卒業論文

マッシュアップ事例に基づく
代替サービスの推薦

指導教官　　村上 陽平 准教授

立命館大学情報理工学部
情報コミュニケーション学科 4 回生
2600160063-9

大久保 弘基

2019 年度（秋学期）卒業研究 3（2Q）

令和 2 年 1 月 31 日

マッシュアップ事例に基づく代替サービスの推薦

大久保 弘基

内容梗概

複数のサービスを組み合わせ，新しい１つのサービスを作成することをマッシュアップという．マッシュアップは本来提供されるサービス以上のサービスを提供することができ，またサービス同士はAPIで連携されることから，既存のサービスを再利用することができる．たとえば，地図情報サービスと天気情報サービスを組み合わせることで地図上に各地方の天気を表示するマッシュアップを作成することができる．このようなマッシュアップでは，組み合わされた一部のサービスが利用できなくなると，マッシュアップ全体の機能に影響が生じる．この問題に対して，WEBサービス記述ファイル（以下，WSDLドキュメントと呼ぶ）をテキストマイニングし，サービスを機能ごとにクラスタリングする研究がある．しかしながら，この手法では，WSDL ドキュメントのテキストマイニングによりサービスの機能を表す用語を特定しており，記述内容に大きく依存するため，提供者の命名規則による影響を受けやすい．

そこで，本研究ではマッシュアップのサービスの依存関係に基づく代替サービスの推薦を提案する．具体的には，マッシュアップで組み合わされたサービスの連携関係をグラフで表し，そのグラフのトポロジカルな特徴で各ノード間の類似度を計算することで，代替サービスを発見する．本手法の実現に辺り，取り組むべき課題は以下の２点である．

グラフのトポロジカルな特徴の抽出方法の評価

グラフからトポロジカルな特徴を抽出するためには，それぞれのノードの特徴を表す探索方法が必要である．無作為に探索を行うと近くのノードと遠くのノードを等価とした探索を行うので特徴は表しづらい．また利用できないサービスと共起関係のあるサービスを重視したいので，近くのノードから探索することが必要になる．

代替サービスの妥当性の基準

グラフ上の全てのノード間で類似度を計算できるため，代替サービスを決めるには，代替可能かどうかの妥当性を判定するための基準が必要である．

１つ目の課題に対しては，グラフ埋め込み技術の１つであるnode2vecを用いた類似度の計算を行なった．マッシュアップに組み合わされたサービスをノード，連携関係をエッジとしたネットワークを構築し，そのネットワークに対し

node2vec で学習を行った．次に学習結果を使い，各サービスに対応するノードのベクトル表現を生成する．生成されたベクトルを用いて，故障サービスと残りのサービス間のコサイン類似度を計算し代替候補のサービスを決定する．最後に代替サービス群に対し F 値を算出する．解集合は故障サービスと同じジャンルのサービス群である．提案手法の有効性を評価するために，ベクトル埋め込み技術を用いない，共通の近傍ノード を基準に計算するコサイン類似度，ジャカード類似度の結果と比較を行なった．

２つ目の課題に対しては，node2vec の次元数に応じて，各閾値における F 値を計算し，最適な閾値を求めた．本研究の貢献は以下のとおりである．

グラフのトポロジカルな特徴の抽出方法の評価

node2vec の探索パラメーターを変化させ,それぞれの場合の F 値を計算し，それらを比較した．その結果，探索ノードリストの長さや１つのノードに対し取得する探索ノードリストの数は値が大きい時に F 値が高くなることが判明した．また，F 値は幅優先探索に探索バイアスをかけた場合よりも，深さ優先探索に探索バイアスをかけた場合の方が高いと判明した ．

代替サービスの妥当性の基準

node2vec を用いた類似度計算の各閾値における F 値を計算した．その結果，閾値 0.6 のときに F 値が最も大きくなった．この結果を他のネットワークで検証した結果，閾値 0.6 の時の node2vec の F 値はコサイン類似度，ジャカード類似度の F 値より大きくなった．この結果に基づいて，閾値を 0.6 に設定し，適当なサービスに対し推薦を行った．その結果， F 値は 0.125 で，コサイン類似度，ジャカード類似度より 8.25%精度が向上した．

Recommendation of alternative services based on Mashup case

Koki Okubo

Abstract

Creating a new service by combining multiple services is called mashup. Mashups can provide more services than originally provided, and because services are linked by API, existing services can be reused. For example, by combining a map information service and a weather information service, it is possible to create a mashup that display the weather of each region on a map. In such a mashup, if some of the combined services become unavailable, the function of the entire mashup is affected. In response to this problem, there is research on text mining WEB service description file (Hereafter, it is called a WSDL document) and clustering services by function. However, in this method, terms that represent service functions are specified by text mining of the WSDL document, and are highly dependent on the description contents, and thus are easily affected by the naming rules of the providers.

In this study, we propose alternative service recommendation based on mashup service dependency. Specifically, an alternative service is found by expressing a cooperative relationship of services combined by mashup in a graph and calculating a similarity between each node based on a topological feature of the graph. There are two issues to be addressed in realizing this method.

Evaluation of extraction method of topological features of graph

In order to extract topological features from a graph, a search method representing the features of each node is required. If the search is performed at random, the search is performed by making the near node and the far node equivalent, so that the feature is hard to represent. Also, since it is desired to emphasize services that have a co-occurrence relationship with services that cannot be used, it is necessary to search from nearby nodes.

Alternative Service Validity Criteria

Since similarity can be calculated between all nodes on the graph, a criterion for judging the validity of the alternative service is required to determine the alternative service.

For the first problem, we calculated the similarity using node2vec, one of the graph embedding techniques. We constructed a network with the web service of mashup service

as a node and a cooperative relationship as an edge, and learned the network using node2vec. Next, using the learning result, a vector representation of a node corresponding to each service is generated. Using the generated vector, the cosine similarity between the broken service and the remaining services is calculated, and the alternative service is determined. Finally, the F value is calculated for the alternative service group. The solution set is a service group of the same genre as the broken service. In order to evaluate the effectiveness of the proposed method, we compared the results of cosine similarity and Jacquard similarity, which are calculated based on common neighboring nodes without using vector embedding technology. For the second problem, the F value at each threshold was calculated according to the number of dimensions of node2vec, and the optimum threshold was obtained. The contributions of this research are as follows.

Evaluation of extraction method of topological features of graph

We changed the search parameters of node2vec, calculated the F-value in each case, and compared them. As a result, it was found that when the length of the search node list and the number of search node lists to be acquired for one node are large, the F value increases. Also, it was found that the F value was higher when the search bias was applied to the depth first search than when the breadth first search was biased.

Alternative Service Validity Criteria

We calculated the F value at each threshold of the similarity calculation using node2vec. As a result, the F value became the largest when the threshold was 0.6. When this result was verified on other networks, the F value of node2vec at the threshold of 0.6 was larger than the F values of cosine similarity and Jacquard similarity. Based on these results, the threshold was set to 0.6 and recommendations were made for the service chosen at random. As a result, the f value was 0.125, which was 8.25% more accurate than the cosine and jacquard similarities.

