自然言語処理的アプローチによるクラス図関連線の予測

井原　輝人
奈良工業高等専門学校　専攻科 *
akiteruto@gmail.com

内田 眞司
奈良工業高等専門学校
uchida@info.nara-k.ac.jp

飯田　元
奈良先端科学技術大学院大学
iida@itc.naist.jp

要旨

ソフトウェア品質特性 *[1]* の一つに理解性がある．理解性とは，成果物を読み手が容易に理解できるかを指す．*UML* 図における理解性を定量的に表現できれば，評価値に基づいた設計品質の評価や向上に繋がると期待される．しかし，理解性は個人の感覚や経験則に依存する部分が大きく，定量化する実用的な手法は存在していない．中村 *[2]* らは読み手の理解性が図の構成要素個々の妥当性判断に基づいて行なわれていることに注目し，妥当性を定量的に評価するメトリクスの定式化を試みた．しかし，*UML* 図においては個々の図形要素が複雑に関連付けられ得るため，理解性に影響を与える要素とそれらの関係性をすべて定式化することは極めて困難である．

よって我々は同様の目的に対して機械学習を用いた妥当性の定量化を提案する．本研究ではその第１段階として，関連線の種類の予測を行うモデルを作成した．被験者実験を実施し，モデルの予測結果と，被験者の判断結果を比較したところ，高い精度で一致したことから，予測モデルは個々の関連線の人間による関連線に対する妥当性評価を摸倣可能であるといえ提案手法で関連線の種類において理解性の定量化が可能であることが明らかとなった．関連線の種類や命名の妥当性を予測モデルに基づいて *UML* 図の理解性について部分的に定量的な評価可能となることを示した．今後，他の要素も含めた予測モデルへと拡張をすることで *UML* 図 *(*クラス図）全体の理解性評価手法に結びつけられると期待される．

*2021 年 4 月より奈良先端科学技術大学院大学

1. 序論

ソフトウェアの開発工程において，開発者間の相互理解を目的とした統一モデリング言語である UML が幅広く利用されている．特に，開発の上流工程では，UML を用いた設計や分析モデルの記述が行われ，その開発成果物を基に以降の実装や検証が進められる．一般に設計文書の品質が以降の工程に与える影響は大きく，UML 図の品質評価はソフトウェア品質の確保に重要であると言える．

品質特性 [1] とはソフトウェアに求められる要求を分類したものである．品質特性の 1 つである理解性とは成果物を読み手が容易に理解できるかを表す．開発者間の相互理解を目的とする UML 図において理解性は重要な品質特性であり，当然理解性の高い UML 図は良い UML 図とされる．

UML 図における理解性を定量的に表現できれば以下のことが可能となる．

- リファクタリング効果の可視化や，理解性のボトルネック検出によるリファクタリング補助
- 理解性を報酬とした強化学習モデルによるリファクタリングの自動化
- 機械学習を用いた UML 図の自動生成における特徴量としての活用

しかし，理解性は個人の経験則や主観に依存する部分が大きく，定量的に評価する実用的な手法は存在していない．中村ら [2] は UML 図の 1 つであるクラス図に対

して理解性の定量化に取り組んだ．彼らは読み手が図の構成要素に対して妥当性評価を行うことで理解性判断を行っていることに注目し，妥当性をメトリクスとして定式化することで定量化を試みた．彼らはクラス図内に存在するクラス名の類似度が高いほど，読み手はクラス名の間の関係を強く感じ，そのようなクラス名の関係が強いクラスがクラス図上で近くに存在していると理解性が高いという仮説を立て，その仮説に基づいて理解性メトリクス CRSU を定義した．クラス名の類似度計測には類義語辞書である WordNet[5] を用いた．評価実験としてメトリクス値と専門家の理解性評価の相関を調べたところ 0.64 となり，仮説が正しいことを明らかにした．しかし，中村らのメトリクスでは汎化や集約といった関連線の種類を考慮していないという問題点がある．読み手は関連線の種類から何かしらの情報を得ていると考えられ，考慮しないことで情報の欠損が発生していると推測できる．そこで筆者らは先行研究と同様に WordNet の「上位語-下位語」,「全体語-部分語」のそれぞれの概念を活用し，汎化，集約に対応したメトリクスを提唱した [4]．被験者実験を行い，被験者の理解性評価とメトリクスの評価値を比較したところ，メトリクスの精度向上を確認する事ができた．このことから関連線の種類が読み手の理解性判断に影響を与えていることが明らかとなった．

しかし，一方でいくつかの手法の問題点が明らかとなった．

1 つ目は，有意な評価実験が困難であることである．現在，クラス図のデータセットは存在しておらず統計的考察を行うために十分な数を用意することは困難であり，どちらの先行研究の結果も有意性について疑問が残る形となった．

2 つ目は自然言語処理技術として WordNet のみを用いていることである．単語の類似度計測が可能な自然言語処理技術は他にも多く存在しておりそれらを導入，併用することで精度向上が可能であると考えられる．

