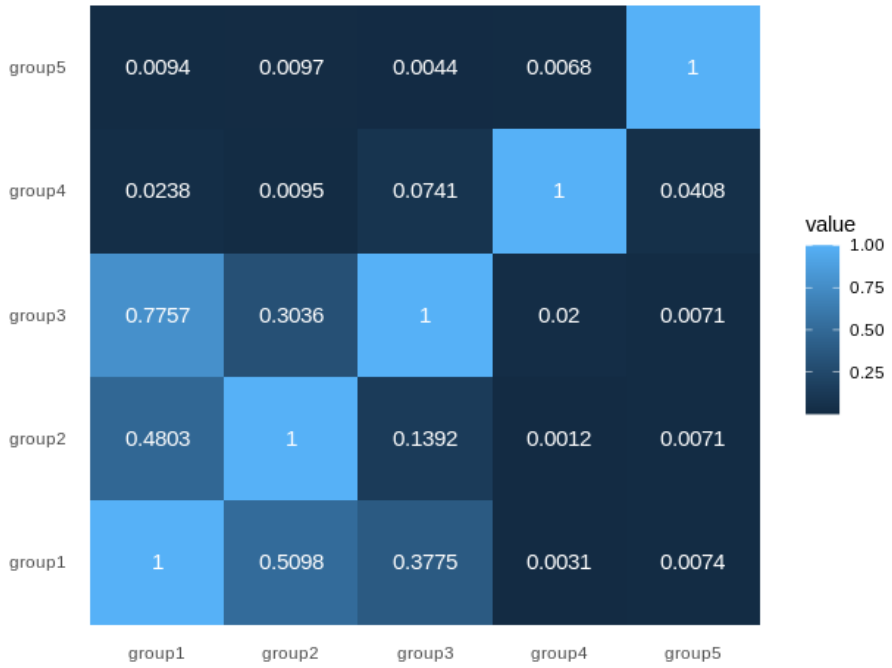


図 4.10: jaccard 分割した heatmap 図

Jaccard				
分割グループ	ゲーム名	tweet 数	user 数	認証数
Group1	fgo	1405	11212	4
	ftg	1		
	gbf	126		
Group2	ebs	497	11902	2
	fgo	10		
	ftg	1		
	ivj	1017		
Group3	fgo	4	5457	3
	fnf	2		
	gbf	835		
Group4	fnf	148	1470	5
	ftg	1		
Group5	ftg	848	8798	44
	mcf	193		
	pug	204		
	fgo	1		

表 4.9: Jaccard の分割結果

4.3.1 分割結果と heatmap 図

4.3.2 結果分析

まず図 4.8 から図 4.10 に示す 3 つの分割結果を見ると、次数と Jaccard は同じく Tweet ネットワークを 5 つグループに分けられる。それに対して、重み値の違いに従って 6 つのグループに分割している。分割の区別が次数と Jaccard は ebs と ivj を 1 つの分類にして、重みはその 2 つを分けた。同じく 5 つのグループに分ける次数と Jaccard の結果からは、分類結果に対して Group4 と Group5 の間はほぼ差が出ないが、Group1 と Group3 の tweet 数と user 数に対し差が大きいことが判明した。原因は次数の分割結果に対し、Group1 はほぼ fgo の tweet 数だけ分類されたが、Jaccard は gbf の tweet 数が 126 個を Group1 に分類した。その影響で、Group1 の場合、Jaccard の user 数は 3093 個多いので、(Group2,Group1) の値を見ると、割合上の差は 0.25(0.75 対 0.5) だが、共通 user 数の差は 413 である (表 4.10)。同様に、(Group3,Group1) の方は、割合上の差は 0.17(0.2 対 0.37) が、共通 user 数の差は 2603 でほぼ 2 倍である (表 4.10)。

グループ	1 と 2	1 と 3	1 と 4	1 と 5	2 と 3	2 と 4	2 と 5	3 と 4	3 と 5	4 と 5
次数	6129	1625	23	68	2018	19	93	110	42	59
jaccard	5716	4233	35	83	1657	14	85	109	39	60