マッシュアップ事例に基づく代替サービスの推薦

目次

第1章 はじめに

近年, WEB サービスにおいて, サービスコンピューティングという概念が急速に発達している. サービスコンピューティングとは, 迅速かつ柔軟なサービス開発を目標として開発された技術である. 具体例として, マッシュアップについて考える. マッシュアップとは複数のサービスを組み合わせて新しいサービスを作成することである. マッシュアップは本来提供されるサービスよりも機能が拡張されたサービスを提供することができ, またサービス同士は API で連携されることから, 既存のサービスを再利用することができる. このようなマッシュアップでは, 組み合わされた一部のサービスが利用できなくなることがある. 例を挙げると, サービスの提供中止や, サーバのメンテナンスなどである. これらの原因でマッシュアップに組み合わされた一部のサービスが利用できなくなり, マッシュアップ全体の機能に影響が生じる. この問題に対して, WEB サービス記述ファイル（以下, WSDL ドキュメントと呼ぶ）をマイニングし, サービスを機能ごとにクラスタリングする研究がある. しかしながら, この手法では, WSDL ドキュメントのテキストマイニングによりサービスの機能を表す用語を特定しており, 記述内容に大きく依存するため, 提供者の命名規則による影響を受けやすい.

そこで, 本研究ではマッシュアップのサービスの依存関係に基づく代替サービスの推薦を提案する. 具体的には, マッシュアップで組み合わされたサービスの連携関係をグラフで表し, そのグラフのトポロジカルな特徴で各ノード間の類似度を計算することで, 代替サービスを発見する. 本手法の実現に辺り, 取り組むべき課題は以下の通りである.

グラフのトポロジカルな特徴の抽出方法の評価

グラフからトポロジカルな特徴を抽出するためには, それぞれのノードの特徴を表す探索方法が必要である. 無作為に探索を行うと近くのノードと遠くのノードを等価とした探索を行うので特徴は表しづらい. また利用できないサービスと共起関係のあるサービスを重視したいので, 近くのノードから探索することが必要になる.

代替サービスの妥当性の基準

グラフ上の全てのノード間で類似度を計算できるため, 代替サービスを決めるには, 代替可能かどうかの妥当性を判定するための基準が必要である.

以下本論文では，２章において従来の研究の推薦方法について説明する．続いて，３章では隣接行列に基づく類似度として用いたコサイン類似度，ジャカード類似度のサービス依存関係グラフ，またその計算方法について説明する．その後，４章ではグラフの分散表現に基づく類似度である Network Embedding について説明を行い，node2vec を用いた類似度計算について説明する．５章では３章と４章で説明を行なった類似度計算に対する考察を行い，最後に今後の展望や課題について述べ結論とする．

第2章 代替サービスの推薦

本章では，既存の代替サービスの推薦方法について説明する．大別して，インターフェース情報に基づいた推薦方法と，オントロジーに基づく推薦方法が存在するので，それらを順に説明する．

マッシュアップが使用できなくなった場合のサンプルケースとして，表 1 のケースを用いる．

このマッシュアップが正常に動作している場合は図 1 の左側のように地図上に対応する各地方の天気に関する情報が表示される．しかし，組み合わされたサービスである地図情報サービスが使用不可になった際は，図1右側のような真っ白な背景に天気情報だけが表示される画面になる．このような使用できないサービスが発生した際に代替サービスに置き換えるために，使用できないサービスと類似する機能を持ったサービスの発見が必要となる．この問題を解決する手法としてインターフェース情報に基づく推薦方法やオントロジーに基づく推薦方法が存在する．

表 1：使用できないマッシュアップのサンプルケース

マッシュアップの機能	組み合わされたサービス	故障状況
地図上に各地方の天気情報を表示する	・地図情報サービス ・天気情報サービス	地図情報サービスが使用不可

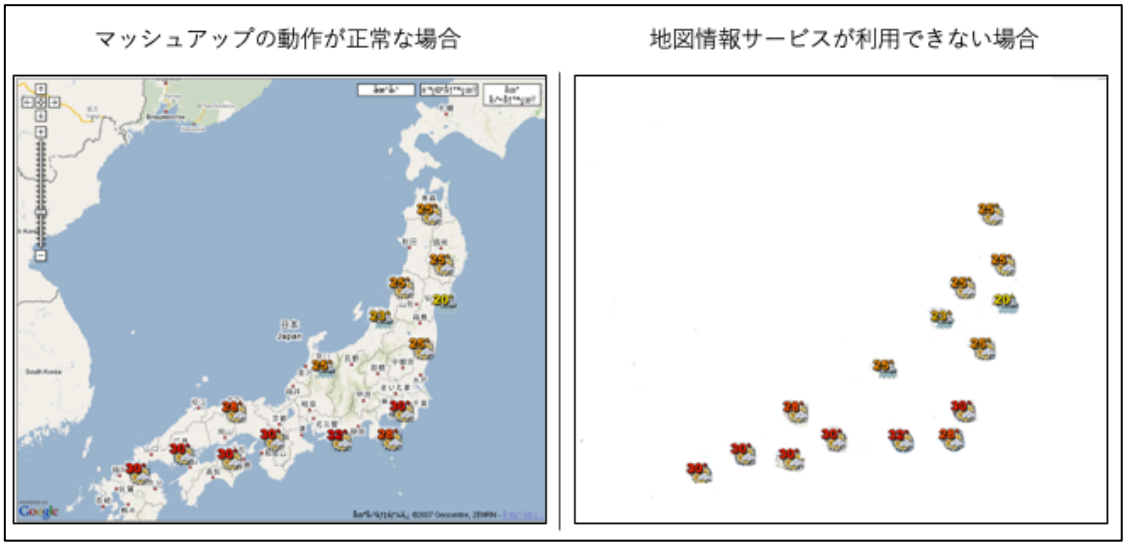

図 1：マッシュアップの動作例

2.1 インターフェース情報に基づく推薦

インターフェース情報に基づく手法において，ユーザが推薦されるものはユーザが指定したサービスの機能を表す単語に類似するものである．Elgazzar らは，WSDL ドキュメントをテキストマイニングし，それらを機能的に類似した WEB サービスグループにクラスター化する方法を提案した[1]．この提案手法の実現には以下のステップが必要になる．

1. WSDL ドキュメントからクラスターを作成する
2. クラスターからユーザの要求する WEB サービスを見つける

１つ目の WSDL ドキュメントからクラスターを作成するステップについて説明する．アルゴリズムの全体像を図 2 に示す．始めに検索エンジンのクローラーを使い，インターネット上から WSDL ドキュメントをクロールする．次に WSDL ドキュメントをテキストマイニングして，WEB サービスの意味や動作を抽出するために，ＷＳＤＬコンテンツ，ＷＳＤＬタイプ，ＷＳＤＬメッセージ，ＷＳＤＬポート，およびＷＥＢサービス名を記述する機能を抽出する．これらの機能を統合して，WEB サービスを機能的に類似したグループにクラスター化する．これはサービス検索エンジンが WEB サービスを識別し，ユーザ要求を満たす WEB サービスを発見するための処理である．

２つ目のクラスターからユーザの要求する WEB サービスを見つけるステップについて説明する．始めにユーザからサービス検索エンジンに目的の単語でクエリを実行する．次に１つ目のステップで作成したクラスターと入力されたクエリを意味的に一致させる．これは例えば，「vehicle」と「car」のような意味は同じだが単語が違うもの同士を一致させるための処理である．最後に，要求された目的を満たす最も関連性の高い WEB サービスをユーザに返す．

この手法を上記のマッシュアップのサンプルケースに適応した場合，ユーザは目的の単語を「map」とし，WEB 検索エンジンにクエリを実行する．その後，事前にクラスター化された WEB サービスから「map」に関連する WEB サービスを発見し，ユーザにその結果を提示する．しかしながら，この手法では，WSDL ドキュメントのテキストマイニングによりサービスの機能を表す用語を特定しており，記述内容に大きく依存するため，提供者の命名規則による影響を受けやすい問題がある．

