ユースケースから COSMIC 法による機能規模を概算する深層学習手法の性能評価

天嵜聡介
岡山県立大学
amasaki@cse.oka-pu.jp

廣石将士
岡山県立大学
sc220043@cse.oka-pu.ac.jp

要旨

ソフトウェアの機能規模は工数の見積もりや生産性の評価など様々な場面で有用な情報である．要求仕様に対応する論理モデルに基づいた *IFPUG* 法や *COSMIC* 法などが機能規模を測定する方法として広く用いられている．一方でこれらの機能規模測定手法は詳細な要求仕様やマニュアルに従った測定ができる人材の確保などの制約がある．特に開発初期において制約を軽減するために過去の実績データから機能規模を概算する手法が提案されている．近年、深層学習を用いてユースケースから *COSMIC* 法による機能規模を概算する手法（*DEEP-COSMIC-UC*）が提案された．しかしながら性能評価では従来型の機械学習との比較評価が不十分であり深層学習を用いる優位性が明らかでない．本研究では、*DEEP-COSMIC-UC* と従来型の機械学習による機能規模の概算精度を比較評価する．先行研究で公開されたユースケースと *COSMIC* 法で計測された機能規模のデータセットを対象としてランダムフォレスト及びサポートベクター回帰と *DEEP-COSMIC-UC* の間で性能を比較した．実証的な評価実験により *DEEP-COSMIC-UC* に予測精度上の優位性がないことを示した．

1. はじめに

ソフトウェアの機能規模は工数の見積もりや生産性の評価など様々な場面で有用な情報である．機能規模を測定するために要求仕様を情報源として論理モデルを構成するアプローチがある．Albrecht が提案したファンクションポイント法 [1] を端緒として様々な手法がこのアプローチに基づいて提案されている．現在広く利用されているものとして International Function Points User Group (IFPUG) が管理している IFPUG 法 [6]、Common Software Measurement International Consortium（COSMIC）が管理している COSMIC 法 [3] などがある．

IFPUG 法や COSMIC 法などの機能規模測定手法は詳細な要求仕様やマニュアルに従った測定ができる人材の確保などの制約がある．開発初期においては要求は詳細化されていないことも多く、また、近年のアジャイル開発などでは開発作業に応じて詳細化を行うことも多い．さらに、人手による機能規模の測定は時間や工数を要する作業である．このような機能規模測定に関する課題に対応するために機能規模を概算で求める手法（Functional Size Approximation）が提案されている．

機能規模の概算では一部の情報を測定せずに機能規模を見積もることが多い．例えば、NESMA 概算法はデータのタイプごとの複雑度を一定の値であるとみなして測定を省略し機能規模を見積もる．このような方法は簡便である一方でデータのタイプを同定するための人手は必要である．

近年人手を介さず過去の実績データを用いてユースケースから機能規模を直接見積もる手法が研究されている．Ochodek らは深層学習を用いた手法 DEEP-COMIC-UC [9] を提案している．DEEP-COSMIC-UC は評価実験で既存手法より高い性能を示した．一方で深層学習でない従来型の機械学習を用いた場合との性能比較が不十分である．過去の実績データが少ない場合はより簡素な手法の方が良い性能を示す可能性がある．また GPU などの追加機材が必ずしも必要でない点も従来型の機械学習手法の利点である．

本論文では自然言語処理で一般的な特徴量である TF-

IDF と従来型の機械学習を用いたモデルの性能を DEEP-COSMIC-UC の性能と比較して深層学習を用いることの優位性について検証した．実際の開発プロジェクトで収集されたデータセットを用いた評価実験により DEEP-COMIC-UC には従来型の機械学習手法に対する優位性はないことを示した．

本論文の以降の構成は以下のとおりである．第 2 節では過去の実績データに基づいた機能規模概算手法について述べる．第 3 節では本論文で行う評価実験の方法について述べる．第 4 節では実験結果を示し考察を述べる．最後に第 5 節で結論を述べる．

2. 関連研究

ユースケースから機能規模を概算する手法については Ochodek の研究 [8] で調査されている．調査では専門家による判断が見積もりに必要な手法と人が介さず自動化が可能な手法があることが示されている．また入力データとしてユースケース名とユースケースシナリオの 2 種類があることが示されている．本研究の評価対象である DEEP-COSMIC-UC はユースケース名のみを入力とする自動化可能な手法に分類される．ユースケース名は「製品を買う（Buy a product）」のように簡素に記述されることが多い．以下では主に DEEP-COSMIC-UC と同種の手法について述べる．

