Vol.2022-SE-211 No.16
2022/7/29

連合学習によるプライバシー保護を考慮したプロジェクト間バグ予測

山本 大貴[1,a)]　Gopi Krishnan Rajbahadur[2]　近藤 将成[1,b)]　亀井 靖高[1,c)]　鵜林 尚靖[1,d)]

概要：バグ予測モデルはソフトウェアプロジェクトに含まれるバグの発生率を予測できるため，デバッグ作業の支援の手法として活用されている．近年，バグ予測モデルを作成するためのデータが無い場合に，他人の持つデータを利用して学習したバグ予測モデルを用いてプロジェクトのバグを予測する Cross Project Defect Prediction（CPDP）への関心が高まっている．CPDP では他人のデータを利用することから，データのプライバシー保護が重要な課題の 1 つである．しかし，多くの CPDP の研究ではモデルの学習のためにプロジェクト間でのデータの共有が必要であり，プロジェクトの機密性を保護しながらプロジェクト間のバグを予測することを考慮していない．そこで本研究では，データ共有が不要な分散型の機械学習アプローチである連合学習（Federated Learning）を用いた CPDP モデルを提案し，CPDP に連合学習のモデルを利用した際の性能や特徴を明らかにする．連合学習の CPDP モデルと既存の CPDP モデルとの性能比較を AUC を用いて行った結果として，連合学習のモデルは実験に用いたデータセットの 68%で予測性能の順位グループの 2 位以上に割り当てられ，既存のモデルと比較しても一定の予測性能があることを明らかにした．また，本研究の実験環境における連合学習のモデルの予測性能に及ぼす影響が大きい特徴量を調査し，本質的複雑度やコード行数のようなコードに関する特徴量は性能に影響を及ぼすことを明らかにした．

キーワード：バグ予測，クロスプロジェクト，プライバシー保護，連合学習

1. はじめに

ソフトウェア開発において，プログラミングの開発工程にかかるコストのうち，デバッグ作業が約 50%を占めると言われている [1]．そのため，デバッグ作業を支援する技術として，ソフトウェアプロジェクトに含まれるバグの発生率を予測できるバグ予測モデルが活用されている．バグ予測モデルにより，バグを含む確率の高いソースコードの修正やテストにより多くの時間を割くことができるため，ソフトウェアの信頼性向上やデバッグの負荷軽減につながる．

近年，バグ予測の中でも，プロジェクト間バグ予測（Cross Project Defect Prediction，CPDP）[2] に関する研究が行われている．CPDP はバグ予測モデルを構築するためのデータが不十分な場合やデータ自体が無い場合に，他人の所有するプロジェクトのバグデータの情報を使うことでバグ予測をする手法である．バグ予測は企業での適用も報告されており [3]，CPDP の活用法の 1 つとして，企業のデータや共同プロジェクトで CPDP を利用することにより幅広いプロジェクトのバグ予測への利用が見込める．しかし，プロジェクトのバグ予測を行うためのデータを公開することは，データ所有者のプライバシーの問題にも関係することから，CPDP の重要な課題の 1 つとなっている [4]．そのため，プライバシー保護の課題に着目した CPDP の研究が行われているが [4–6]，このような研究では複数の参加者の間でのデータ共有が必要であり，データ自体の共有は避けられない．

そこで本研究では，プライバシー保護を目的とした機械学習のアプローチである連合学習（Federated Learning）[7] を用いた CPDP モデルを提案する．連合学習とは分散型の機械学習アプローチの 1 つである．そのアプローチとして，まず未学習のモデルを参加者に提供した後，参加者がそれぞれのもとでモデルを学習し，学習したモデルのパラメータを提供元のモデルと共有する工程を繰り返すことで元のモデルを学習させる．その後，学習させた元のモデルを用いることでバグ予測が可能となる．そのため，このよ

[1] 九州大学
Kyushu University
[2] Centre for Software Excellence
[a)] h.yamamoto@posl.ait.kyushu-u.ac.jp
[b)] kondo@ait.kyushu-u.ac.jp
[c)] kamei@ait.kyushu-u.ac.jp
[d)] ubayashi@ait.kyushu-u.ac.jp

うなアプローチを CPDP に適用することで，データの共有は行わないままバグ予測を行うことが可能となる．

本研究では，連合学習を用いた CPDP の初期実験として，教師あり学習の Logistic Regression に連合学習を組み込んだ，Federated Logistic Regression（FLR）モデルと最先端の CPDP 手法を比較し，連合学習を用いた CPDP の実現可能性を調査した．また，連合学習アプローチで利用するパラメータについて調査し，FLR モデルの予測性能に影響を与える要素について明らかにした．最後に，本研究の環境における連合学習のモデルについて，予測性能に影響を与える特徴量を明らかにした．

以降，第 2 節では本研究に関連する研究，第 3 節では実験設計について説明する．第 4 節では調査の結果と考察を行う．その後，第 5 節では妥当性に対する脅威について，第 6 節では結論と今後の課題について述べる．