3 つ目は定式化による高精度な妥当性定量化が困難なことである．どちらの先行研究も関連要因の洗い出し，加重平均による重み付け，尺度の調整等をすべて考慮し式に落とし込むことによる定式化を行うことで妥当性の定量化を試みている．

我々の提案する手法では，教師あり学習を用いてクラス図内の任意の要素の予測モデルを構築する．予測モデルでは対象のクラス図の構成要素の一つを空欄に置き換え，周辺の情報から空欄部分に当てはまる要素を予測する．予測モデルを用いた妥当性評価方法について図 1 に例を示す．

妥当性評価では，はじめに対象のクラス図の構成要素の一つを空欄に置き換え，次に予測モデルを用いて空欄部分に当てはまる要素候補とそれぞれの確率を予測する．最後に予測結果と元の構成要素を比較し，予測結果の確率に応じて妥当性評価値を算出する．用いた予測モデルが人間の妥当性判断を模倣していることを証明することで妥当性を保証できる．

機械学習モデルを用いることで定式化では困難であった各要因の重み設定並びに尺度の調整が可能となり，妥当性評価の精度向上が期待される．

図 1. クラス名予測モデルを使った妥当性評価の例

2. 提案手法

本研究では研究の初段階として，関連線の種類を予測する予測モデルを作成する．予測モデルは教師あり学習を用いて生成する．図 2 に学習の全体図を示す．説明変数は対象の関連線の両側に位置するクラス名 2 つ，目的変数は関連線の種類と設定する．関係線は 2 つのクラス間の関係性を表現するため，関連線が結ぶ 2 つのクラス名の関係性を複数の自然言語処理技術を用いて特徴量として数値化し，予測モデルの入力として用いることで精度向上を試みる．

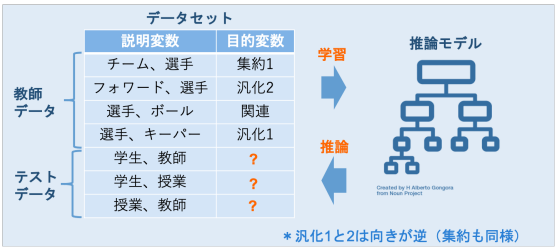

図 2. 教師あり学習の全体図

第 1 章でも述べたとおりこの予測モデルを用いて妥当性評価を行う．関連線予測モデルを用いた妥当性評価の

例を図 3 に示す．妥当性評価ではまず評価対象となる関連線の両側に位置するクラス名から予測モデルを用いて関連線の予測を行い，関連線候補とその確率を算出する．そして元の関連線と予測結果を照らし合わせ，確率を妥当性評価値として受け取る．

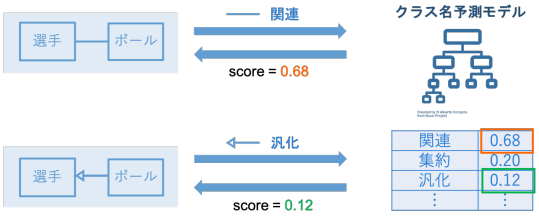

図 3. 関連線予測モデルを用いた妥当性評価の例

また，提案手法のメリットとして 2 つの先行研究が抱えていたその他の問題点も解決可能である．1 つ目は先行研究では限られたデータを用いて評価実験を行っており，データの有意性の部分で問題があった．一方，提案手法では分析対象をクラス図全体から関連線へと置き換えているため，ある程度の数が手作業でも確保可能となる．2 つ目は先行研究では自然言語処理技術として WordNet のみを用いており，他手法を適用，併用することで精度向上の余地があるという問題があった．提案手法では教師あり学習を用いており，特徴量として他の手法を複数組み合わせることができる．よって，特徴量を組み合わせ，複合的に妥当性評価を行うことができるとともに，それらの特徴量に対する予測モデルの学習結果を見ることで，どの特徴量がより予測に役立っているかを判断する事ができる．

3. 予測モデルの作成

この章では予測モデルの作成過程について説明する．

3.1. データセット作成

現在，有力な公開データセットは存在していないため参考書や web ページといった複数のドメイン上に存在するクラス図から，クラス名とその間の関連線を抽出することでデータセットを手作業で作成した．最終的に汎化 1，集約 1，関連をラベルとした 260 データを用意した．ただし，汎化や集約は方向性を持った関連線であり，その向きによって関係性が異なるため，向きに応じて別のラベルを与える必要がある．よって汎化 1 と集約 1 をそのまま反転させ，それぞれ汎化 2 と集約 2 として追加した．つまり，ラベルに含まれる数字がそれぞれの向きを表している．これにより 425 個のデータセットが完成した．データセットの関連線の種類ごとの分布を表 1 に示す．

表 1. 関連線の分布

関連線の種類	ラベル	データ数
関連	k	95
汎化 1	h1	83
汎化 2	h2	83
集約 1	s1	82
集約 2	s2	82

次にデータセットのドメインを分類した際の各データ数を表 2 に示す．

表 2. データセットにおけるドメインの大まかな分類

ドメイン	ラベル	データ数
サンプルサイト	creately	100
web	web	67
過去の実験データ	me	68
参考書	book	156
OSS	swing_game	34