COSMIC 法を管理している団体である COSMIC が提供している機能規模概算手法の一つに AUC (average use-case approximation) がある [11]．AUC は過去に類似システムの開発で計測された機能要求ごとの COSMIC ファンクションポイント数を平均して対象システムの機能要求の数と積算して機能規模を概算する．機能要求の記述を用いない簡便な手法でありユースケース毎に機能規模が計測されたデータを入力とした概算も可能である．

Hussain ら [5] が提案した手法（HKO）は要求文書に自然言語処理を適用して得られる品詞の種類毎の出現頻度を入力として機械学習手法 C4.5 を用いて機能規模を概算する．ユースケースに対する適用可能性が Ochodek により検討された [7]．

Ochodeck は AUCG（Average Use-Case Goal-aware Approximation）および BN-UCGAIN（Bayesian Network–based Use-Case-Goal-aware ApproxImatioN）を提案した [8]．これらの手法は事前にユースケース名の記述を分析してユースケースを動詞に基づいた階層的なカテゴリに分類する．例えば「更新する（Update）」というカテゴリの下位には「状態を変更する（Change State）」というカテゴリがある．AUCG は 過去の類似したシステム開発で記録したユースケースをカテゴリに分類してカテゴリ毎に記録されている COSMIC ファンクションポイント数の平均値を求める．対象システムのユースケースは下位のカテゴリ優先で分類して AUC と同じ手順で機能規模を概算する．BN-UCGAIN はベイジアンネットワークを利用して機能規模を概算する．ベイジアンネットワークは上述のカテゴリを入力としてユースケースシナリオの種別を表すノードを経由し COSMIC ファンクションポイントを出力する．各ノードに対応する条件付き確率分布表は類似システムのデータから求める．これらの手法は AUC および HKO に比べて予測誤差が減少した．

DEEP-COSMCI-UC [9] は深層学習を用いた機能規模概算手法である．DEEP-COSMCI-UC にユースケース名が入力されると単語ごとに単語ベクトルに変換される．変換には事前学習モデルを用いている．これらの単語ベクトルは畳み込みニューラルネットに入力され最終的に機能規模の概算を出力する．DEEP-COSMCI-UC は HKO・AUCG・BN-UCGAIN より高い性能を示した．

DEEP-COSMCI-UC と比較された HKO は従来型の機械学習を用いている一方で入力は品詞に限られている．DEEP-COSMCI-UC と同様にユースケース中の個々の単語に基づいた特徴量を用いた場合における従来型の機械学習による予測性能を評価する必要がある．

3. 実験方法

3.1. 実験手順

DEEP-COSMIC-UC を提案した論文 [9] では十重交差検証 (10-fold cross-validation）を 10 回繰り返した結果に基づいて性能を評価している．十重交差検証はまずデータセット D をランダムに 10 個のサブセットに分割する．次に 1 個のサブセットを除いた 9 個のサブセットを結合して機械学習モデルの訓練に用いる．最後に取り除いたサブセットを構築されたモデルの性能評価に用いる．この手順を全てのサブセットが 1 回ずつ性能評価に用いられるまで繰り返し 10 個の性能評価指標の値を得る．ランダムな分割は偶然の偏りが発生するため 10 回繰り返している．本研究でも十重交差検証を 10 回 繰