2. 関連研究

プライバシー保護を目指した CPDP: プロジェクトのバグ予測に用いるデータはプライバシー保護の問題から，ほとんどの企業は機密情報であるデータの共有に消極的であると言われている [8]．そのため，ソフトウェアのバグ予測や，作業工数の推定などにおいてプライバシーの問題が研究されている．中でも，プライバシー保護を考慮したプロジェクト間でのバグ予測のために，Peter ら [4] はプライバシー保護されたデータ共有法に焦点を当てた MORPH のアプローチを提案した．MORPH はあるデータのメトリクスの値を変異させる手法であり，変異させるデータのラベルと他のラベルのデータとの境界を変更しないように値を変異させる方法を検討した．

Peter ら [5] は更に CLIFF と呼ばれるアプローチも提案している．CLIFF はデータを削減するためのアプローチである．特徴量ごとにデータを一部分割し，他のラベルよりもそのラベルで最も頻繁に出現している一部分のデータを影響力が強いとみなすことで，影響力の低いデータを削減した．MORPH に CLIFF を合わせた CRIFF+MORPH により，データセットのプライバシーが保たれ，バグ予測への有用性が損なわれないことを示した．

その後, Peter ら [6] は CLIFF+MORPH を拡張することで，LACE2 と呼ばれるデータ共有法を提案した．LACE2 では，データ所有者がデータを追加していく共有キャッシュを導入している．所有者の持つ新しいデータは，リーダーフォロワーアルゴリズム [9] に基づいたデータ同士の距離に応じて，そのデータがキャッシュ内のデータで表せないような場合のみにキャッシュに追加される．この手法では，所有者のデータ全てがキャッシュに追加されることはなく，共有されるデータが少なくなるため，プライバシーの向上を見込むことができる．しかし，これらの研究では複数の参加者の間でのデータ共有が少なからず必要であり，解決すべき課題であると考えられる．

表 1 データセットの概要

グループ	プロジェクト数	全バージョン数	特徴量数	タイプ	言語
AEEEM	5	5	31	OSS	Java
METRICSREPO	12	46	24	OSS	Java
RELINK	3	3	26	OSS	Java
SOFTLAB	5	5	29	CSS	C
計	17	51			

教師なし学習を利用したバグ予測: 最近の研究で Yuming ら [10] は，プロジェクト内のモジュールサイズをメトリクスとしてバグ予測を行う教師なし学習 Manualdown を利用したバグ予測の研究を行った．結果として，利用したデータのうち約 88%以上のデータにおいて，Manualdown のバグ予測モデルは既存の複数の CPDP モデルに比べ同等またはそれ以上の予測性能を持つことが示された．Yuming らは結果から，新たなプライバシー保護を目指した CPDP 研究の比較のベースラインとして Manualdown のモデルを利用を推奨している．

Research Question: 本研究では，データ共有の必要がない連合学習を CPDP に適用した際の有効性と特徴を確かめるため，CPDP に FLR モデルを適用した際の性能に関する 4 つの Research Question（RQ）について調査する．はじめに，FLR モデルの性能が既存の CPDP モデルの性能に比べてどれほどの違いがあるかについて調査する（RQ1）．また，CPDP における FLR モデルのバグ予測精度に影響を与える要素に関する調査として，連合学習のアプローチにおいて利用するパラメータによる影響（RQ2）や参加者数（RQ3）による性能への影響について調査する．最後に，本実験環境で利用した FLR モデルについて，どのような特徴量が FLR モデルのバグ予測性能に影響を与えるかについて調査する（RQ4）．

3. 実験設計

3.1 データセット

本研究では，Yuming ら [10] の CPDP 研究で使用された，プロジェクトごとの特徴量とバグの有無に関するデータセットを利用した．データセットの概要を表 1 に示す．データセットはプロジェクトの言語や用途によってデータのグループ分けがなされている．表 1 におけるプロジェクト数の列はグループごとのプロジェクト数を表している．プロジェクトの中にはいくらかのバージョンを持つものも含まれており，全バージョン数の列はプロジェクトの全てのバージョンの合計数を表す．特徴量数は，グループ内に含まれる全プロジェクトで共通の特徴量の数を表す．タイプの列の OSS は Open Source Software を，CSS は Closed Source Software を表す．本研究では，データ数の関係から 4 つのグループのデータセットからなるプロジェクト合計 25 個を利用した．