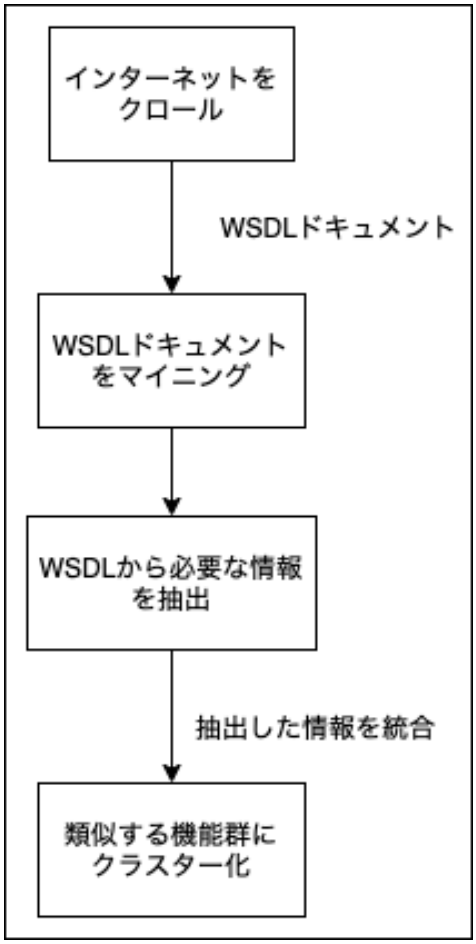

図 2：WSDL ドキュメントからクラスターを作成するフローチャート

2.2 オントロジーに基づく推薦

Xie らは，ドメインオントロジーの構造を分析し，それらを機能的に類似した WEB サービスグループにクラスター化する方法を提案した[2]. WEB サービス機能と進捗状況を使用して，類似度を計算する．オントロジーとは，概念，プロパティ，概念の様々な機能と属性，およびスロットの制限の正式な使用である．またドメインオントロジーは概念の説明とドメイン内の概念間の関係である．本節では，ドメインオントロジー階層 2 つの概念関係を計算することにより，概念間の類似性を取得する．類似サービスのクラスター化手順は以下の通りである．

1. 概念測定の深さ，経路，概念密度，アンチセンス間の測定を求める
2. 概念的な意味的類似性を求める

3. 機能とプロセスの類似性のアルゴリズムにより, WEB サービスの類似性を求める
4. WEB サービスをクラスター化する

はじめに，１の階層に基づく概念の意味的類似性の求め方について説明する．階層に基づく概念の意味的類似性の影響要因は次の通りである．

1. 階層の深さ：ドメインオントロジー階層はより深く，概念分類がより網羅的であり，階層ないの２つの概念がより類似性ていることを表す．
2. 最短経路のステップ：２つの概念のステップが多いほど，それらの距離は遠く，類似性は低くなる．
3. 概念の深さ：２つの概念がより深い場合，概念がより包括的に分析されることを意味する．
4. 概念の密度：概念の深さと同様に，概念にさらに兄弟の概念がある場合，概念がより包括的に分析されることを意味する．したがって，密度の高い概念ほど，類似性は高くなる．

上記の分析から，概念測定の深さと経路が計算される．また，概念密度はある概念と同じ概念に属する概念の数とドメインオントロジー階層ないの全ての概念の数から計算することができる．さらに，反対語の関係をアンチセンス関係といい，これは類似度の計算で肯定的な役割を果たすので，類似性の計算中に２つのアンチセンス概念の意味的類似性を減らす．

次に，２つ目の概念的な意味的類似性の求め方を説明する．これは１で求めた概念測定の深さ（Dep），経路（Path），概念の密度（p），アンチセンス間の測定（Anti）を用いることで以下のように計算できる．A, B は概念を表し，α, β, δは調整係数であり，2 つの概念のパス，密度，およびアンチセンス関係の異なる重みを表す．

$$Sim(A,B) = Dep(A,B) \times (\alpha Path(A,B) + \beta p(A,B) + \delta Anti(A,B))$$

続いて，３つ目の機能とプロセスの類似性のアルゴリズムにより，WEB サービスの類似性の求め方を説明する．WEB サービスの類似性を求めるには機能の類似性とプロセスの類似性をそれぞれ求める必要がある．定義を下記に示す．

WSA, WSB：サービスの説明で，名前と記述内容を含む

AStr, Bstr：プロセスの概念文字

AC, BC：WSA または WSB から概念を抽出したもの

プロセスの類似性は２つ目で用いた計算の引数に Astr と Bstr を入力すること

で求める．機能の類似性を求めるためには入出力の一致度とサービス記述内容の類似性を求める必要がある．入出力の一致度は WSA と WSB を用いることで求めることができる．サービス記述内容の類似性は２つ目で用いた計算の引数に AC, BC を入力することで計算できる．最後に，サービス類似度は機能類似度とプロセス類似度を足し合わせることで計算できる．

最後に４つ目の WEB サービスをクラスター化する手法については，３つ目までに求めた WEB サービスの類似度と k-means を使用することで類似サービスグループにクラスター化することができる．

しかしながら，この手法では類似度アルゴリズムで密度と関係の重みが考慮されるため，オントロジーの構造を正確に構築する必要がある．

第3章 隣接行列に基づく類似度

本章では，コサイン，ジャカード類似度の計算で用いるネットワークの定義とその計算方法について説明する．ネットワークの作成に用いたデータは，WEB サービスやマッシュアップを掲載したサイトである Programmableweb[1]のデータを用いた．

3.1 サービス連携の依存グラフ

Programmableweb から取得したデータには，マッシュアップサービスと組み合わされているサービスが記述されている（図 3）．このフォーマットの定義を図 4 に示す．取得したデータから target_id のみを取得し，ノードを target_id とし，エッジを組み合わされたことのあるサービス間で繋ぐグラフを構築する．具体的には，図 5 のようなグラフを作成する．このグラフでは図 3 のマッシュアップで target_id が”62687”と”62697”が組み合わされていたことから，この２つの target_id 間でエッジを繋いでいる．また他のマッシュアップ例で”62687”と target_id である”10000”と”20000”が組み合わされていた場合は図 5 のように組み合わされたサービス同士でエッジを繋ぐ．

```
[“items”, {“vid”:“81386”, “uid”:“61422”, “title”:“News Map”, “type:“mashup”},
        [“fieldapi”,
                [“target_id”, “62687”],
                [“target_id”, “62697”]],
        [“fieldapiendpoint”], [“filedmashupurl”]]
```

図 3：Programmableweb から取得したデータの一例

```
[“items”, {“vid”:サービスID, “uid”:ユーザI D, “title”:サービス名,
“type:マッシュアップなら”mashup”, WEBサービスなら”api”},
        [“fieldapi”,
                [“target_id”, 組み合わされたサービスI D],
        [“fieldapiendpoint”], [“filedmashupurl”]]
```