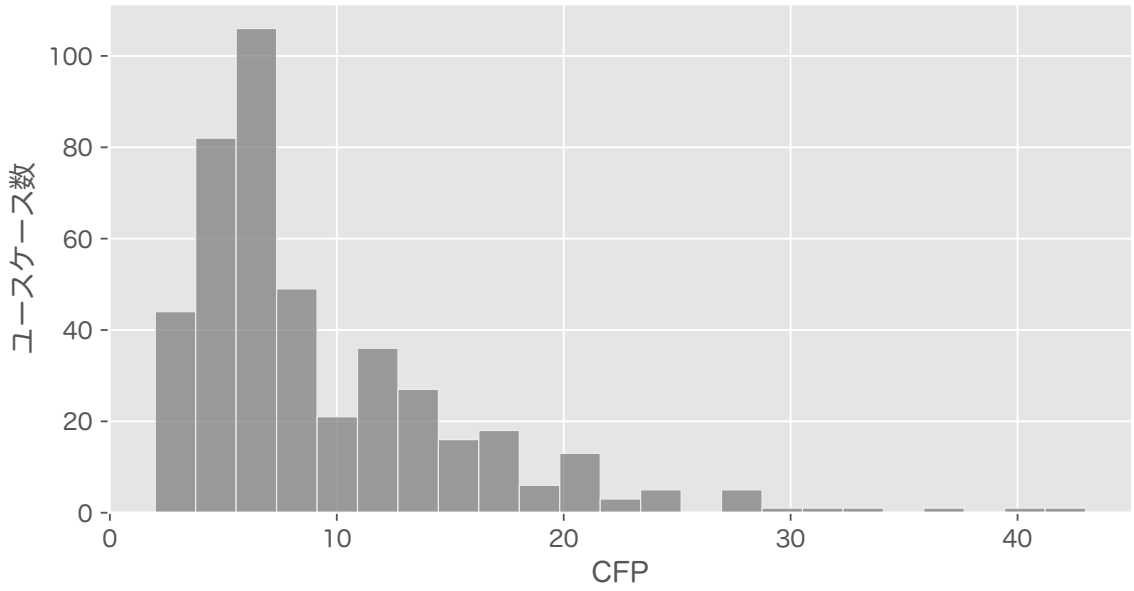

図 1. COSMIC ファンクションポイントの分布

り返した結果で性能を評価する．

本研究では実験手順を文献 [9] とできるだけ一致させるために論文中で提供されたレプリケーションキット [1] を使用した．レプリケーションキットで使用されているプログラミング言語は Python と R である．DEEP-COSMIC-UC の実装では Tensorflow が用いられている．レプリケーションキットには実験結果の再現性を確保するために十重交差検証 10 回分の分割データをファイルから読み込む形式となっている．しかしながらこの分割データには誤って同じ分割データが繰り返し 10 回記録されており評価実験には適さない．そのため本研究では新たに互いに異なる 10 回分の分割データを用意して評価実験を実施した．

3.2. 性能評価指標

文献 [9] と同様に見積もり精度の評価指標として Standardized accuracy (SA) [10] を用いた．SA は絶対誤差の平均値に基づいて以下のように定義されている．

$$\mathrm{SA} = 1 - \frac{\mathrm{MAE}_{P_i}}{\mathrm{MAE}_{P_0}}$$

ここで MAE_{P_i} は評価対象となる予測システム P_i による概算の絶対誤差の平均値を表す．また MAE_{P_0} はランダムな予測を行うシステム P_0 による概算の絶対誤差の平均値を表す．SA は評価対象となるシステムがランダムな予測より良い予測を行なっている場合に 0 から 1 の値を示す．機能規模の見積もり精度が高いほど SA が 1 に近くなる．

3.3. データセット

本研究では文献 [9] と同じデータセットを評価実験に用いた．先述のレプリケーションキットにデータセットは同梱されている．データセットは 27 のソフトウェア開発プロジェクトから収集された．これらのプロジェクトは 6 つのソフトウェア開発組織と 1 つの大学内開発組織で実施された．27 のプロジェクトのうち学生が主体の開発組織は 1 つのみである．データセットで提供されるユースケースの総数は 437 個である．文献 [9] と同様にプロジェクトの区別なく全てのデータを十重交差検証に用いた．

COSMIC ファンクションポイントの値の分布を図 1 に示す．平均値は 9 で標準偏差は 6 である．右側に裾が長い分布となっておりユースケース間で機能規模の粒度がばらついていることが観察できる．

[1] https://github.com/mochodek/deep-cosmic-use-cases

3.4. 特徴量

先行研究で提案された HKO では品詞に着目した特徴量が使用されていた．データセットに同じ単語が繰り返し現れるユースケース名があったため，本研究では単語に着目した特徴量として出現単語数を考慮できる TF-IDF を使用した．

TF-IDF は文書コーパス内の文書を対象とした特徴量である．まず文書に出現する単語毎に出現頻度（Term Frequency）を求める．次にコーパス内でその単語を含む文書数の逆数（Inverse of Document Frequency）を求める．これらの値の積がその単語の TF-IDF となる．したがってコーパス内の文書に含まれる単語の数が特徴量の数となる．

本研究では Python のライブラリである scikit-learn に含まれる `TfidfVectorizer` を用いて TF-IDF を計算した．また TF-IDF を求める前に小文字化、ストップワード・空白文字・記号の除去、見出し語化を行っている．また，TF-IDF はファンクションポイントを参照しない特徴量であるため事前にデータセット全体のユースケース名から計算して実験を行った..