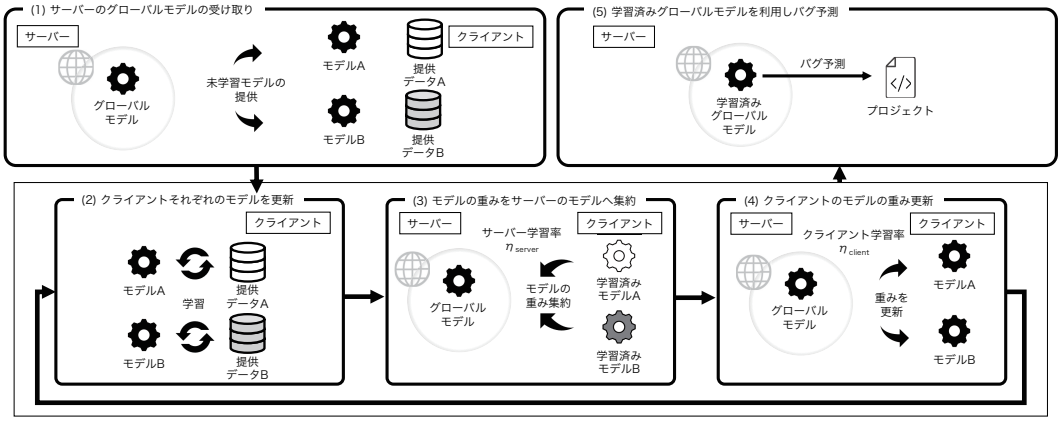


図 1 クライアントが 2 人の際の連合学習による CPDP の流れ

3.2 連合学習モデル

本節では，本研究で扱う連合学習モデルについて説明する．連合学習はプライバシー保護を目的の 1 つとした，分散型の教師ありの機械学習アプローチである．連合学習は特徴としてデータを集約せず分散させた状態でモデルを学習する．そのアプローチは，ある 1 つのグローバルなモデルを提供するサーバーと，グローバルモデルを受け取り，自分の手元のデータでモデルを学習するクライアントにより連合学習の学習工程を進める．本研究では，この連合学習を CPDP に適用する．連合学習を CPDP に適用した際のプロジェクトのバグ予測の流れは以下のように行う．クライアントが A，B の 2 人の場合の流れを図 1 に示す．

(1) サーバーは参加するクライアント全員に未学習のグローバルモデルを提供する．
(2) それぞれのクライアントは所有するデータを利用し，受け取ったモデルの学習をローカルで行い，モデルの重みを更新する．
(3) クライアントは更新されたモデルの重みの情報のみをサーバーのモデルへ共有する．この際に，サーバー学習率 η_{server} と呼ばれるパラメータを利用し，サーバーモデルへの重みの学習率を決定する．サーバーはクライアントのモデルから集約した重みの情報に平均化の処理を行い，グローバルモデルのパラメータ情報を更新する．
(4) サーバーは更新されたグローバルモデルのパラメータ情報を再びクライアントのモデルに分配し，重みを更新する．この際に，クライアント学習率 η_{client} と呼ばれるパラメータを利用し，クライアントモデルへの重みの学習率を決定する．これによりそれぞれのクライアントのモデルはグローバルモデルから受け取った重みを持ち，クライアント全員のモデルが共通の重みを持つよう更新される．
(5) （2）〜（4）のサーバーとクライアントの持つモデルのパラメータの更新と受け渡しを 1 ラウンドとし，指定したラウンドの数だけ繰り返すことにより，最終的にサーバーの元にあるグローバルモデルの学習が完了する．そして最終的に学習が完了したグローバルモデルを用いてプロジェクトのバグ予測を行う．

このような流れで連合学習を用いた CPDP では，差分プライバシーが確保される．差分プライバシーとは，個々のデータの推測や識別をできないようにする方法である．上記のアプローチであれば，データ自体の共有ではなくクライアントの持つ全てのデータの学習結果を重みとして共有するため，クライアントの持つ個々のデータ自体の推測は不可能となる．

本研究では連合学習の実装フレームワークとして，Tensorflow Federated[*1]を利用した．Tensorflow Federated は，ニューラルネットワークを用いたモデルに対応し，連合学習のシミュレーションをローカル環境で行うことができるフレームワークである．そのため，理論的にはニューラルネットワークでアルゴリズムを再現することでどのような機械学習アルゴリズムも利用可能ではあるが，連合学習に組み込むことのできる再現の容易な CPDP の機械学習モデルは限られる．本研究では連合学習のモデルとして，Sigmoid 関数を利用した単一ノードのニューラルネットワークモデルで Logistic Regression を再現し，連合学習に組み込んだ FLR モデルを使用した．実験で使用した連合学習のパラメータについて，損失関数の推移を確認し，ラウンド数は 100 とした．

4. 調査結果

4.1 連合学習による CPDP のモデルは従来のプライバシー保護を考慮した CPDP 手法と比較し、どれほどの性能差があるか

目的：CPDP に利用するための連合学習アプローチの実現可能性をはかるため，FLR モデルによる CPDP モデルが従来の CPDP モデルと比較しどれほどの性能差があるかを明らかにする．

[*1] https://www.tensorflow.org/federated

アプローチ：本研究の手法（Federated Learning，FL）と比較する既存の CPDP の手法に 3 種類の手法と予測性能を比較した．比較する手法は，教師あり学習（Supervised Learning，SL），プライバシー保護を考慮した CPDP モデルとして LACE2 [6] を用いた教師あり学習のモデル，そして教師なし学習（Unsupervised Learning，UL）の 3 種類である．本調査では，これらを合わせた 4 種類の手法を用いて以下の 8 種類のモデルの間で比較した．

- FL: 3.2 節で説明した Federated Logistic Regression（FLR）の連合学習モデル
- SL: Logistic Regression（LR），Random Forest（RF），K-Nearest Neighborhood（KNN）の教師あり学習の 3 種のモデル
- LACE2: Pater ら [6] の提案した LACE2 のアルゴリズムを上記の教師あり学習のモデルに適用した 3 種のモデル
- UL: Yuming ら [10] の提案した Manualdown の教師なし学習モデル