このようにデータセットではドメインを可能な限り分散させることで予測モデルの汎化性能を確保した．また，データセットの言語を英語で統一するために，日本語で記述されたデータは翻訳サービス DeepL[12] を用いて英語に翻訳した．さらに語句は表記ゆれの観点からすべて名詞単数形に変換した．その後，データセット内に存在する単語に対して WordNet の synset 形式に変換するための辞書 CSV ファイルを手作業で作成した．

3.2. 特徴量作成

関係線はクラス間の関係性を表現するものであるため，関連線の両端にある 2 つのクラス名の関係性を複数の自然言語処理技術を用いて特徴量として数値化する必要がある．本研究で作成した特徴量一覧を表 3 に示す．

表 3. 特徴量一覧

ラベル名	説明	使用コーパス	次元数
sim_by_mcg	is-a群の類似度	Microsoft Concept Graph	1
is_a_by_mcg	is-a関係かの評価値	Microsoft Concept Graph	1
lexvec_sim	ベクトルのcos類似度	LexVec(word2vec)	1
lexvec_diff	ベクトルの差分	LexVec(word2vec)	300
wiki2vec_sim	ベクトルのcos類似度	Wiki2Vec(word2vec)	1
wiki2vec_diff	ベクトルの差分	Wiki2Vec(word2vec)	300
wn_hu_ave	先行メトリクス（汎化関係）	WordNet	1
wn_pu_ave	先行メトリクス（集約関係）	WordNet	1
wn_sim_path	pathsimilarity（関連関係）	WordNet	1
wn_sim_wup	WordNetを用いた類似度	WordNet	1
bert_sim	説明文ベクトルのcos類似度	BERT/WordNet	1
bert_diff	説明文ベクトルの差分	BERT/WordNet	768
is_include_word	同じ単語を含むか	なし	1

表 3 においてオレンジ色は先行研究で用いられていた理解性メトリクスを改良したものとなっている．これらの特徴量をデータセットに対して算出した．欠損値が発生する特徴量に対しては最頻値補間を行い対応した．以下の節でそれぞれの特徴量について説明を行う．

3.2.1. Microsoft Concept Graph を用いた特徴量

MCG は Microsoft 社が行っている自然言語処理プロジェクトである [6]．彼らは Probase という自然言語コーパスを所持しており，その活用方法として，自然言語処理を用いて物事の概念化をゴールに研究を行っている．その研究過程での成果物として彼らは単語間の is-a 関係を定量的に評価する事のできるサービスを提供している．具体的には特定の英単語を与えるとその単語と is-a 関係を持つ単語群と各単語の Score が返される．この Score は is-a 関係に強さを表している．MCG は is-a 関係がベースとなっているため，同様に is-a 関係がベースとなっている汎化の分類に有用であることが期待される．まず 1 つ目の特徴量として，sim_by_mcg を定義した．sim_by_mcg では 2 つの単語に対して，それぞれ単語の is-a 単語群における共通単語を調べることで関係性を表現する．アルゴリズムの説明にあたって核となるコードを図 4 に示す．

まず，_get_dict_form_ms_api メソッドを用いて word_a と is_a 関係を持つ上位 30 個とその評価値を辞書形式

```
def word_sim(word_a, word_b):
    di_a = MCG._get_dict_from_ms_api(word_a)
    di_b = MCG._get_dict_from_ms_api(word_b)
    eva = 0
    if di_a == {} or di_b == {}:
        return np.nan
    for key, value in di_a.items():
        if key in di_b.keys():
            eva += 10 * (value + di_b[key])
    return np.average(eva)
```

図 4. sim_by_mcg のコード

(key:単語,value:評価値) で MCG の API にアクセスして取得する．word_b に対しても同様の処理を行う．MCG 内で word_a もしくは word_b が存在していない場合は欠損値として扱うため NaN を返す．どちらも存在した場合 word_a の辞書と word_b の辞書で共通している単語のそれぞれの評価値の合計に 10 をかけたものの合計を返す．したがって共通する単語が多ければ多いほど値は大きくなり，一方で共通単語が存在しなかった場合は 0 を返す．

複合語の場合は複合語 A と複合語 B 間の単語のすべての組み合わせに対してこの値を算出し平均を取る．

同じく MCG をベースとした特徴量として is_a_by_mcg を定義した．この特徴量は単純に MCG 上で 2 つの単語が is-a 関係でつながっているかどうか，つながっていた場合は評価値がどの程度かを確認する．前述の sim_by_mcg との差として，is_a_by_mcg は 0 の値をとる傾向があり，MCG の単純な評価値を直接用いるためノイズが紛れ込みくい利点があると考える．アルゴリズムの説明のため，図 5 に特徴量算出の核となる部分のコードを示す．

まず，_get_dict_form_ms_api メソッドを用いて word_a と is_a 関係を持つ上位 30 個とその評価値を辞書形式 (key:単語,value:評価値) で取り出す．word_a が MCG に存在していない場合は欠損値として扱うため NaN を返す．存在した場合は先程の辞書内に word_b があればその評価値に 100 を掛けたもの，そうでなければ 0 を返す．