表 4.10: 各グループ間の共同 user 数

さらに、各 heatmap 図を見ると、次数と Jaccard に対する Group1 から Group3、重み値の違いに従って Group1 から Group4 は皆スマートフォンゲームである。共通 user 数からは、スマートフォンゲームと残りのゲームとの繋がりが薄い。一方、マートフォンゲーム同士の間、特に fgo と gbf の繋がりが強い。4.2.1 節で述べたように、fgo と gbf は遊び方が類似する男性向けのゲームであることが、その原因と解釈できる。表 4.7 と表 4.9 から、次数と Jaccard の分割結果において同じグループに分類された ebs と ivj の間にはある程度の繋がりが考えられるが、ebs は女性向けのゲームであり、ivj はある程度は女性ゲーマーに好まれていることが推測される。

分割結果に対して、fnf と ftg、mcf、pug は 2 つのグループに分けた結果から、fnf はノードの数が少ない以上、ノート間の繋がりが弱く、他のゲームとの繋がりにも薄くて、他のゲームと孤立する結果は明白である。一方、ftg と pug は fgo と gbf のように遊び方が類似するゲームであり、mcf は遊び方やゲーム設計に対して全然違うゲームである。図 4.3 の可視化から、mcf は ftg と pug にはそれ程近くないことも認識できる。ftg、mcf、pug の 3 つのゲームは海外のゲーマー数が日本のゲーマー数より遥かに多く、共通 user 数やネットワークのリンクが多くないが、他のゲームと比べると多く、同じ分類になっている。

4.4 Userネットワークの分析結果

本節は、4.3節から求めた各コンミュニティ抽出によるuser数を利用して、retweet関係に基づくUserネットワークに対する分析結果を示す。緑色矢印の方向はretweeterからtweeterの方向、矢印は大きい程、user間のリンクした回数が多くなる。ノードについて、ノードの大きさはin-degreeと繋がり、ノードがより大きい程、多くのin-degreeが集まる。

グラフ内の黒色ノードは認証済みのuser、赤色は未認証のuserである。認証済みとは、著名人のアカウントなど、世間の関心を集めるアカウントが本物であることを示す。例えば、主要な政府関係者や政府機関、著名な組織を代表するアカウント、報道機関の要件を満たす組織の公式アカウントなどである。もしuserが認証済みになら、収集したUserデータ (図4.1) の中に「verified」マークが付与される。

4.4.1 Userネットワークの可視化

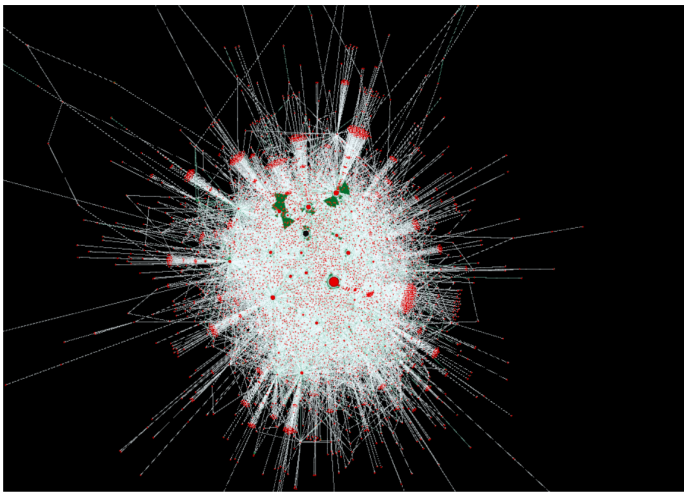
表4.7の次数のみによるグループ分割結果を基づき、3種類の重み指標をそれぞれ可視化して比較する。例えば、次数のみ、重み付きとJaccardの3種類分割結果の中に、主なゲーム名は「fgo」のGroup1を取って、3種類の分割結果は可視化結果1(図4.11) と比較する。次数のみの場合は「ebs」と「ivj」はGroup2に分割したが、重み付きの場合はGroup3とGroup4をに分割したので、比較し易いように、可視化結果2(図4.12) は次数のみとJaccardのGroup2可視化結果各1枚、重み付きのGroup3とGroup4合わせて4枚ある。これら、Userネットワークの可視化として以下の図4.11から図4.15のに示す。