教師あり学習の比較対象としたモデルについて，Logistic Regression は Federated Logistic Regression との連合学習アルゴリズムの有無の性能比較として採用した．Random Forest は CPDP の教師あり学習の中で高い性能を持つモデルの 1 つとして採用した．K-Nearest Neighborhood は Pater ら [6] のアルゴリズムで LACE2 が適用されたモデルであるため採用している．また，教師なし学習の比較対象とした Manualdown は Yuming ら [10] が新たな CPDP 研究のベースラインモデルとして Manualdowm と比較することを推奨しているため採用している．

本調査でのデータセットは 3.1 節で説明した全てのデータを利用した．まず FL，SL，LACE2 の 7 種類のモデルについては，グループ内のプロジェクトのうち 1 つのプロジェクトの最新バージョンをテストデータ，残りのプロジェクトを学習データとして，学習データをランダムにサンプリングして 100 個のモデルを作成した．テストデータには最新のバージョンのみを利用したため，59 個のプロジェクトのうち 25 個のプロジェクトがテストデータとなった．複数バージョンを持つプロジェクトがテストデータとなった際には，そのプロジェクトの過去バージョンのプロジェクトは学習データとして利用しなかった．これは，手元に過去のデータが無い状況の元で実験をするである．また，UL のモデルは教師なし学習であるため，テストデータから 1 個のモデルを作成した．

本調査の評価指標には多くの CPDP 研究で用いられている指標の 1 つである，AUC を利用した．また，モデルの性能の順位尺度を図る指標として，Scott-Knott Effect Size Difference (ESD) Test [11] による検定を行った．Scott-Knott ESD Test は，階層クラスタリングを利用し，モデルのパフォーマンスの平均値を統計的有意差（ $\alpha = 0.05$ ）のあるグループに分割するアプローチである．複数のモデルの指標が同じ 1 つの順位のグループに割当られることもあるが，それぞれの指標は複数の順位間をまたぐようなグループ分けにならず，明確にランク分けできることが利点である．本調査では，モデルごとの 100 回の AUC に対して検定を行ったため，1 個のモデルのみを作成する教師なし学習 UL を除く 7 種類のモデルに対して検定した．

結果：それぞれのモデルの AUC を，データセットのグループごとに中央値を算出した結果を表 2 に示す．1 行目は分類器の種類と分類器を表し，1 列目はデータセットのグループを表す．また，教師なし学習を除く 7 種類のモデルに対して Scott-Knott ESD test を実行した際の結果を図 2 に示す．Scott-Knott ESD test による順位グループの割当では，順位が高いほど良い予測性能のグループにモデルが割り当てられたことを示している．図 2 は，モデルが 25 個のテストデータそれぞれでの検定により AUC で順位のグループ分けがなされた際に，モデルが割り当てられた各順位の累計回数を表している．

教師あり学習（SL）について着目すると，FLR モデルの AUC は教師あり学習の Logistic Regression（LR）モデルよりも低い結果となった．また，図 2 の検定結果について見ると，FLR モデルはどのテストデータの検定でも LR モデルよりも高い順位のグループに属することはなかった．これは，LR モデルがそのままのデータを学習データとして利用できるのに対し，FLR モデルではクライアントが学習したモデルのパラメータを集約したモデルを利用することにより，データ自体の情報が秘匿されている影響だと考えられる．次に，LACE2 についての結果は，LACE2 を適用したすべてのモデルの AUC よりも FLR モデルの AUC が高かった．また，検定結果を見ると FLR モデルは全てのデータで LACE2 を適用したどのモデルよりも高い順位のグループに属していた．この性能差については LACE2 のアプローチに含まれるデータ削減による情報量の低下による影響が大きいためであると考えられる．最後に，教師なし学習（UL）については，FLR モデルの AUC が教師なし学習の Manualdown モデルよりも低い結果となった．

これらの結果から，FLR モデルは教師あり学習のモデルと教師なし学習のモデルに対しての予測性能は高くないが，プライバシー保護アルゴリズムの LACE2 を取り入れた教師あり学習のモデルに対しての予測性能は高いと考えられる．また，図 2 の検定結果を見ると，FLR モデルは 25 個のテストデータに対して予測性能が 1 位のグループに 5 回 (20%)，2 位のグループにも 12 回 (48%) 割り当てられていることから，教師あり学習のモデルと比較してもある程度の予測性能があると考えられる．

表 2 データセットグループごとのモデルの AUC の中央値

	FL(FLR)	SL(LR)	SL(RF)	SL(KNN)	LACE2(LR)	LACE2(RF)	LACE2(KNN)	UL(Manualdown)
AEEEM	0.684	0.718	0.739	0.648	0.471	0.456	0.512	0.716
METRICSREPO	0.679	0.679	0.697	0.602	0.515	0.502	0.576	0.693
RELINK	0.712	0.740	0.679	0.677	0.293	0.572	0.583	0.760
SOFTLAB	0.770	0.796	0.775	0.778	0.632	0.568	0.679	0.786