複合語の場合は複合語 A と複合語 B 間の単語のすべての組み合わせに対してこの値を算出し平均を取る．

```
def eva_is_a(word_a, word_b):
    di_a = MCG._get_dict_from_ms_api(word_a)
    if di_a == {}:
        return np.nan
    if word_b in di_a.keys():
        return di_a[word_b] * 100
    else:
        return 0
```

図 5. is_a_by_mcg のコード

3.2.2. Word2Vec 系の技術を用いた特徴量

Word2Vec[7] は単語をベクトル化する Word Embeddings 手法の一つであり，その汎用性の高さから多くの派生手法が研究されている．本研究ではその中でも LexVec[8] と Wiki2Vec[9] について取り扱う．LexVec は Word2Vec の後継モデルであり，純粋により高い精度が期待できる．Wiki2Vec は Wikipedia をコーパスとしており，ほかの Word2Vec 系の特徴量とは異なる特徴量として機能することが期待できる．本研究ではタイプの異なる 2 つの Word2Vec 派生の手法を用い，その差を比較することで，Word2Vec の手法が異なることで本タスクに影響があるのかどうか考察を行う．

一般的に Word2Vec を用いて単語間の類似性を見る際は cos 類似度を用いるため，LexVec を用いた cos 類似度による特徴量 lexvec_sim，Wiki2Vec を用いた cos 類似度による特徴量 wiki2vec_sim を作成した．基本的には cos 類似度をそのまま返すが，モデルに登録されていない未知語であった場合は欠損値として NaN を返す．クラス名が複合語の場合は複合語を構成する複数の単語のベクトルの加算平均を取りそのベクトルを用いて最終的にもう一方のクラス名のベクトルとの cos 類似度を求める．

今回のタスクである関連線の分類では向きを考慮する必要があるが前述の cos 類似度では向きは考慮する事ができない．よって cos 類似度に加えてベクトル差分による特徴量 lexvec_dff，wiki2vec_diff を作成した．ベクトル差分では引き算であるため向きを考慮する事ができる．さらに cos 類似度では多次元を 1 次元に圧縮することになるため情報が抜け落ちてしまいやすいが，ベクトル差分では情報量を保ったまま関係性を表現できることに加え，分類に適した形で次元圧縮を併用する事ができる．基本は word_a から word_b を引いたベクトル差分をそのまま返す．モデルの未知語であった場合は欠損値として NaN を返す．複合語の場合は複合語を構成する複数の単語のベクトルの加算平均を取り，そのベクトルを用いて最終的にもう一方のクラス名のベクトルとのベクトル差分を求める．

3.2.3. BERT による特徴量

本来 BERT[10] は文章のベクトル化に特化したアルゴリズムであるため単語のベクトル化に用いることは一般的ではない．しかし，近年の NPL 分野でデファクトスタンダードな BERT と WordNet を組み合わせることで精度向上が期待できる．

WordNet の概念を表す各 Synset には定義文が定義されている．この文章を事前学習済み BERT モデルに投げることで定義文ベクトルを取得できる．定義文は単語をその単語を使わずに説明しているわけであるため単語と定義文ベクトルは意味的に等しいはずである．

この考えのもと word_a，word_b それぞれで定義文ベクトルを生成し，Word2Vec 系と同様に bert_sim は 2 つのベクトル間の cos 類似度，bert_diff は 2 つのベクトル間の差分を特徴量として定義した．複合語の場合は複合語を構成する複数の単語のベクトルの加算平均を取ったものを定義文ベクトルとして用いる．

3.2.4. 理解性メトリクスを用いた特徴量

先行研究で定義された理解性メトリクスを特徴量として用いることで 2 章で述べたとおり，今回新たに作成する特徴量との比較を行う事ができる．理解性メトリクスは木構造上に単語が格納された類義語辞書である WordNet をベースに作られている．WordNet では上位語，下位語という概念が定義されており 2 つの間には is-a 関係が成り立つ．木構造はこの上位語，下位語の関係性で形成されたネットワークである．

理解性メトリクスは関連の評価を行う CWsim メトリクス，汎化の評価を行う HU メトリクス，集約の評価を行う PU メトリクスに分解できる．よってそれぞれを特徴量として定義した．

まず，汎化関係の理解性評価を行う HU メトリクスから特徴量 wn_hu_ave を生成した．アルゴリズムの説明のため，図 6 に特徴量算出の核となる部分のコードを示す．

```
def hu(word_a, word_b):
    wn_a = WordNet.get_wn(word_a)
    wn_b = WordNet.get_wn(word_b)
    if wn_a == wn_b:
        return 1.0
    coms = wn_a.lowest_common_hypernyms(wn_b)
    stack = []
    for com in coms:
        if wn_a == com:
            return 1.0
        stack.append(wn_a.path_similarity(com))
    return max(stack)
```

図 6. wn_hu_ave のコード

まず，get_wn でそれぞれの単語を Synset 形式に変換する．Synset 形式とは WordNet に

おける同音異義語を集めた一つの概念を表す．wn_a.lowest_common_hypernyms(wn_b) はもっとも下位に存在する共通の上位語を返す．複合語の場合は複合語 A と複合語 B 間の単語のすべての組み合わせに対してこの値を算出し平均を取る．