図 4：Programmableweb から取得したデータのフォーマット

[1] https://www.programmableweb.com

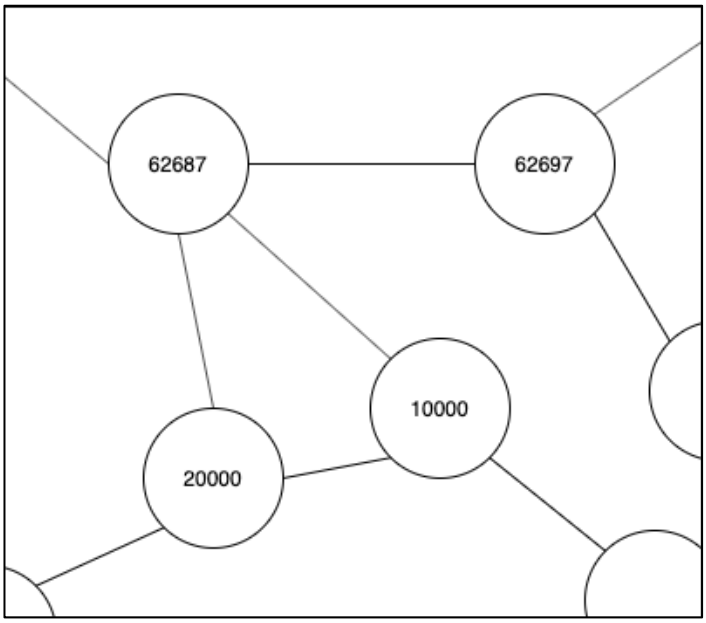

図 5：ネットワーク例

3.2 コサイン，ジャカード類似度を用いた類似度計算

サービス間の類似度の計算にはコサイン類似度とジャカード類似度を用いる．コサイン類似度では，２つのベクトルの角度の近さで類似度を算出する．具体的には，２つのベクトルを A, B とすると以下の式によって算出できる．

$$cos(A,B)\frac{A\cdot B}{|A||B|}$$

またジャカード類似度では，２つの集合同士の和集合の割合で類似度を算出する．具体的には，２つの集合を U, V とすると以下の式によって算出できる．

$$jaccard(U,V)\frac{|U\cap V|}{|U\cup V|}$$

それらを用いて類似度計算を行うアルゴリズムを図 6 に示す．コサイン，ジャカード類似度を用いた類似度計算では，まず 3.1 で述べたネットワークを構築し，全サービスの隣接ノード群を取得する．図 5 を用いると”62697”の隣接ノード群は”62687”と”10000”と”20000”になる．次に使用不可と仮定されたサービスの隣接ノード群と各サービスの隣接ノード群からコサイン類似度，またはジャカード類似度の計算を行う．最後に，類似度の計算結果を降順に並び替えて類似度計算結果として出力する．

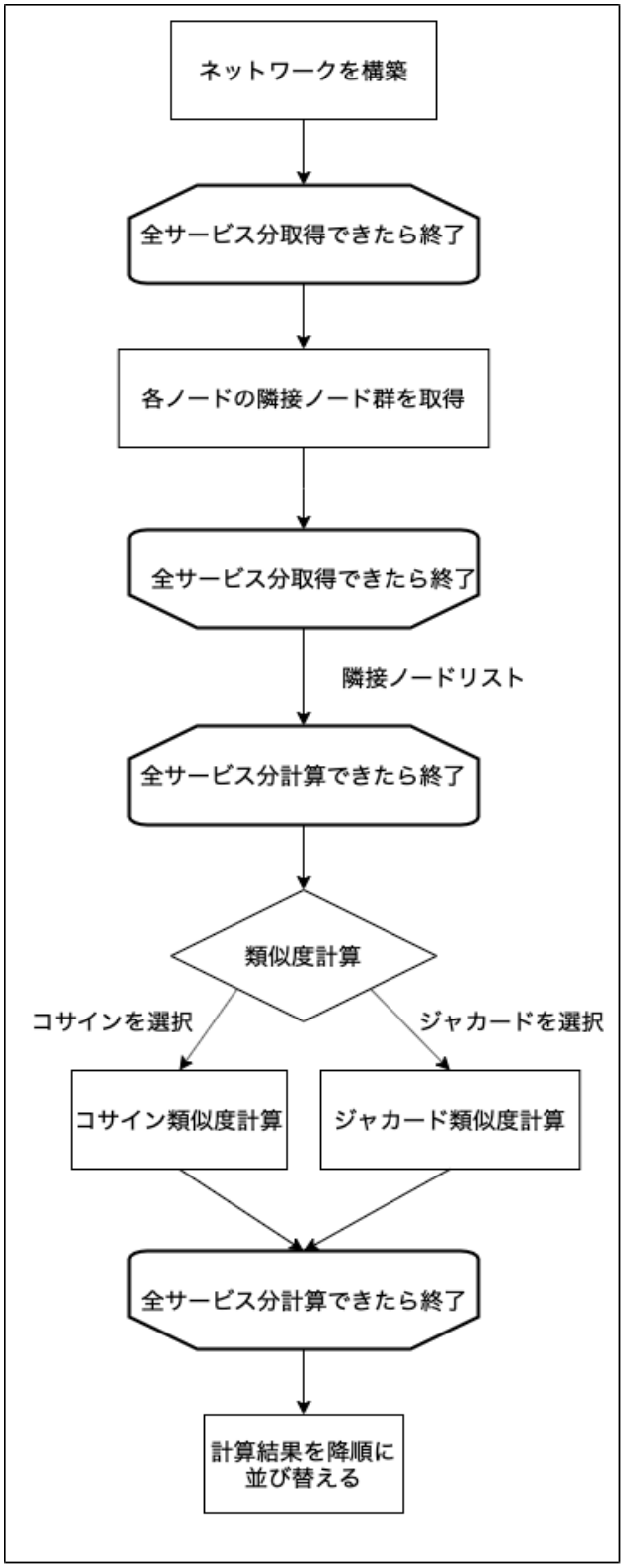

図 6：コサイン，ジャカード類似度計算のフローチャート

第4章 グラフの分散表現に基づく類似度

4.1 Network Embedding

Network Embedding とはネットワークデータのノードやエッジ，サブグラフなどの分散表現を得る手法のことであり，特に Skip-gram Model をネットワーク構造に対して拡張した手法がいくつか提案されている．本節では，Skip-gram Model のネットワーク構造への拡張手法を説明した後，Network Embedding の手法の１つである node2vec を説明する．

4.1.1 Skip-gram Model

Skip-gram Model[3]とはニューラルネットワークモデルの１つで，単語の分散表現を得る学習モデルのことである．単語の分散表現の獲得には，意味が近い単語は周辺の単語も似ているという学習より実現する．例えば，「私は琵琶湖線の南草津駅周辺に住んでいる」と「京都の山科には琵琶湖線と湖西線が走っている」という２つの文章があるとする．南草津と山科に注目した時，この２つの単語は共に琵琶湖線という単語から近い位置にあることから，この２つの単語は比較的近い属性であると予測することができる．この時の南草津や山科のような単語を入力データ，琵琶湖線のような周辺単語を教師データとし，単語に対するその周辺単語の重みを学習させることで各単語の分散表現を得ることができる．この手法をネットワーク構造に対応させると，入力データは各ノード，教師データは各ノードのサンプリングで表すことができる．

4.1.2 node2vec

node2vec[4]とはランダムウォークでノードのサンプリングを行う際，幅優先探索と深さ優先探索の度合いをハイパーパラメーターで制御するサンプリング方法である．ランダムウォークは静的エッジの重みに基づいて次のノードをサンプリングするが，ネットワークの構造を考慮し，検索手順をガイドして異なるタイプの近隣ネットワークを探索することはできない．さらに，幅優先と深さ優先が競合または排他的ではない．そこでパラメーターreturn parameter（以下，p と呼称）および In-out parameter（以下，q と呼称）を調節してサンプリングを行う．これらを使用した２次のランダムウォーク を下記に定義する．またエッジ（t, x）を横断し，次に探索するノードs_i(i=1, 2, 3)を考える（図 7）．d_{tx}はノード t と x 間の最短経路距離であり，d_{tx}は{1, 2, 3}のいずれかである．

$$\alpha_{pq}(t,s) = \begin{cases} \frac{1}{p}\ if\ d_{ts} = 0 \\ 1\ if\ d_{ts} = 1 \\ \frac{1}{q}\ if\ d_{ts} = 2 \end{cases}$$