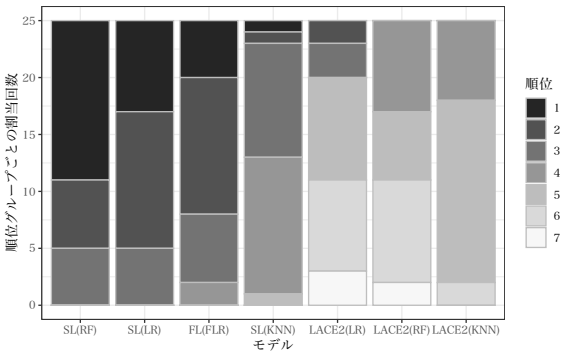


図 2 モデルの AUC に対する Scott Knott ESD test の順位グループの分布

FLR モデルは LACE2 を用いた教師あり学習より高い予測性能を示したが，教師あり学習と教師なし学習のモデルに対しては予測性能は低い結果となった．しかし，FLR モデルは利用したデータの 68%で予測性能の順位グループの 2 位以上に割り当てられ，教師あり学習のモデルと比較してもある程度の予測性能があると考えられる．

4.2 サーバー/クライアント学習率によって、予測性能は向上するか

目的：FLR モデルのアプローチでは，サーバーモデルとクライアントモデルの間でモデルの重みの授受が複数回行われる．その際に使用される 2 つのパラメータが η_{server}，η_{client} である．η_{server} はクライアントモデルからサーバーモデルに重みを集約する際の学習率（図 1(3)）であり，η_{client} は逆にサーバーモデルからクライアントモデルに重みを更新する際の学習率（図 1(4)）である．2 つの学習率はともに 0 から 1 で表される値であり，機械学習における勾配降下法と同様にパラメータの更新時に以下のように計算される．w はパラメータ，η はサーバー学習率またはクライアント学習率，g はコスト関数の勾配を表す．

$$w = w - \eta \times g \tag{1}$$

これら 2 つの学習率はモデルのパラメータをどれだけ受け取るかにつながるパラメータであり，本調査ではこれらのパラメータとモデルの性能との関連を明らかにする．

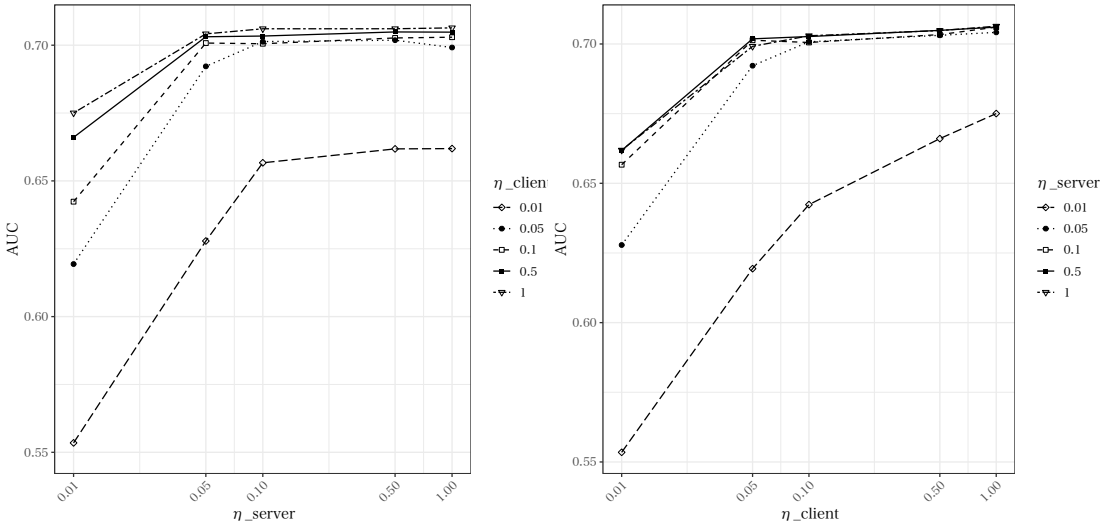
アプローチ：FLR モデルにおけるクライアントモデルとサーバーモデルに関するパラメータ η_{client} と η_{server} の値をそれぞれ [0.01, 0.05, 0.1, 0.5, 1.0] の値で組み合わせた際の FLR モデルの性能変化を調査する．データセットは全てのデータを利用し，RQ1 と同様の方法で 8 種類のモデルの学習，テストを行った．評価指標には AUC を用いた．

結果：η_{client} の値を固定し，η_{server} を変化させた際の FLR モデルの AUC の変化を図 3(a) に，η_{server} の値を固定し，η_{client} を変化させた際の FLR モデルの AUC の変化を図 3(b) に示す．それぞれ図の横軸は変化させた側のパラメータの変動を，縦軸は AUC のスコアを示す．図 3(a) から，クライアント学習率 η_{client} を固定し，サーバー学習率 η_{server} を変動させた結果，いずれの η_{client} においても，η_{server} が 0.01 から 0.05 の間が他と比べて大きな差が見られた．一方で，0.10 から 1.00 までの間では，大きな性能向上は見込めないことがわかる．図 3(b) から，サーバー学習率 η_{server} を固定し，クライアント学習率 η_{client} を変化させた結果についても同様に η_{client} が 0.01 から 0.05 の間での性能差は他と比べて大きい結果となった．これらのことから，η_{server} ，η_{client} いずれにおいても，0.05 までの値については CPDP の性能向上に影響が大きいといえ，0.10 より大きな値については大きな性能向上は見られないといえる．学習率が小さい間に性能が低くなったのは，学習率を低くしたことにより通常の勾配降下法と同様に，収束するまでの計算回数が大きくなったためであると考えられる．