次に，集約関係の理解性評価を行う PU メトリクスから特徴量 wn_pu_ave を生成した．アルゴリズムの説明のため，図 7 に特徴量算出の核となる部分のコードを示す．

```
def pu(word_a, word_b):
    part_wn = WordNet.get_wn(word_a)
    wn_b = WordNet.get_wn(word_b)
    max_pu = 0.0
    if part_wn.part_holonyms() != []:
        part_holo_wns = part_wn.part_holonyms()
        for part_holo_wn in part_holo_wns:
            pu = part_holo_wn.path_similarity(wn_b)
            max_pu = max(max_pu, pu)
    return max_pu
```

図 7. wn_pu_ave のコード

まず，get_wn でそれぞれの単語を Synset 形式に変換する．part_wn.part_holonyms() で word_a の全体語のリストを取得している．この全体語のリストと word_b の path_similarity を算出し，最大値を返している．全体語が１つもない場合は 0 を返す．

最後に関連の理解性評価を行う CWsim メトリクスから wn_sim_path ならびに wn_sim_wup ２つの特徴量を生成した．CWsim は WordNet が提供する単語類似度算出手法の PathSimilarity 法をベースに作成されている．しかし，WordNet では他にもいくつかの単語類似度算出手法が定義されており，当然各手法で表現できる特徴も異なる．よってそれぞれの手法において，各手法間の相関値を算出し，PathSimilarity 法と最も相関が低くなった WuPalmer 法をベースとして wm_sim_wup ２つの特徴量を生成した．

3.2.5 その他の特徴量

is_include_word は単純に２つの語句間で共通している単語があるかどうかを確かめる特徴量である．共通していれば 1 を返し，そうでなければ 0 を返す．他特徴量の多くはクラス名間で共通単語が出てきた場合，値が著しく高くなる傾向があり，それによりモデルが単語間の関係性ではなく共通単語に対して学習を行うリスクがある．よって is_include_word をモデルが重視していれば，上のリスクが高いことがわかり，そうでなければ想定通り単語間の関係性を学習できている事がわかる．

3.3 次元圧縮

現在のデータセットは 1377 次元の特徴量を持つがその中でベクトル系の特徴量が占める割合が表 3 で示したとおり非常に大きい．これによって次の２つの問題が発生するリスクがある．１つは次元の呪いによって学習がうまく行かないリスクがあることである．そしてもう１つは特徴量の評価が困難になることである．XGBoost で Feature Importance を算出した際，１つの特徴量が多くの次元を持つと次元ごとに評価値が算出され，他特徴量との比較が複雑化する．これらの理由から本実験ではデータセット内のベクトル系特徴量に対してのみそれぞれ主成分分析 (PCA)[11] を行うことで次元圧縮を行う．

各ベクトル特徴量ごとに１つ PCA のモデルを用意し次元圧縮を行う（以後これら３つのモデルの集合体をまとめて次元圧縮モデルと呼ぶ）．各 PCA モデルの圧縮次元数についてはデータセットすべてのデータでの累積寄与率を算出しそれぞれ決定した．

3.4 学習

学習推論は層化 90 分割交差検証法をベースとして行った．

予測モデルに用いる数理モデルとして XGBoost を選択した．理由について説明する．一般的に用いられる NN 系の数理モデルではパラメータチューニングに依存する部分が大きく，過学習のリスクが高く，さらに，説明性が低いという問題点があり，本研究には適切でないと判断した．よって次に精度が出やすいとされる Boosting アルゴリズムの中でも，最も古く，多くの知見がある XGBoost を使用する．チューニングの結果，パラメータは木の数を 300，木の最大深さを 17，学習率を 0.25 と設定した．学習，推論の全体図を図 8 に示す．

図について解説する．まず層化 90 交差検証法に従いデータをラベル分布が等しくなるように 90 等分する．その中から１つをテストデータとし，その他をトレーニングデータとする．この際，テストデータの正解ラベルは予め記録しておく．まず次元圧縮モデルに対してトレーニングデータを用いて学習を行う．この際の圧縮次元数は 3.3 節で定めたものを使う．学習した次元圧縮モデルに対してトレーニングデータ並びに，テストデータを

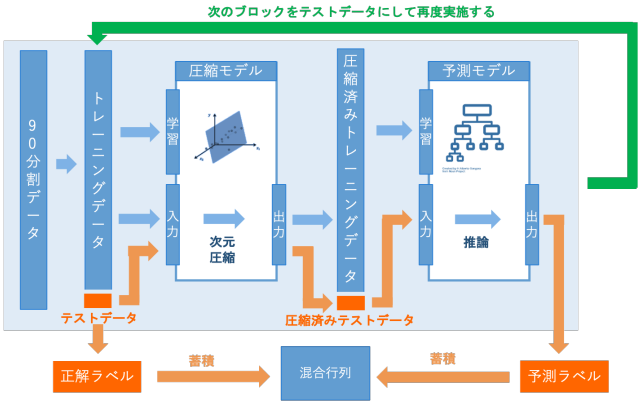

図 8. 学習，推論の全体図

入力し，圧縮済みトレーニングデータ，圧縮済みテストデータを受け取る．次に圧縮済みトレーニングデータを使って予測モデルの学習を行う．その後学習した予測モデルに対して圧縮済みテストデータを入力し，予測ラベルを受け取る．そして予め記録しておいた正解ラベルと予測ラベルを混合行列に追加する．ここまで完了したらテストデータを次のブロックに変更し，同様の工程を行う．これを 90 ブロックすべてで行う．これによって混合行列には正解ラベルと予測ラベルが 425 の全データ分格納される．