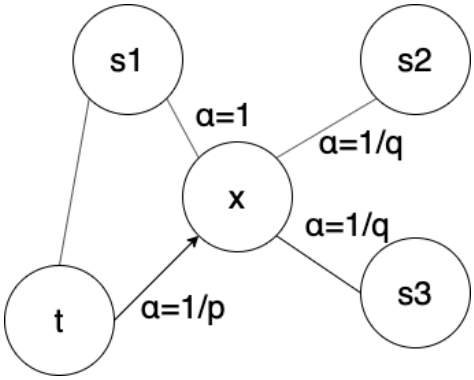

図 7：ノードの遷移例

パラメーターp は歩行中のノードを再訪する確率を制御する．このパラメーターを高い値（>max(q, 1)）に設定することで既に探索したノードをサンプリングする可能性を低くすることができる．一方，低い値（<min(q, 1)）にすることでサンプリングを開始ノードから離れにくくすることができる．

パラメーターq は幅優先探索と深さ優先探索の確率を制御する．q>1 の場合はランダムウォークはノードt に近いノードに偏り，幅優先探索に似た探索を行うことができる．対照的に，q<1 の場合はノード t からさらに離れたノードを訪問する傾向があり，深さ優先探索に似た探索を行うことができる．しかし，幅優先探索を重視しすぎると近いノードの情報しか得ることができず，一方で深さ優先探索を重視しすぎるとサンプリングの範囲が大きく深くなるので，この２つのバランスが重要である．

4.2 node2vec を用いた類似度計算

ここでは，node2vec を用いた類似度の計算について説明する．アルゴリズムの全体像を図 8 に示す．node2vec を用いた類似度計算では，まず 3.1 のようなネットワークを構築した後，そのネットワークに対し，使用不可と仮定したノード S_i からランダムウォークで探索を行う．次に，node2vec の探索パラメーターを設定する．続いて，Skip-gram Model で学習するために，S_i を入力データ，先程

の探索で取得した探索ノードリストを教師データとして重みを学習することで S_i 分散表現を獲得する．その後, S_i の分散表現と各サービスの分散表現からコサイン類似度を計算する．最後に，その結果を降順に並び替え, node2vec を用いた類似度計算結果として出力する．

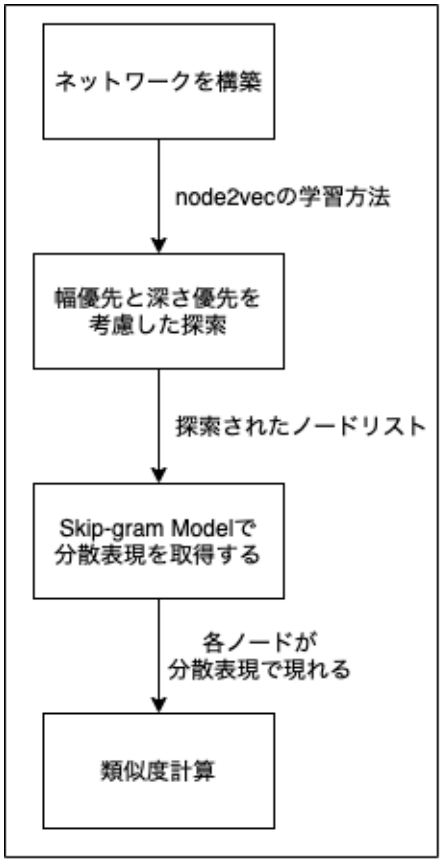

図 8：node2vec を用いた類似度計算のフローチャート

第5章 評価

第４章で説明したシステムを用いて使用不可と仮定したサービスの類似サービスを推薦する．本章では，はじめに実験データの説明をする．次に，それぞれ手法を比較する際に使用する評価指標について説明する．続いて，node2vec の探索パラメーター変化による精度の比較について説明する．最後に，最適な閾値について説明した後，その結果について考察する．

5.1 評価手法

5.1.1 実験データ

実験データは Programmableweb に掲載されているマッシュアップデータ 8970 件を用いる．これらのマッシュアップそれぞれに組み合わされているサービスを抽出し，3.1 節で説明したネットワークを作成する．その作成したネットワークのそれぞれのノードに対し，node2vec およびコサイン類似度，ジャカード類似度を用いた類似サービスの推薦を行う．

5.1.2 評価指標

node2vec とコサイン類似度，ジャカード類似度を用いた推薦結果を比較する際の評価指標として F 値を求めた．F 値は以下の式によって算出した．また F 値を求める際に使用した Recall と Precision の定義も以下に示す．正解群は使用不可と仮定されたサービスと同じジャンルのサービス群，推薦結果群は推薦された類似サービス群である．

$$F = \frac{2 * Recall * Precision}{Recall + Precision}$$

$$Recall = \frac{\text{正解群} \cap \text{推薦結果群}}{\text{正解群}}$$

$$Precision = \frac{\text{正解群} \cap \text{推薦結果群}}{\text{推薦結果群}}$$

5.1.3 node2vec の探索パラメーター変化による精度の比較

node2vec の探索パラメーター変化による精度の比較を行う．node2vec の探索パラメーターは１回のウォークの探索数（以下，walk_length と呼称），１つのノードあたりのウォーク数（以下，num_walk と呼称），4.1.2 で説明した p と q の

４つである．これらの探索パラメーターを変化させ，それぞれパラメーターがいくつの時に F 値が高くなるか検証した．

5.1.4 閾値の最適化

5.1.3 で求めた最適なパラメーターを用いて，全サービスに対し類似サービスを推薦する．それぞれの閾値の F 値を計算，比較し，最適な閾値がいかほどか検証する．ここでは，ネットワークを２つに分け，片方で最適な閾値を求め，もう片方のネットワークでその結果が有効であるか検証する．

5.1.5 コサイン類似度，ジャカード類似度を用いた推薦との精度比較

5.1.3 と 5.1.4 で求めた最適なパラメーターと閾値を用いて実際にサービス推薦を行う．適当なサービスに対し，node2vec を用いたサービス推薦を行い，その結果を，コサイン類似度，ジャカード類似度を用いたサービス推薦の結果と比較し，F 値を使って推薦精度がどのくらい違うか検証する．

5.2 結果

5.1 節で説明した評価手法を用いて node2vec を用いた類似度計算の評価を行った結果を以下に示す．

1. node2vec の探索パラメーター変化による精度比較

   はじめに，パラメーターwalk_length を変化させた結果を図 9 に示す．この結果から，walk_length の値が高くなるほど，F 値が高くなることが見て取れる．

   次に，パラメーターnum_walk を変化させた結果を図 10 に示す．この結果から，num_walk の値が高くなるほど，F 値が高くなることが見て取れる．

   続いて，パラメーターp を変化させた結果を図 11 に示す．この結果から，p は 1 に近いほど F 値が高くなることがわかる．

   最後に，パラメーターq を変化させた結果を図 12 に示す．この結果から，q は q<1 の値を取る時，F 値が高くなることがわかる．

   以上のことから，node2vec の探索パラメーターは walk_length と num_walk は高い値の時，p は値が 1 に近い時，q は値が q<1 を取る時に推薦精度が高くなることがわかった．今回の検証から，node2vec のパラメーターを walk_length=100, num_walk=600, p=1, q=0.6 で設定すると推薦精度を高くすることができると考えられる．この設定を用いて，最適な閾値を検証，またコサイン類似度，ジャカード類似度と推薦精度の比較を行う．