サーバーとクライアントのパラメータの授受に関する 2 つの学習率 η_{server} ，η_{client} いずれにおいても，0.05 までの値については CPDP の AUC の性能向上に影響が大きく，0.10 より大きな値については大きな性能向上は見られなかった．

4.3 クライアントの参加が増えると、予測性能は向上するか

目的：連合学習のアプローチの特徴の 1 つが，クライアントの存在である．どれほどの数のクライアントが予測性能に影響するかを知ることは，連合学習を CPDP に適用する際の必要な参加者数やデータ数を推測する 1 つの指標となると考えられる．本調査では，クライアントの数が増えた際に，予測性能に影響があるのかについて確かめることで，クライアント数と予測性能の関係性を明らかにする．



(a) η_{client} **固定時の** η_{server} **ごとの AUC 変化**　**(b)** η_{server} **固定時の** η_{client} **ごとの AUC 変化**

図 3　η_{server} と η_{client} による AUC の変化

アプローチ：データセットのうち，プロジェクト数の多い Metrics Repo のグループのデータを利用した．モデルには FLR モデルを利用し，クライアント数を [1，10，20，30，40] と変化させた際の性能の変化を計測した．FLR モデルの学習とテストは RQ1 と同様の方法で行った．評価指標には AUC を用いた．

結果：METRICSREPO のプロジェクトそれぞれにおいてクライアント数を増加させたときの FLR モデルにおける AUC のスコアの変化を図 4 に示す．クライアント数の変化が 1 → 10 のときについてはどのプロジェクトにおいても性能向上が見られた．10 以上のクライアント数の増加に関しては，Ivy，PBeans，Synapse のプロジェクトはクライアント数の増加ごとに性能向上が見られた．一方で，その他のプロジェクトに関しては，必ずしもクライアント数を増加させることによる性能向上は見られず，性能向上が頭打ちとなる傾向が見られた．これは特徴量の統一のために同じグループ内のデータを利用したことで，クライアント数の増加に伴い同じグループ内のプロジェクトの参加数が増加していくため，クライアントのモデルから集約したグローバルモデルの重みが平均に近づいていったためであると考えられる．

> FLR モデルのクライアント数を増加させるとクライアント数が 10 まででは性能向上がみられた．一方で，クライアント数が 10 以上ではクライアント数を増加させることによる性能向上は見込めなかった．

4.4 どのような特徴量が連合学習を利用した CPDP モデルに影響を与えるか

目的：バグ予測モデルの解釈は，どのような特徴量がバグ発生率に与える影響が大きいかを明らかにし，バグの回避

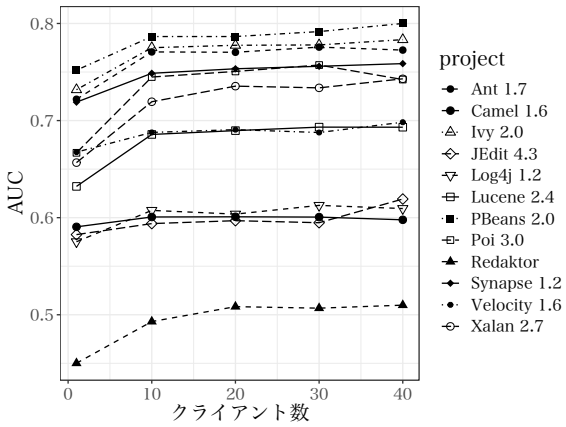


図 4　Metrics Repo のプロジェクトのクライアント数による AUC の変化

やソフトウェア品質の向上のための洞察に利用されている．しかし，連合学習を CPDP に適用してプライバシーを保護した際にどういったメトリクスが有効であるかは明らかでない．そのため本調査では，本研究で利用した環境における連合学習の CPDP モデルの予測性能にどの特徴量が影響を与えたかを明らかにする．

アプローチ：バグ予測モデルの解釈には，各特徴量がどれほどモデルの予測に寄与しているかを測定する特徴重要度を用いるのが一般的である．本研究ではニューラルネットワークのモデルを利用したため，モデルの内部構造を利用しない分析手法の 1 つであるパーミュテーション分析 [12] を用いてモデルの解釈を行った．パーミュテーション分析では，学習データのある特徴量を完全にランダムに並び替えてモデルを学習する．そのため，並び替えた特徴量にはもはやターゲットを説明する能力はなくなる．つまり，ある特徴量を並び替えて学習した際の予測性能が，元のモデルの予測性能に比べて大きく低下した場合，その特徴量が

表 3 削減された特徴量の概要

AvgEssential	すべての入れ子の関数またはメソッドの本質的複雑さの平均
SumEssential	すべての入れ子の関数またはメソッドの本質的複雑さの合計
RatioCommentToCode	コード行数に対するコメント行数の割合
AvgLineBlank	空白行数の平均