4.4.2 可視化の分析

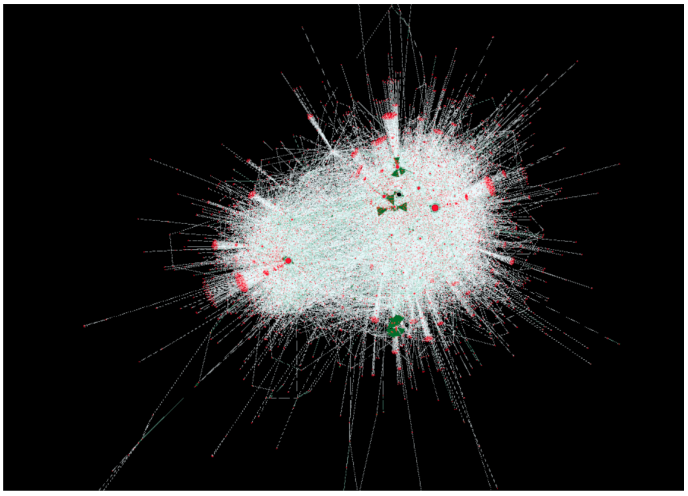
データクリーニングした29800人のuserの中に、僅か59人が認証済みのuserである。研究対象になった8個のゲームは全部認証の条件を満して、認証済みの状態になっている。異なる重み指標と各グループに対する、userの間認証済みの数は表4.7から表4.9に認証数を示す。

それらの認証数を見ると、ftg、mcf、pugの3つのゲームをグループの認証数は、他グループの1桁の数より44個あることで圧倒的に多い。その中で2番目多いのはfnfあるグループ、user数は少ないが認証数は5個ある。それに対して、fgo、gbf、ebsとivjこの4つのスマートフォンゲームでは、user数は多いが、認証数は極めて少なく、特にebsの認証数は0人である。

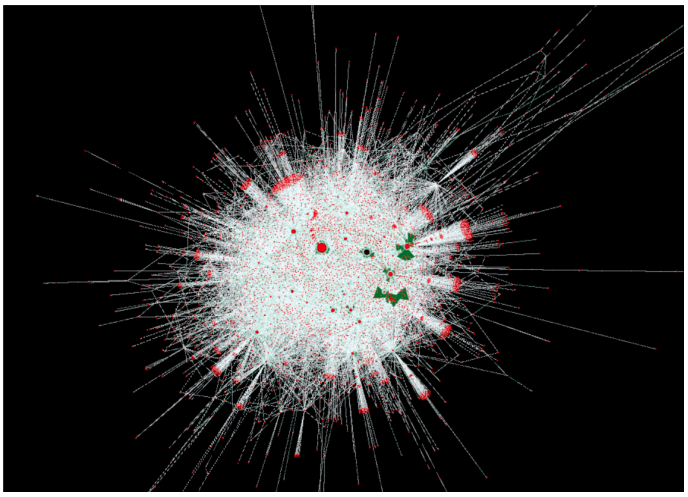
従って、fgo、gbf、ebsとivjこの4つのUserネットワークを見ると、図4.11(主にfgoのtweetを分割される) と図4.13(主にgbfのtweetを分割される) の可視化結果の中に黒色ノードは全く見えない以上、ネットワーク内の大きいノードはほぼ