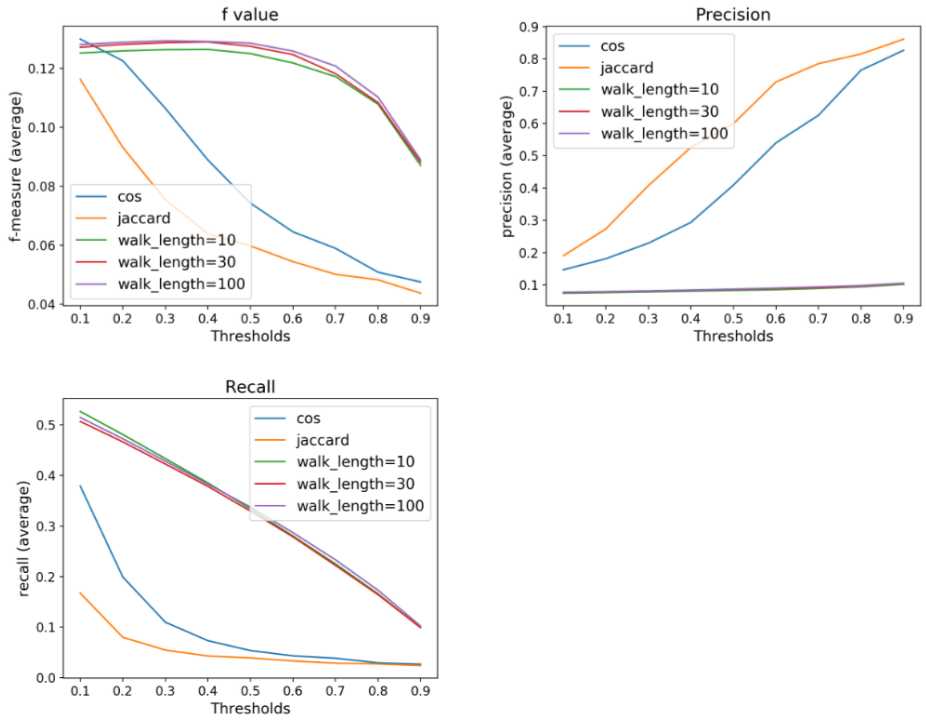

図 9：node2vec (walk_length=10, 30, 100)の精度

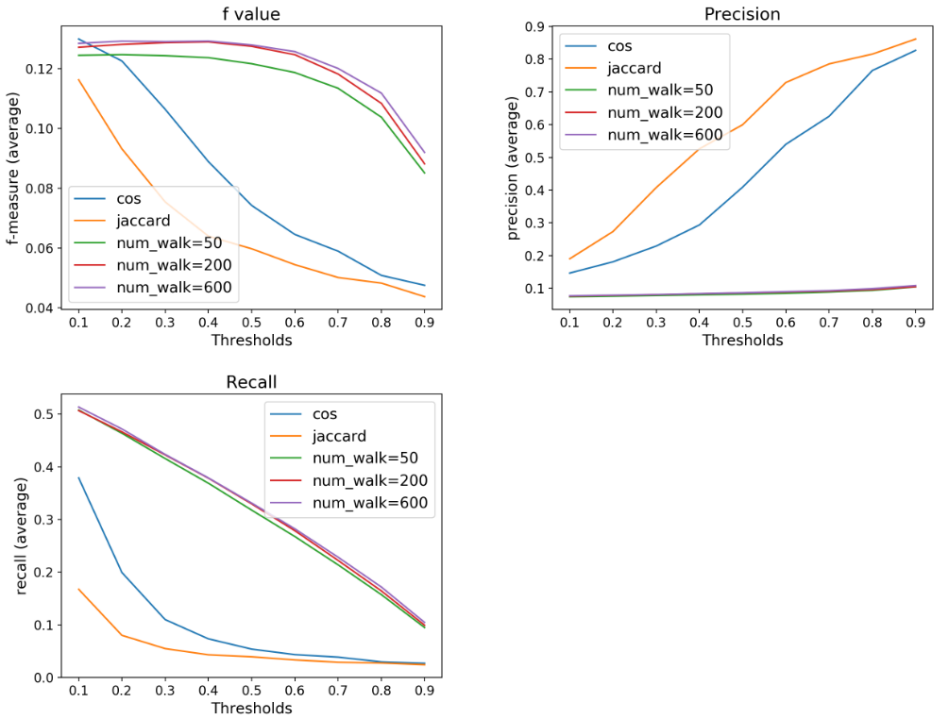

図 10：node2vec (num_walk=50, 200, 600)の精度

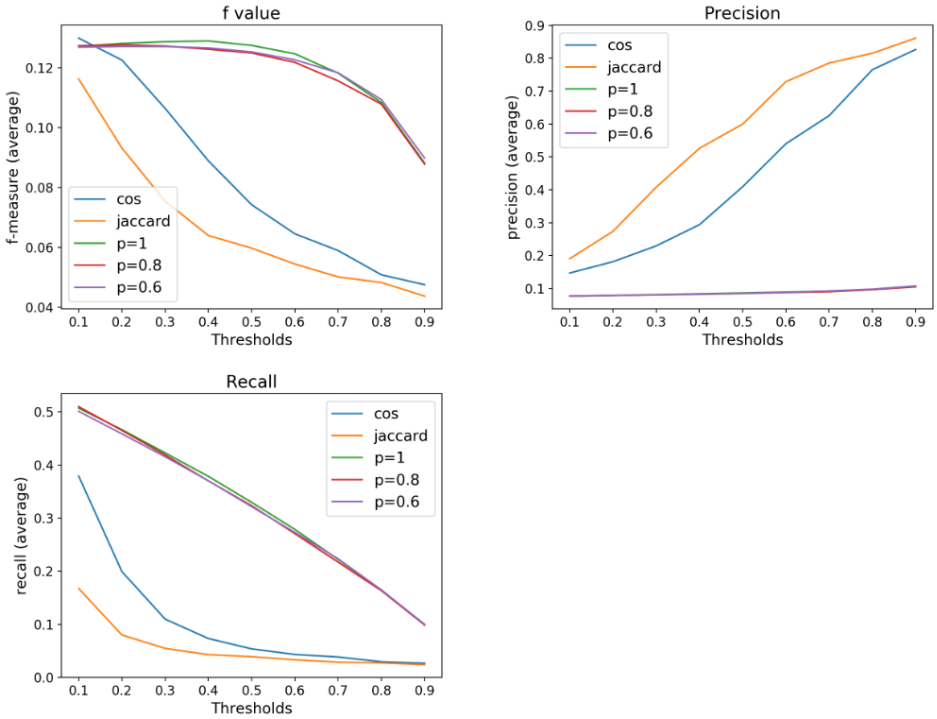

図 11：node2vec (p=1, 0.8, 0.6)の精度

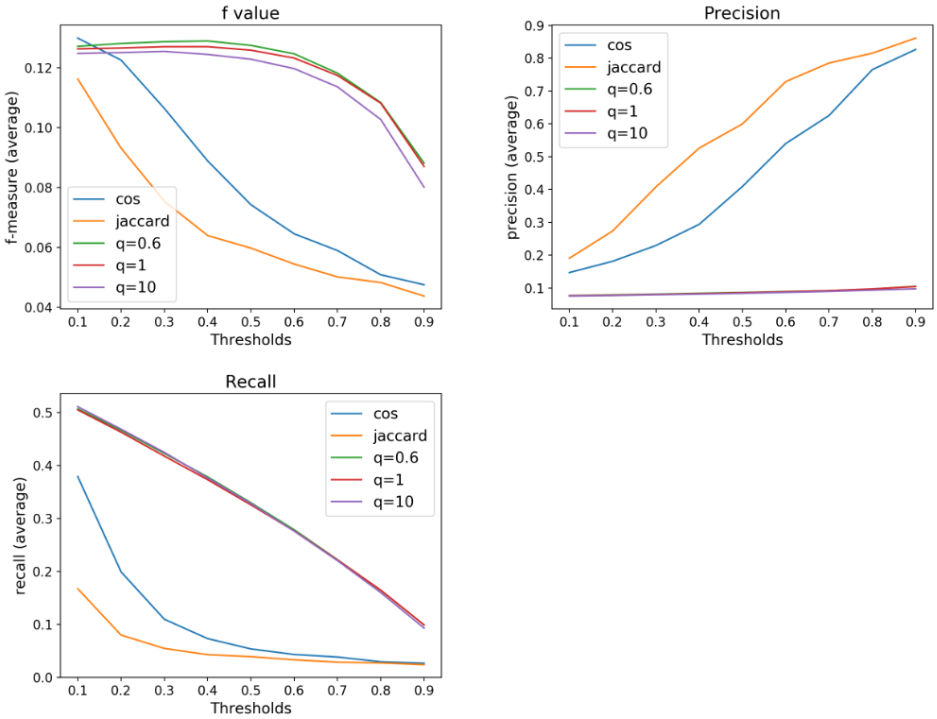

図 12：node2vec (q=0.6, 1, 10)の精度

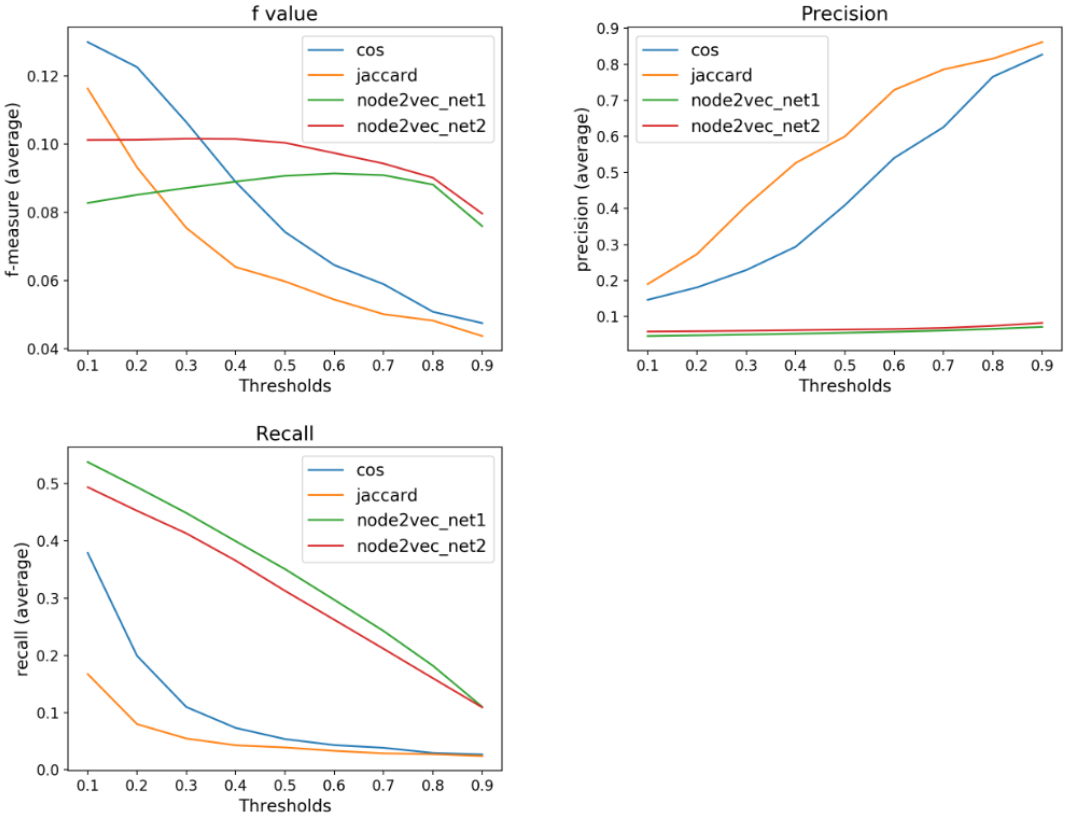

図 13：２つの node2vec のネットワークを比較

2. 最適な閾値

node2vec における最適な閾値を求めるために，ネットワークを２つに分け,それぞれを node2vec_net1 と node2vec_net2 と呼称する（図 13）. node2vec のパラメーターは先ほど述べたように，walk_length=100, num_walk=600, p=1, q=0.6 である．