結果として得られた混合行列を図 9 に示す．

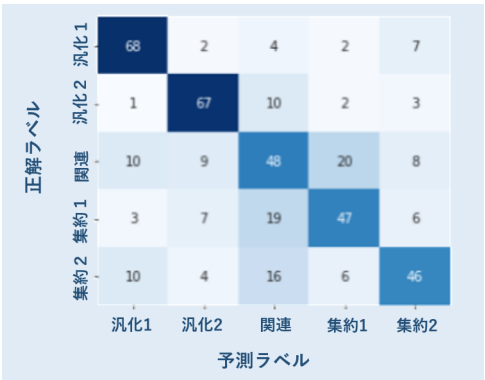

図 9. 予測モデルの混合行列

図のセル内の数字はデータ数を表している．ここから Accuracy を算出すると 0.65 となった．

4. 評価実験

本章では作成した予測モデル評価のために行った評価実験について述べる．

4.1. 実験内容

本校情報工学科 5 年生 1 名，システム創成工学専攻情報システムコース 1 年生 1 名，同コース 2 年生 4 名で被験者実験を行った．彼らはいずれも授業でクラス図について学習を行っている．ただし，前 2 人は半期の授業 1 つ，後ろ 4 人は半期の授業 2 つで学習時間が異なっている．実験は提案手法と同様にデータセットに含まれるクラス名のペアから関連線の種類 (汎化 1，汎化 2，関連，集約 1，集約 2）を予測し回答するという形式で実施した．時間は無制限とし，クラス図についての知識が曖昧になっている可能性や，データセットに英語が含まれるため，個人の英語能力に依存する可能性を考慮しデータセットのドメインを除く辞書や web の利用を許可した．

この評価実験の目的は，予測モデルと可能な限り同等の環境で人間のパフォーマンスを計測し，予測モデルと比較することで，予測モデルの精度を評価することである．人間のスコアが予測モデルと同等程度であれば予測モデルは人間の妥当性判断を模倣できているといえ，提案手法は妥当性評価が可能であると導ける．

ただし以下の点で被験者と予測モデルは異なっている．

- 予測モデルはクラス図についての学習先がデータセットのみであるが，被験者は授業を通して学習を行っている．
- 予測モデルは事前学習から各単語の前提知識を構築するが，人間は，日常生活やこれまでの豊富な知見から前提知識を構築する．
- 予測モデルは推論時，1 つのデータのみから推論を行うが，人間は前後データからの情報を推論に活用できる．(1 つのクラス図から複数のデータを抽出しているため)

4.2. 実験結果

まず実験結果として各被験者の Accuracy を表 4 に示す．

表 4 より被験者全体での平均は 0.58 となった．しかし，Accuracy に注目してみると被験者 1〜3 と 4〜6 で大きく 2 グループに分割することができる．accucary が低かった 1〜3 のグループについては学習時間が少ない本科 5 年生，専攻科 1 年生が含まれることからクラス図の理解度がもう 1 つのグループに比べて低かったことが

表 4. 関連線の分布

被験者番号	学年	Accuracy
1	本科 5 年	0.40
2	専攻科 1 年	0.40
3	専攻科 2 年	0.48
4	専攻科 2 年	0.69
5	専攻科 2 年	0.72
6	専攻科 2 年	0.76

読み取れる．よって 4〜6 のグループをクラス図に対して十分な理解がある習熟グループとして Accuracy の平均を算出すると 0.72 となった．

ここの結果からいえることはある程度クラス図に対して習熟している学生でも 7 割程度の精度が限界であるということである．この原因として，もともとクラス図はプログラマによって一意に定まらないケースが存在する曖昧性の高いタスクであることに加え，本実験では 2 クラスのみという情報量が削減された環境（被験者は前後から推測可能ではある）だったことにより，より一層，答えを定めるのが困難なタスクとなってしまったと考える．

また習熟グループについてそれぞれ混合行列を出力し，それらの平均をとったものを図 10 に示す．

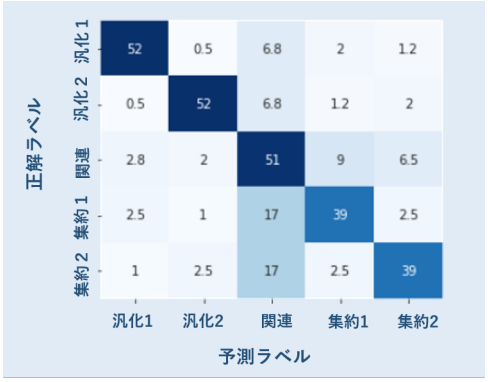

図 10. 習熟グループの結果から算出した混合行列

5. 考察

この章では考察について述べる．

5.1. 予測モデルの有意性についての考察

作成した予測モデルの有意性について考察を行う．我々は以下の 3 つの根拠から予測モデルが人間の妥当性評価を模倣できていると主張する．

- 幅広いドメイン・異なる種類の図からデータセットを作成しており汎用性が高いこと
- 被験者の正答率が 0.7 程度あり，データセットの信頼性が保証されていること
- 被験者の回答結果と予測モデルの予測結果間での相関が高いこと