(a) 次数 (degree) グループ 1

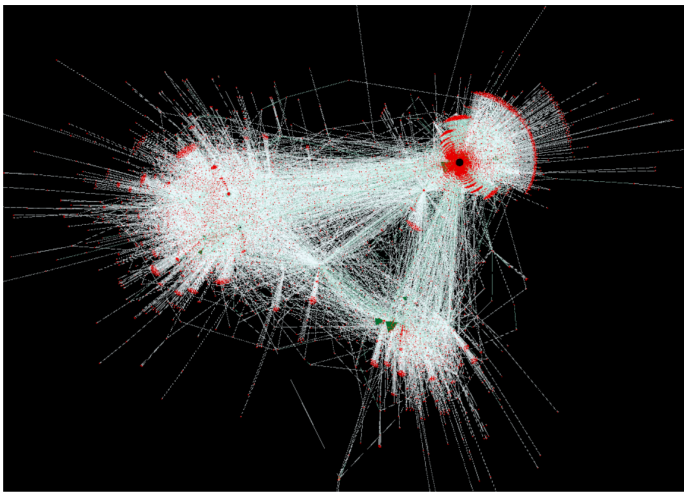


(b) Jaccard グループ 1

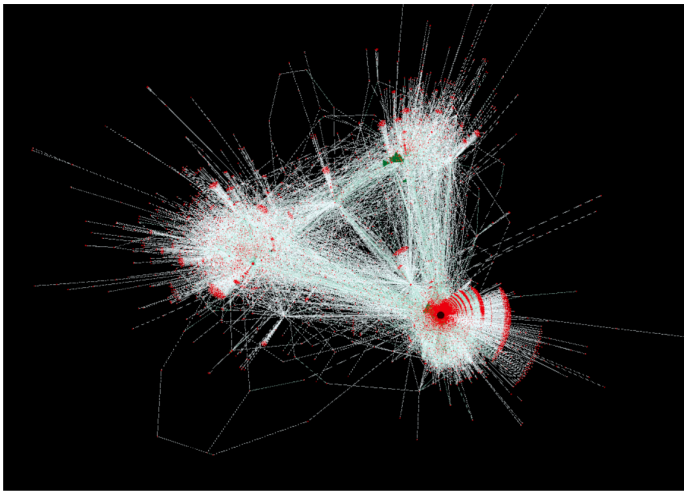


(c) 重み (weight) グループ 1

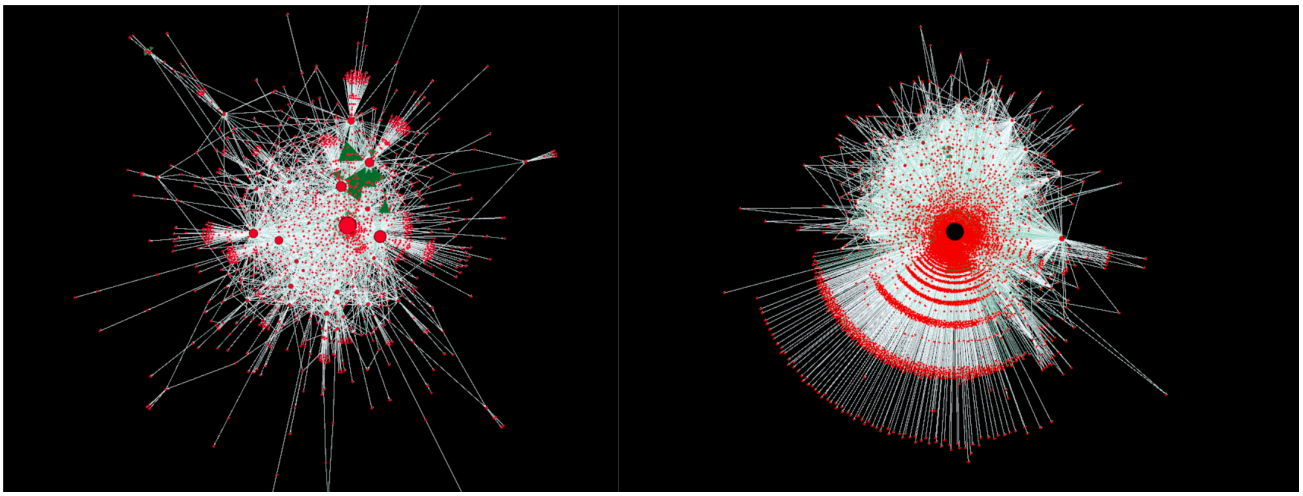
図 4.11: 可視化結果 1



(a) 次数 (degree) グループ 2



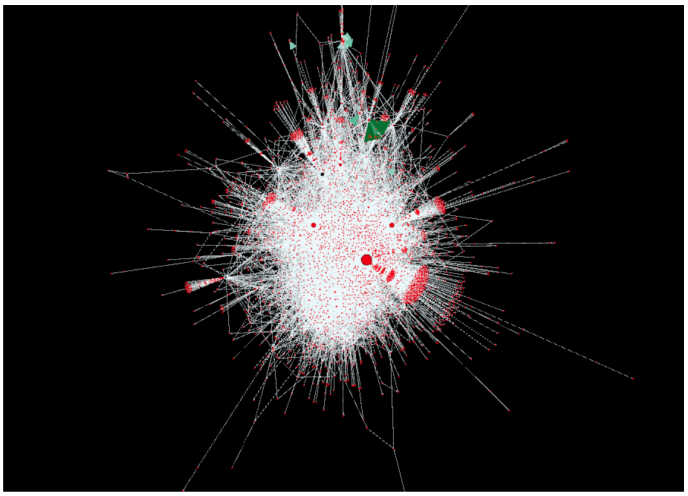
(b) Jaccard グループ 2



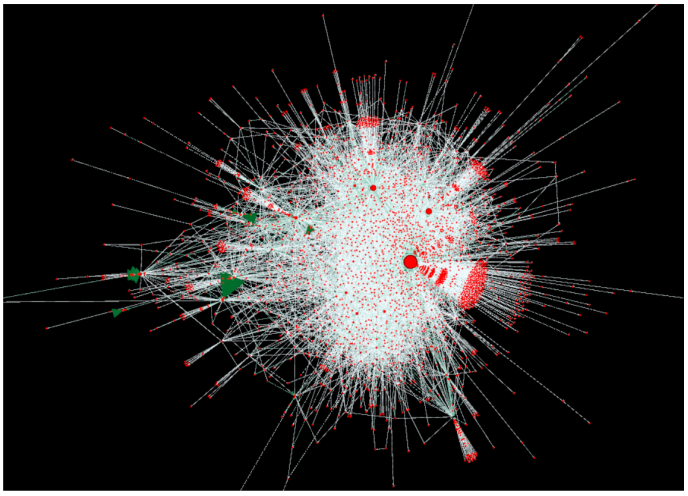
(c) 重み (weight) グループ 3

(d) 重み (weight) グループ 4