3.5. 従来型の機械学習手法

本研究ではランダムフォレストとサポートベクター回帰の二つを従来型の機械学習手法として採用した．ソフトウェア工学分野では様々な機械学習手法が予測に用いられているが上記の手法は多くの研究で良い性能を示しているためである．

ランダムフォレスト [2] は Breiman が提案した手法である．ランダムフォレストではランダムにデータの一部を選択して決定木を作成する手順を繰り返し複数の決定木を用意する．予測ではそれぞれの決定木の予測の平均値を出力する．

サポートベクター回帰 [4] は Drucker らが提案した手法である．分類モデルであるサポートベクターマシンを回帰に応用した手法でありカーネル関数によって非線形回帰モデルを構築できる．また一定以下の誤差を許容して正則化を行うなどの特徴がある．

ランダムフォレストおよびサポートベクター回帰はハイパーパラメータを調整することで予測精度を向上させることができる．複数のハイパーパラメータを調整するためにグリッドサーチを用いた．グリッドサーチはハイパーパラメータの値の候補の全ての組み合わせの中から最も見積もり精度が高い組み合わせを出力する．本研究では見積もり精度の計算に五重交差検証（5-fold cross-validation）を用いた．ランダムフォレストおよびサポートベクター回帰のハイパーパラメータの候補値を表 1 と表 2 にそれぞれ示す．いずれの候補値もこれらの機械学習手法を用いた既存研究を参考に決定した．

表 1. ランダムフォレストのハイパーパラメータの候補値

ハイパーパラメータ	候補値
`n_estimators`	100, 300, 500, 1000, 1500
`max_depth`	50, 100, 300, 500, None
`max_features`	sqrt, log2

表 2. サポートベクター回帰のハイパーパラメータの候補値

ハイパーパラメータ	候補値
`kernel`	linear, rbf
`gamma`（`kernel` が rbf の場合）	0.001, 0.01, 0.1, 1
`C`	0.1, 1, 10, 100, 1000
`epsilon`	0.01, 0.1, 1, 10

表 3. 性能評価の結果 (平均値)

モデル	絶対誤差の平均値	SA
AUC	4.74	25.56
AUCG	3.92	38.34
BN-UCGAIN	3.80	40.31
DEEP-COSMIC-UC	3.48	45.26
サポートベクター回帰	3.60	43.41
ランダムフォレスト	3.52	44.62

4. 結果と考察

実験結果を表 3 に示す．いずれの数値も 100 個の性能評価指標の平均値である．第 3.1 項で述べたようにレプリケーションキットに含まれる交差検証の分割データ

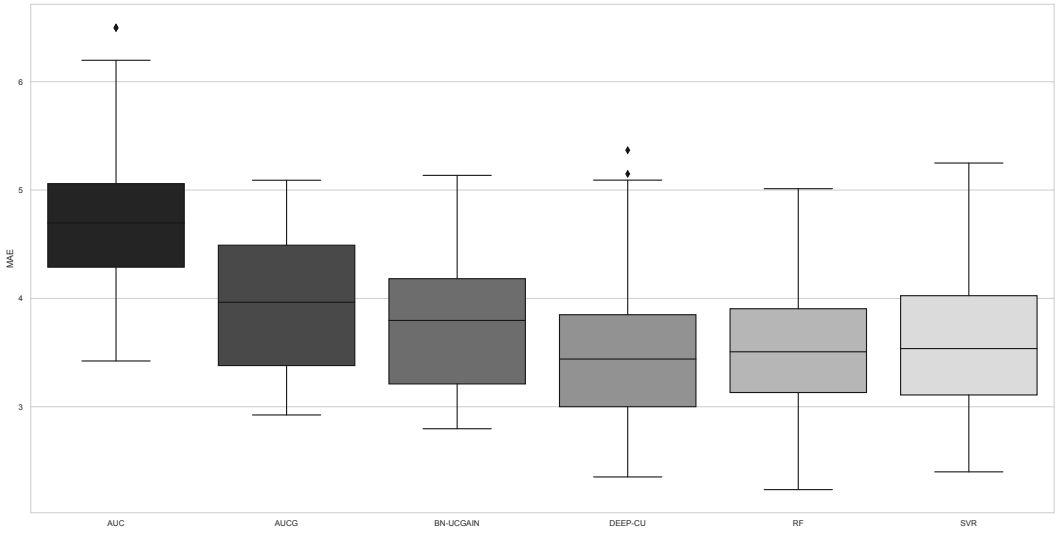

図 2. 絶対誤差の平均値の箱ひげ図

には誤りがある．そこで新しい分割データを用いた場合における DEEP-COSMIC-UC の優位性を確認するために第 2 節で述べた 3 種類の機能規模概算手法の結果を含めてある．また DEEP-COSMIC-UC では単語埋め込みの事前学習モデルとして文献 [9] で最も高い性能を示した Wikipedia+Gigaworld (50d) を使用している．SA の計算にはランダムサンプリングで得られた数値を利用しているため計算の根拠となる絶対誤差の平均値も併記した．