予測性能に影響していたと捉えられるため，誤差の分だけその特徴量が元のモデルの性能に寄与していたと考えることができる．

本実験では RELINK データセットを利用し，FLR モデルを対象にパーミュテーション分析を行った．実験の前処理として，データセット内の特徴量間に関連性があったため，McIntosh [13] らの特徴量削減の手法に従い，以下のように相関分析と冗長性分析により特徴量を削減した後にパーミュテーション分析を行った．相関分析では，説明変数間の共線性を低減できる．スピアマン順位相関検定（ρ）により，$|\rho| > 0.7$ を持つ互いに高い相関を持つ説明変数があった場合，他との相関係数の和が最小の説明変数のみを残した．冗長性分析では，相関分析で検出できない説明変数間の関連性を除くことが可能である．rms パッケージの redun 関数[*2]を利用し，閾値 0.9 以上の R^2 を持つ変数を削除した．2 つの特徴量削減の手法を実行した後，残った特徴量全てに対してパーミュテーション分析を行い，AUC による性能差を調査した．また，各特徴量の AUC の性能差に対して Scott Knott ESD Test を実行し，特徴量について統計的に性能に関する順位のグループ分けを行った．

結果：RELINK のデータセットに対して，特徴量を削減した後の特徴量の概要を表 3 に示す．元の特徴量 26 個から特徴量として残ったのは 4 個で，主に 2 つの種類の特徴量に分けられる．1 つはコードの構造に関する特徴量であり，AvgEssential と SumEssential が該当する．この 2 つの特徴量については，相関分析と冗長性分析によって他の多くの特徴量と関連性が高いとされており，削除された特徴量を見ると，AvgEssential は循環的複雑度の平均値などの，循環的複雑度に関する特徴量との関連が大きかった．SumEssential は，コード自体の行数やコードの実行行数のような，コード自体に関する特徴量との関連が大きかった．もう 1 つの種類の特徴量として，RatioCommentToCode と AvgLineBlank のコードの実行に関連しない特徴量である．これらはもともと関連性の高い他の特徴量が 1 つだけであり，いずれもコメント行数に関する特徴量との関連が大きかった．

これらの特徴量に対しパーミュテーション分析を行った結果として，図 5 に元のモデルの AUC と各特徴量のパーミュテーション後の AUC の差の箱ひげ図を示す．値が小さいほどパーミュテーションによる性能低下が起こっている．また，図 6 は AUC の性能差に対する特徴量ごとの Scott Knott ESD Test の結果であり，各特徴量が AUC の性能差に基づいた順位のグループに分けられている．

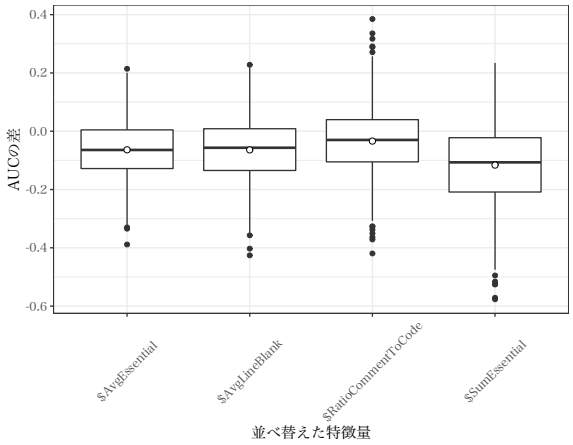


図 5 元のモデルとパーミュテーション後の AUC の差の箱ひげ図

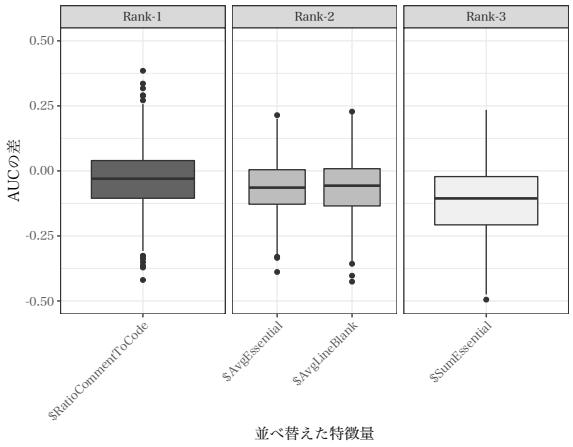


図 6 パーミュテーションを実行した AUC に対する Scott Knott ESD test の順位グループの分布

この結果から，SumEssential の特徴量を並び替えた際に元のモデルの AUC から最も大きく性能が下がり，FLR モデルの性能に大きな影響を及ぼす特徴量であることがわかった．SumEssential はコードに関する特徴量との関連が大きかったことから，FLR モデルの性能に影響を及ぼす特徴量は，コード自体の特徴に関する特徴量が重要であると考えられる．一方で，RatioCommentToCode の特徴量を並び替えた際には，他の特徴量の結果と比較しても AUC の低下が最も小さい結果となった．RatioCommentToCode 自体はコメント行数の割合の特徴量であり，他にコメント行数の特徴量との関連が大きかったことから，コメントに関する特徴量はバグに対しての影響が大きくないため性能への影響が少なかったと考えられる．

[*2] https://www.rdocumentation.org/packages/Hmisc/versions/4.7-0/topics/redun

本研究で用いたFLRモデルの予測性能に影響が大きいものとして，本質的複雑度やコード行数のようなコードに関する特徴量は統計的にも性能に大きな影響を及ぼすため重要な特徴である．一方でコード内のコメントに関する特徴量は予測性能に対して大きな影響はなかった．