それぞれについて説明を行う．1 つ目について説明する．作成したデータセットでは予測モデルに十分な汎化性を持たせるため，幅広いドメインから抽出しており，かつ図の種類も分析用の図や教科書の例題，コードからの生成等様々である．例を出すとテトリスや，野球，ショッピングシステムと言ったように全く類似性のないクラス図が対象となっている．さらにこれらは特定の作者によって作成されたものではなく，いずれも異なる作者によって異なる目的を持って作成されたクラス図である．このような分散の大きくかつ少量のデータセットで精度を出すのは非常に困難なことである．その点を考慮すると本研究の予測モデルの精度は十分に高いといえる．さらに，同様の理由から未知のデータに対しても精度を落とさず予測が可能であるといえる．

2 つ目について説明する．上で述べたとおり，データセットの一部は Web サイト等のフリーソースから抽出しており，理解性が低く外れ値となるデータが含まれている可能性がある．外れ値が多く含まれたデータを学習してしまうと予測モデルが外れ値に対して過学習を起こす可能性が高くなる．しかし，評価実験での被験者の正答率が 0.7 程度であることから外れ値データは 0.3 未満であることが推測できる．よって上で述べた可能性は存在せず，正しく学習ができていると推測できる．

3 つ目について説明する．予測モデルと，被験者のそれぞれの混合行列，図 9 と図 10 を比較するとかなり類似している．さらに，習熟グループの 3 人の答え (subject_ans1-3) とモデルの予測ラベル (model_prediction) 間でクラメール連関係数を計算した結果を図 11 に示す．

図 11 を見てみるといずれも 0.5 を超えていることから仮説は棄却され，非独立性が証明されたため本提案手法で作成した各特徴量が人間の判断基準を十分にとらえているといえる．

これらの根拠から，予測モデルが人間の妥当性評価を模倣できていると主張する．

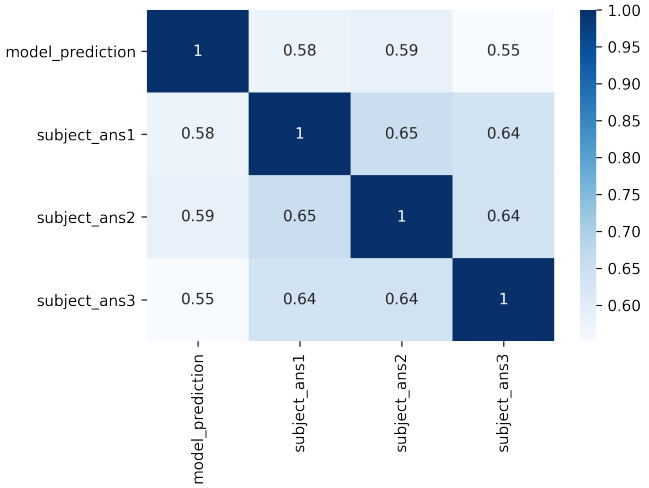

図 11. 各被験者とモデル間のクラメール関連係数

5.2. 各特徴量についての考察

各特徴量について考察を行う．まず特徴量間の相関を示す相関行列を図 12 に例を示す．ただし 1 次元のみの比較であるため diff 系は含まない．

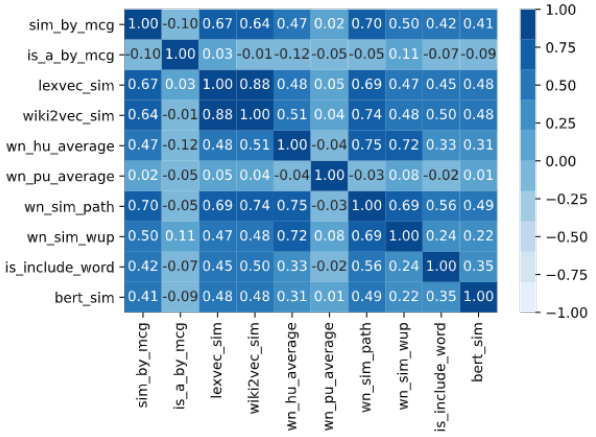

図 12. 相関行列

図 12 を見ると，事前に相関が高くなりすぎる特徴量に関しては除外しているため基本的にはそこまで高いものは存在していないことが確認できる．これは多重共線性の観点から学習にとって好ましいことである．ただし wiki2vec_sim と lexvec_sim は wn_sim_wup と wn_sim_path については相関が比較的高い値となっている．これらのことから，3.2 節で述べたように Word2Vec 系，WordNet の Similarity 系どちらも手法間の差は本タスクにおいては発生しにくいと判断する事ができる．