SA および絶対誤差の平均値から DEEP-COSMIC-UC は先行研究で提案された機能規模概算手法より概算の精度が高い．また文献 [9] に記載された数値と大きな違いはないため本研究で得られた結果は妥当であると考えられる．

ランダムフォレストおよびサポートベクター回帰による概算の精度は DEEP-COSMIC-UC の精度より低い．一方で先行研究の手法の中で最も精度が高い BN-UCGAIN よりもランダムフォレストおよびサポートベクター回帰の精度は高い．このため DEEP-COSMIC-UC と従来型の機械学習手法との間の性能の差は十分に小さいと考えられる．

図 2 に絶対誤差の平均値のばらつきを箱ひげ図で示す．いずれの手法においても交差検証による分割の違いにより絶対誤差の平均値に 0.6 から 1 程度のばらつきがあることを箱の大きさが示している．DEEP-COSMIC-UC と従来型の機械学習法の箱ひげ図は近い位置にある．これらの手法の性能差を確認するために有意水準 5%でウィルコクソンの符号付順位検定を行った．ランダムフォレストとの比較では $p = 0.308$ であり統計的な有意差は確認できなかった．サポートベクター回帰との比較では $p = 2.368 \times 10^{-6}$ であり統計的に優位な差が確認できた．しかしながら効果量の指標である Cliff の Delta は -0.1012 であり「ごくわずか（negligible）」とみなせる．このため概算の精度において深層学習手法と従来型の機械学習手法の間に実質的に差があるとは言えなかった．

5. まとめ

本研究ではユースケースから機能規模の概算を自動で行う深層学習モデルの優位性について従来型の機械学習手法と比較しながら評価した．特徴量に TF-IDF を用いたランダムフォレストとサポートベクター回帰に基づいたモデルの性能はいずれも先行研究で提案された深層学習モデル DEEP-COSMIC-UC の性能と実質的な差がないことを実証的に示した．

本研究ではユースケース名に使用される個々の単語の

みを特徴量としていたが HKO で用いられている品詞情報を加えることで深層学習モデルより高い精度を示す可能性がある．またユースケースシナリオのテキストを加えた TF-IDF を特徴量として使用することで精度向上に寄与する可能性が考えられる．

一方，今回は先行研究にならい全てのプロジェクトのデータを区別なく学習・見積もりを行ったが，必ずしもプロジェクト間で共通の語彙が用いられているとは言えない．学習データに出現しない未知の単語を含む場合の性能評価は今後の課題である．

参考文献

[1] A. J. Albrecht. Measuring application development productivity. *Proc. of Joint SHARE/GUIDE/IBM Application Development Symposium*, pp. 83–92, 1979.

[2] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001.

[3] COSMIC. *The COSMIC Functional Size Measurement Method version 5.0*, 2021. `https://cosmic-sizing.org/measurement-manual/`.

[4] H. Drucker, C. J. C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In M. Mozer, M. Jordan, and T. Petsche eds., *Advances in Neural Information Processing Systems*, Vol. 9. MIT Press, 1996.

[5] I. Hussain, L. Kosseim, and O. Ormandjieva. Approximation of cosmic functional size to support early effort estimation in agile. *Data & Knowledge Engineering*, 85:2–14, 2013.

[6] International Function Point Users Group (IFPUG). *Function Point Counting Practices Manual, Release 4.3.1.*, 2010.

[7] M. Ochodek. Approximation of cosmic functional size of scenario-based requirements in agile based on syntactic linguistic features—a replication study. In *2016 Joint Conference of the International Workshop on Software Measurement and the International Conference on Software Process and Product Measurement (IWSM-MENSURA)*, pp. 201–211, 2016.

[8] M. Ochodek. Functional size approximation based on use-case names. *Information and Software Technology*, 80:73–88, 2016.

[9] M. Ochodek, S. Kopczyńska, and M. Staron. Deep learning model for end-to-end approximation of cosmic functional size based on use-case names. *Information and Software Technology*, 123:106310, 2020.

[10] M. J. Shepperd and S. MacDonell. Evaluating prediction systems in software project estimation. *Information and Software Technology*, 54(8):820–827, 2012.

[11] F. Vogelezang, C. Symons, A. Lesterhuis, R. Meli, and M. Daneva. Approximate cosmic functional size – guideline for approximate cosmic functional size measurement. In *2013 Joint Conference of the 23rd International Workshop on Software Measurement and the 8th International Conference on Software Process and Product Measurement*, pp. 27–32, 2013.