5. 妥当性の脅威

内的妥当性 本研究で使用した連合学習の手法と比較した教師あり学習やLACE2，教師なし学習の各手法はアプローチを再現して実装したものであり，実装方法やパラメータにより性能への影響を与える可能性がある．また，連合学習で構築したLogistic Regressionのモデルはニューラルネットワークで再現したものであるため，ニューラルネットワークでないLogistic Regressionを連合学習で構築した場合と比較して結果に影響を与える可能性がある．

外的妥当性 本研究で用いたモデルの性能指標はAUCのみである．AUCは多くのCPDPに関する研究で利用されている指標の1つであるが，その他の性能指標についても今後調査が必要である．また，Tensorflow Federatedはシミュレーション環境での実行を可能にするフレームワークであり，実際にデータを分散させた環境ではないため，データを分散させた環境における調査を今後行う必要がある．

6. おわりに

本研究では，プライバシー保護を考慮したCPDPのモデルとして，連合学習にLogistic Regressionを搭載したFLRモデルを作成し，その性能や特性を明らかにする調査を行った．その結果として，FLRモデルはプライバシー保護アルゴリズムのLACE2を適用したモデルより高い予測性能となることを明らかにした．また，モデルごとの予測性能の検定の結果，FLRモデルは利用したデータの68%で，予測性能の順位グループの2位以上に割り当てられ，教師あり学習モデルと比較してもある程度の予測性能があることを明らかにした．次に連合学習に関するパラメータとモデルの予測性能に関して調査した結果，FLRモデルのサーバー/クライアント学習率は0.05までは性能向上が見られ，クライアント数については10以下のときAUCの性能向上が見込めることを明らかにした．最後に，本実験環境におけるFLRモデルの予測性能に影響が大きい特徴量を調査し，本質的複雑度やコード行数のようなコードに関する特徴量は性能に影響を及ぼすことを明らかにした．

今後の課題として，RQ3とRQ4の実験に関して，単一のデータセットでの実験のみの結果となっているため，データを追加し一般性を高める必要がある．また，RQ4の実験に関しては連合学習を適用しないCPDPと比較し，連合学習による特徴量の影響を確かめる必要がある．

謝辞 本研究の一部はJSPS科研費JP18H04097・JP21H04877，および，JSPS・国際共同研究事業（JPJSJRP20191502）の助成を受けた．

参考文献

[1] Britton, T., Jeng, L., Carver, G., Cheak, P. and Katzenellenbogen, T.: Reversible Debugging Software: Quantify the time and cost saved using reversible debuggers, Technical report, Cambridge Judge Business School (2013).
[2] Zimmermann, T., Nagappan, N., Gall, H., Giger, E. and Murphy, B.: Cross-Project Defect Prediction: A Large Scale Experiment on Data vs. Domain vs. Process, *Proceedings of the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, p. 91–100 (2009).
[3] Caglayan, B., Turhan, B., Bener, A., Habayeb, M., Miransky, A. and Cialini, E.: Merits of Organizational Metrics in Defect Prediction: An Industrial Replication, *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Vol. 2, pp. 89–98 (2015).
[4] Peters, F. and Menzies, T.: Privacy and Utility for Defect Prediction: Experiments with MORPH, *Proceedings of the 34th International Conference on Software Engineering*, p. 189–199 (2012).
[5] Peters, F., Menzies, T., Gong, L. and Zhang, H.: Balancing Privacy and Utility in Cross-Company Defect Prediction, *IEEE Transactions on Software Engineering*, Vol. 39, pp. 1054–1068 (2013).
[6] Peters, F., Menzies, T. and Layman, L.: LACE2: Better Privacy-Preserving Data Sharing for Cross Project Defect Prediction, *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering* (2015).
[7] Kone, J., McMahan, H. B., Yu, F. X., Richtarik, P., Suresh, A. T. and Bacon, D.: Federated Learning: Strategies for Improving Communication Efficiency, *NIPS Workshop on Private Multi-Party Machine Learning* (2016).
[8] Li, Z., Jing, X.-Y. and Zhu, X.: Progress on approaches to software defect prediction, Vol. 12 (2018).
[9] Richard O. Duda, Peter E. Hart, D. G. S.: *Pattern Classification*, Wiley (2012).
[10] Zhou, Y., Yang, Y., Lu, H., Chen, L., Li, Y., Zhao, Y., Qian, J. and Xu, B.: How Far We Have Progressed in the Journey? An Examination of Cross-Project Defect Prediction, *ACM Trans. Softw. Eng. Methodol.* (2018).
[11] Tantithamthavorn, C., McIntosh, S., Hassan, A. E. and Matsumoto, K.: The Impact of Automated Parameter Optimization on Defect Prediction Models, *IEEE Transactions on Software Engineering*, Vol. 45, pp. 683–711 (2019).
[12] Rajbahadur, G. K., Wang, S., Ansaldi, G., Kamei, Y. and Hassan, A. E.: The impact of feature importance methods on the interpretation of defect classifiers, *IEEE Transactions on Software Engineering*, pp. 1–1 (2021).
[13] McIntosh, S., Kamei, Y., Adams, B. and Hassan, A. E.: An empirical study of the impact of modern code review practices on software quality, *Empirical Software Engineering*, Vol. 21 (2015).