各特徴量を比較するために全データを用いて XGBoost を学習させ，Feature Importance を算出した．ただし，ベクトル系の特徴量については 1 次元に圧縮を行っている．結果を図 13 に例を示す．

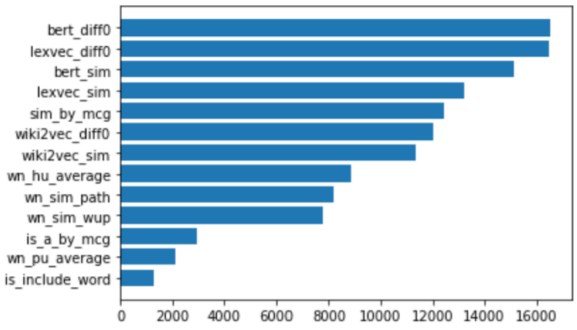

図 13. Feature Importance

全体的に見てみると，BERT や LexVec といった特徴量が高い値を出しており，一方で WordNet 等が低い値となっている．これは概ね一般的な自然言語処理タスクの結果の傾向と類似している．

まず，ベクトル系特徴量のベクトル差分と cos 類似度について比較するとベクトル差分が高いスコアであることから，やはり 3.2 節で述べたとおり，cos 類似度によってデータの特徴が圧縮されてしまうため，純粋にベクトル差分を取る方法が有効であると言うことが明らかとなった．また，今回作成した WordNet と BERT を組み合わせた特徴量である bert_diff 並びに bert_sim の精度が高く，単語という限られた情報量かで有用な手法であることが示された．

また MCG について見てみると sim_by_mcg がベクトル系についで高い値となった．MCG は未だマイナーな手法であるがこのように自然言語処理タスクの特徴量として用いる事ができるという可能性を明らかにする事ができた．ただし is_a_by_mcg については低い値となり，この原因としては殆どの単語の組み合わせで特徴量が 0 となってしまったからであると推測する．

Word2Vec 系の特徴量について見てみると，LexVec がわずかに良い値であるという結果となった Wiki2Vec について，データセット内で 12 データ分未登録単語があったため欠損値が発生していた．よってこの差が精度の差として現れたといえる．

また先行研究で用いられた WordNet 系の特徴量につ

いて見てみると，どれも下位の方に位置しており，従来研究よりも高い精度での予測が可能であると分かる．

6. まとめ・今後の展望

本研究ではクラス図の理解性の定量化を最終的な目的とし，先行研究と同様に妥当性の定量化を試みた．本研究では予測モデルを用いた妥当性の定量化手法を提案した．手法提案の初段階として，本研究では関連線の種類の予測を行う予測モデルを作成した．被験者実験を実施し，予測モデルの予測結果と，被験者の判断結果を比較したところ，予測モデルは人間の妥当性判断を模倣可能であることが明らかとなった．このことから提案手法を用いることで関連線の妥当性評価が可能であることが示された．今後の展望としては以下の通りである．

- クラス名等のクラス図の他要素に対して提案手法を適用する．
- 予測モデルを関連線の自動生成の目的で使用する．
- 本研究では WordNet の synset との紐づけを手作業で実施しているが，仕様書の文章等から紐づけを自動化する．

参考文献

[1] Jorgen Boegh, A New Standard for Quality Requirements, IEEE Software, pp.20-27, January 2008.

[2] 中村悠人, 鷲崎弘宜, 深澤良彰，単語の類似度を用いた UML クラス図の理解性の測定と評価，ソフトウェアエンジニアリング最前線， 情報処理学会 SE シンポジウム 2009，pp.165-172，September 2009.

[3] 高橋直久, 丸山勝久:ソフトウェア工学, 森北出版株式会社, 2010

[4] 井原輝人, 内田眞司 , 単語類似度を用いた UML クラス図理解性メトリクスの改良, 信学技報, vol. 119, no. 246, SS2019-24, pp. 65-70, October 2019

[5] WordNet - Princeton University Cognitive Science Laborator，http://wordnet.princeton.edu/，2019/9/23 閲覧

[6] Zhongyuan Wang and Haixun Wang, Understanding Short Texts, in the Association for Computational Linguistics (ACL) (Tutorial), August 2016.

[7] Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, Efficient Estimation of Word Representations in Vector Space, arXiv:1301.3781v3 [cs.CL] 7, September 2013

[8] A. Salle, A. Villavicencio and M. Idiart, Matrix factorization using window sampling and negative sampling for improved word representations, In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 419–424, 2016

[9] Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji,Joint Learning of the Embedding of Words and Entities for Named Entity Disambiguation, Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning,pp 250-259, 2016

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, arXiv preprint arXiv:1810.04805v2, May 2019.

[11] Sebastian Raschka，Vahid Mirjalili，［第 2 版］Python 機械学習プログラミング 達人データサイエンティストによる理論と実践，インプレスブックス，2018

[12] DeepL, https://www.deepl.com/ja/translator, 2020/12/2 閲覧