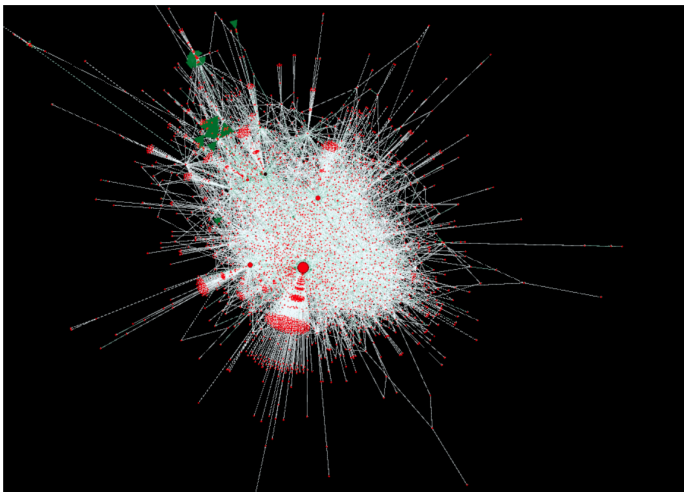
図 4.12: 可視化結果 2



(a) 次数 (degree) グループ 3

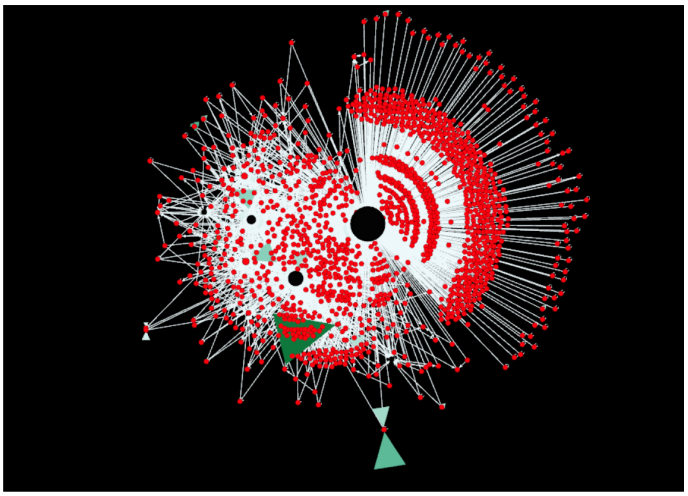


(b) Jaccard グループ 3

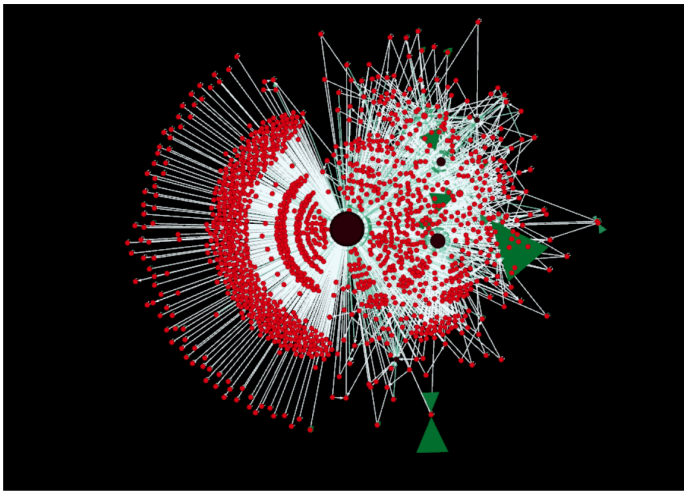


(c) 重み (weight) グループ 2

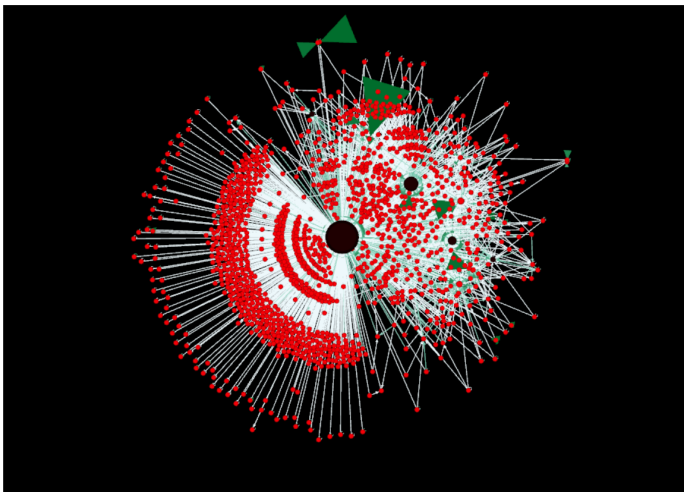
図 4.13: 可視化結果 3



(a) 次数 (degree) グループ 4

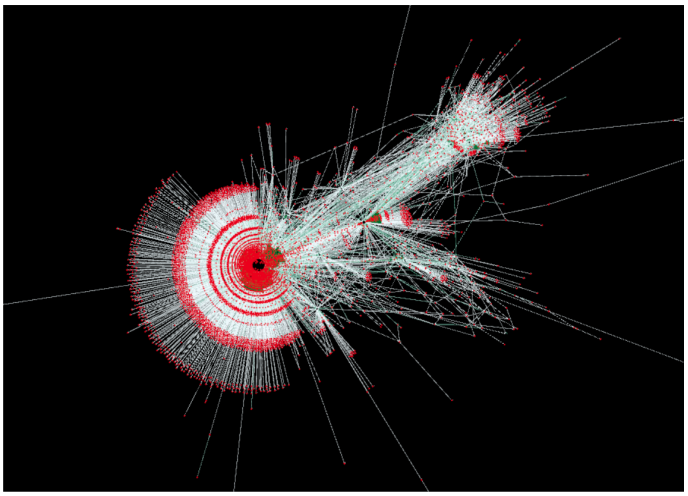


(b) Jaccard グループ 4