図 13 の F 値のグラフの node2vec_net1 を見ると，類似度 0.6 の時に node2vec の F 値が高くなる，すなわち推薦精度が高くなることが見て取れる．このことから，最適な閾値は閾値 0.6 であると仮定する．次に，図 13 の F 値のグラフの node2vec_net2 を見ると，閾値 0.6 の時の F 値はコサイン類似度，ジャカード類似度より F 値が高くなり，閾値は有効であるとわかる．以上のことから node2vec における最適な閾値は 0.6 である．

3. コサイン類似度，ジャカード類似度を用いた推薦との精度比較

これまでに述べた最適な探索パラメーター，閾値を用いた node2vec の推薦結果とコサイン類似度，ジャカード類似度を用いた推薦結果を図 14 に示す．

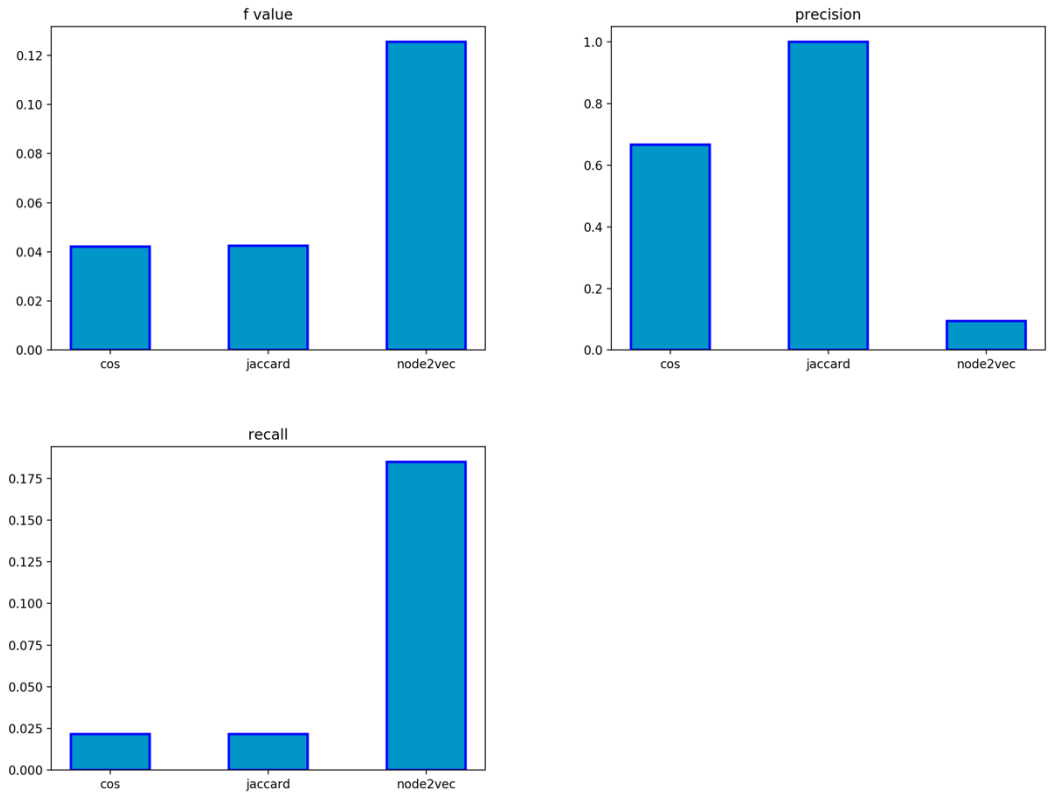

図 14：最適値を用いてサービス推薦を行った際の精度比較

図 14 の F 値を見ると, node2vec を用いた手法の F 値は 0.125 をとり，コサイン類似度，ジャカード類似度を用いた手法と比べると 8.25%向上することがわかる．このことから node2vec を用いたサービス推薦の手法はコサイン類似度，ジャカード類似度を用いたサービス推薦手法と比べると有効であることがわかる．