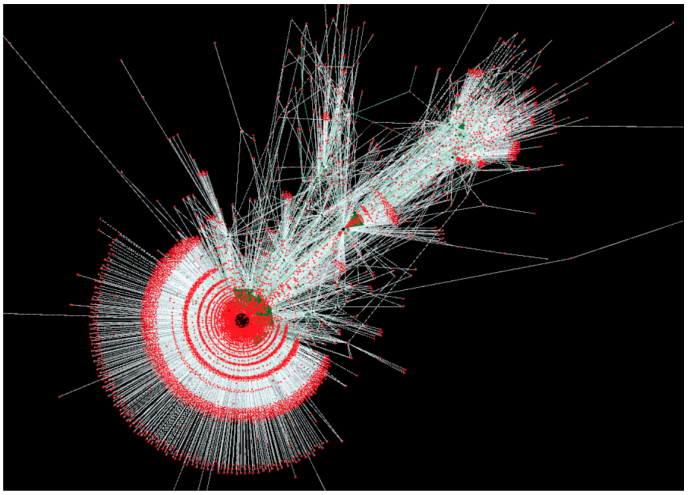


(c) 重み (weight) グループ 5

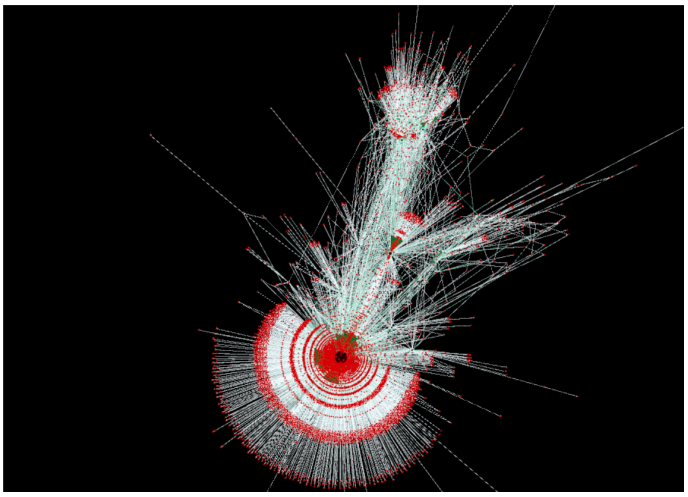
図 4.14: 可視化結果 4



(a) 次数 (degree) グループ 5



(b) Jaccard グループ 5



(c) 重み (weight) グループ 6

図 4.15: 可視化結果 5

赤色のノードである。図 4.11(主に ebs と ivj の tweet を分割される) は特例として、はっきりする黒色ノードが 1 個いる。その黒色ノードの位置は図 4.12 の (d) と合って、主に ivj の tweet を分割されるグループである。4.1.2 節の紹介にて、ivj は中国の会社が開発したスマートフォンゲームで、海外進出する為、有力な認証者 (インフルエンサー) と連携する可能性は十分ある。ここで、黒色と赤色は認証と未認証をそれぞれ表す。

図 4.14 は主に fnf の tweet を分割した User ネットワークである。認証済みの黒色ノードがネットワーク中心に存在して、他のはっきりと見える黒色ノードもある。それは ivj も同様で、有力な認証者が参入する結果である。

ftg、mcf、pug の 3 つのゲームが同じグループ分割する User ネットワーク図 4.15 に示す。図中、user 数が多く、可視化すると、ノードが小さくなっているが、1 番眼立つのはネットワーク中心には型が大きい黒色ノードが 1 個存在する。黒色ノードの数は前述のように 44 個あるで、他のグループより多いが、型が大きいのノードはあまりいないことは図 4.15 により見えてきた。

第5章　おわりに

本研究では、2019年に1番話題になっているゲームランキングに基づき、「プロジェクト版」twAwlerを用いて、Twitterのクローラを作り、前処理としてデータ管理保存からクリーニングまで行った後に、ネットワーク構造の観点から分析を行った。分析には3種類の抽出方法を利用して、二部グラフからTweetネットワークを抽出、ネットワークの分布指標と可視化を用いて、時間変化とコンミュニティ抽出からデータの特徴を議論した。以下、得られた結果をまとめる。

- Twitterにおける、APIによる大規模データ収集の可能性が示した。
- TweetネットワークのK-core分布はノード分散度の傾向と1致する。
- 次数や、k-coreの値と平均値を見ると、2020年1月から9月の変化に対して、ゲーム間にはそれ程統一的な上昇や下降傾向などはない。
- スマートフォンゲームは他のゲーム間より、繋がりが薄い一方、スマートフォンゲーム同士の繋がりが強い。
- EnsembleStarのretweeterはそのゲームに対するretweet意欲は極めて高い。
- FgoprojectとGranblueFantanyは次数分布や共通userから、それらの2つのゲームは繋がりが強い。
- IdentityVJPには女性向けのEnsembleStarに繋がりは多い一方、男性向けのFgoprojectとGranblueFantanyにも繋がりが多く存在する。ゲーマーの性別バランスは他のスマートフォンゲームよりも均衡すると考えられる。また、Userネットワークを見ると、有力な認証者がいる。
- FortniteGame 、PUBG、Minecraftをretweetするuserの中に、認証済みの人はスマートフォンゲームよりも圧倒的に多い。

本研究が収集した8個のゲームは、ただ一部分のゲームの代表であり、すべてのゲーム種類は網羅できていない。今後の課題として、収集範囲を拡大して、Tweetネットワークの分割について、他の分割方法についても検討するなどが考えられる。

参考文献

[1] Rishi Chadha. *2019 Gaming on Twitter*. URL: https://blog.twitter.com/en_us/topics/events/2019/2019-gaming-on-twitter.html. (accessed: 07.01.2020).

[2] Rdot Chadha. *2020*年上半期の *Twitter* におけるゲームと *e* スポーツに関するインサイト. URL: https://blog.twitter.com/ja_jp/topics/events/2020/gaming_esports_twitter_insights_first_half_2020.html. (accessed: 19.08.2020).

[3] Twitter Japan. *Twitter*上のゲームに関する会話の盛り上がり. URL: https://blog.twitter.com/ja_jp/topics/company/2020/April_gaming_on_Twitter.html. (accessed: 10.04.2020).

[4] Twitter マーケティング. *Twitter* にはゲーム好きが集まる. URL: https://blog.twitter.com/ja_jp/topics/marketing/2020/GamingResearch.html. (accessed: 01.05.2020).

[5] Carolina Becatti et al. "Extracting significant signal of news consumption from social networks: the case of Twitter in Italian political elections." In: *Nature Communications* 5.91 (2019). DOI: https://doi.org/10.1057/s41599-019-0300-3.

[6] Polyvios Pratikakis. "twAwler: A lightweight twitter crawler." In: (2018).

[7] Polyvios Pratikakis. *twAwler*. URL: https://github.com/polyvios/twAwler. (accessed: 01.08.2020).

[8] Vincent D. Blondel et al. "Fast unfolding of communities in large networks.." In: *IOP Science* (2008). DOI: 10.1088/1742-5468/2008/10/P10008.

[9] Mark EJ Newman and Michelle Girvan. "Finding and evaluating community struc-ture in networks.." In: *Physical review E* 69.2 (2004). DOI: 10.1103/PhysRevE.69.026113.

[10] 林幸雄 編著. *Python* と複雑ネットワーク分析—関係性データからのアプローチ— *(*ネットワーク科学の道具箱 *II)*. 近代科学社, 2019.

[11] Paul Jaccard. “THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE..” In: *New Phytologist.* 11.2 (1912). DOI: https://doi.org/10.1111/j.1469-8137.1912.tb05611.x.

[12] Shawn Martin et al. “OpenOrd: An Open-Source Toolbox for Large Graph Layout.” In: *Proceedings of the SPIE* 7868 (2011). DOI: 10.1117/12.871402.