5.3 考察

本節では, 5.2 節で提示した結果を受けて node2vec の探索パラメーター変化による精度に関して考察を与えた後, node2vec を用いたサービス推薦の有効性について考察を行う．まず，探索パラメーター変化による精度の考察について述べる．

はじめに , walk_length と num_walk についての考察を述べる．図 9，図 10 の結果より, walk_length と num_walk は数値が高いほど推薦精度は上がると述べた．しかし，数値が大きくなるほど F 値の変化は小さくなっていることもわか

る．このことから，walk_length と num_walk の値を無限に大きくしていった時，ある値で F 値の上昇が止まる，もしくは F 値が下がることが考えられる．また，どの値をとる時に F 値が最大になるかを検証するよりも，最低でも値はいくら以上に設定すると F 値は大体同じになるかを検証することが必要になると考えられる．

次に，パラメーターp についての考察を述べる．図 11 の結果から，p は 1 に近いほど F 値は高くなることがわかった．つまり，既にサンプリングしたノードを探索しない方が推薦精度は良くなることを意味している．このことから，探索ノードリストを生成する際，同じノードを重複させないことが推薦精度の向上につながると考えられる．

最後に，パラメーターq についての考察を述べる．図 12 の結果から，q が 1 より小さい値をとる時に F 値が高くなることがわかった．つまり，幅優先探索に探索バイアスをかけるよりも，深さ優先探索に探索バイアスをかけた方が推薦精度は良くなることを意味する．これは同じネットワークで，同じ探索数で幅優先探索と深さ優先探索を行った際，深さ優先探索の方がより多くのノードとリンクしていることが示せるからであると考える．さらに 1 つのノードから複数の探索ノードリストを生成することを考えた際，幅優先探索の探索ノードリスト重複するノードが多いが，深さ優先探索は開始ノードに近いノードを網羅しつつ，より遠くのノードとの関係も表すことができる．つまり，深さ優先探索の方が探索した結果得られる情報量が多いので，ネットワークのトポロジカルな特徴を表現することができ，同じ用途で使われるサービスを推薦することが可能になったと考える．

続いて，node2vec を用いたサービス推薦の有効性についての考察を述べる．図 14 の結果から node2vec を用いたサービス推薦はコサイン類似度，ジャカード類似度を用いた場合よりも推薦精度が良いことがわかった．しかし，図 14 の値だけでは，node2vec を用いた推薦が実用的か測ることができない．そこで，この時の推薦件数，推薦件数内で故障サービスと同じジャンルのサービス数，同じジャンルの総数を調査した．その結果，推薦件数は 179 件，推薦件数内の故障サービスと同じジャンルのサービス数は 17 件，同じジャンルの総数は 92 件であった．このことから，推薦されたサービス群はユーザにとって有用ではないことがわかる．

以上のことから，以下の結論が導ける．node2vec を用いた推薦方法は，コサイ

ン類似度，ジャカード類似度を用いた推薦に比べると有用である．しかし，ユーザにとって有用なサービスを推薦できているかという点に関しては改善の余地が存在する．

第6章 おわりに

本研究では，従来の WSDL ドキュメントやオントロジーを用いた推薦の代わりに，マッシュアップのサービスの依存関係に基づく代替サービスの推薦を提案した．そして，node2vec のパラメータは探索の長さ，１つのノードに対し探索する回数は値を増やすほど推薦制度は向上し，探索の仕方は幅優先探索よりも深さ優先探索に探索バイアスをかけると推薦制度が向上することを示した．また，サービス推薦における node2vec を用いた推薦はベクトル埋め込み技術を用いない，共通の近傍ノードを基準に計算するコサイン類似度，ジャカード類似度よりも有効に作用することを示した．

本研究の貢献は以下の２点である．

グラフのトポロジカルな特徴の抽出方法の評価

node2vec の探索パラメーターを変化させ，幅優先探索または深さ優先探索に探索バイアスをかけた場合とランダムウォークの場合を比較するために，それらを用いた類似度計算の F 値を計算した．その結果，深さ優先探索に探索バイアスをかけた場合が他２つよりも F 値が高いことが判明した．

代替サービスの妥当性の基準

node2vec を用いた類似度計算の各閾値における F 値を計算した．その結果，閾値 0.6 のときに F 値が最も大きくなった．この結果を他のネットワークで検証した結果，閾値 0.6 の時の node2vec の F 値はコサイン類似度，ジャカード類似度の F 値より大きくなった．この結果から閾値を 0.6 とし，適当なサービスに対し推薦を行ったところ，F 値は 0.125 で，コサイン類似度，ジャカード類似度より 8.25%精度が向上した．

今後， 実世界において実際に node2vec を用いて類似サービスを推薦することを考えた場合，正解であるサービスの割合が推薦候補に多いことが望まれる．なぜなら推薦結果を受け取ったユーザーはどれが代替サービスとして機能するか確認する際に， １つ１つのサービスの動作を確認する必要があるからである．本手法の推薦結果は， 推薦結果数に対し正解数が少ないことが見受けられた．この問題を解決する手法として， MAP（Mean Average Precision）[5]や MRR（Mean Reciprocal Rank）を用いてランキングから類似サービスを推薦することや，マッシュアップの情報を組み込んだ２部グラフと本手法をきみ合わせて類似サービスウィ推薦することが考えられる．また， 本手法と従来の推薦方法

と推薦精度の比較し，どのくらい精度が違うのか調査する必要がある．

謝辞

本研究を行うにあたり，熱心なご指導，ご助言を賜りました指導教官の村上陽平准教授に深謝申し上げます．また普段からお世話になっている社会知能研究室の皆さまにも感謝の意を表します．

参考文献

[1] Khalid Elgazzar, Ahmed E. Hassan and Patrick Martin: Clustering WSDL Documents to Bootstrap the Discovery of Web Service, Proceedings of IEEE International Conference on Web Services, pp. 147-154 (2010)

[2] Ling-li Xie, Fu-zan Chen, Ji-song Kou: Ontology-based semantic web service clustering, Proceedings of IEEE 18th International Conference on Industrial Engineering and Engineering Management, pp. 2075-2079 (2011)

[3] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, Jeff Dean: Distributed Representations of Words and Phrases and their Compositionality, Proceedings of Neural Information Processing Systems, pp.3111-3119 (2013)

[4] Aditya Grover, Jure Leskovec: node2vec: Scalable Feature Learning for Networks, Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 855-864 (2016)

[5] Stephen Robertson: A new interpretation of average precision, SIGIR, pp.689-690(2008)