機械学習モデル構築のための データ・処理共有フレームワークの提案

小西宏樹 [1,a)]　川谷卓哉 [2,b)]　峯恒憲 [3,c)]

概要：機械学習モデル構築の工程は，主に，データの取得，前処理，学習モデルの構築，評価に分けられるが，この一連の作業は，個人の実行環境に依存していることが多く，再現実験や比較実験の障害となっている．本研究では，機械学習モデル構築の過程，結果を記録し，モデル構築時の課題やデータ，類似作業の検索と共有を可能とするフレームワークについて提案する．

1. はじめに

機械学習モデル構築の工程は，主に，データの取得，前処理，学習モデルの構築，評価に分けられるが，この一連の作業は，個人の実行環境に依存していることが多い．そのため，実験環境の再現にはかなりの労力が必要であり，再現したとしても，同一の処理を行っていることを保証することは難しい．また，そのような環境では，モデル構築の際に発生した課題や問題についても，共有化を図ることが難しいため，個人内に留めたままということもある．

近年，機械学習を用いたサービスは増加傾向にある．そして実社会でそのようなサービスを運営する場合，一回モデルを構築してしまえばよいということはなく，フィードバックを受けながらモデルの改善を図り，必要に応じてモデルの差し替えを行わなければならない．そういった場合にモデル構築手法が属人化していると，サービスの開発に支障が生じる．また，サービス運営に限らずとも，研究の引き継ぎなど，様々な場面で支障が生じる．

これらの問題を解決するため，本研究では，機械学習モデル構築のためのデータ・処理共有フレームワークの提案を行う．本フレームワークでは，機械学習を大きく分けて四つの工程として考える．

・機械学習に用いるデータの取得
・得られたデータを特定の形式に変換する処理
・学習モデルの構築
・構築されたモデルの評価

これらの工程について,生成されたデータやソースコード，学習モデル，学習モデルの評価をデータベースに登録することで，機械学習モデル構築における，過程，結果の記録を行う．

データベースに登録されたデータについては，タグを付与し，検索と共有を行えるようにする．モデル構築が個人の環境に依存していると，それに用いるデータセット，前処理とモデル構築のソースコードを死蔵する傾向にある．そのため，同一処理を行うプログラムがすでに存在しているにもかかわらず，複数人間で共有されていないため，同じプログラムを作成してしまう二度手間が発生する．そのようなデータやソースコードの再利用性が失われてしまう事態に対処するため，本フレームワークでは検索と共有を行えるようにする．

以上のような仕組みをもったフレームワークが，どのような機能を提供するのか，また，どのように使用するかについて本研究では述べる．

以下 2 節では関連研究について述べ，本稿の立場を明らかとする．3 節では本研究で提案するフレームワークが具体的にどのような機能を持つのかを述べ，4 節でそれら機能に伴うデータベーススキーマを示す．5 節ではフレームワークの基本的な使用方法，6 節ではバス運行プローブデータを対象としたフレームワークの具体的な使用方法を述べ，7 節でまとめと今後の課題を示す．

2. 関連研究

機械学習モデル開発における問題として，Sculley ら[1]の研究では，学習モデルのデータ依存性など様々な点が挙げられている．これら問題を解決するため，MLOps という概念が提唱されている．MLOps とは，開発部門と運用部門が協調してソフトウェア開発を行う DevOps と呼ばれる手法をもとに考えられた，機械学習モデル開発における DevOps である．学習モデルの構築だけでなく，実社会で運用されるまでのプロセスを焦点にした仕組みとも言える．

MLOps を踏襲したシステムの提案・開発は多数行われており，OSS として公開されているソフトウェアも存在している．

OSS として代表的なものの一つに MLflow[2]がある．MLflow とは，Databrick 社や Microsoft 社も開発に参加している実験管理ソフトウェアである．実験の過程の追跡・記録, 実験の再現, 学習モデルのデプロイの三点に関する API を主に提供している，また，実験概要を示すインターフェースも提供しており，実験で生成された中間生成物，ハイ

1 九州大学工学部電気情報工学科
Department of Electrical Engineering and Computer Science, School of Engineering, Kyushu University
2 九州大学大学院システム情報科学府
Graduate School of Information Science and Electrical Engineering, Kyushu University
3 九州大学大学院システム情報科学研究院
Faculty of Information Science and Electrical Engineering, Kyushu University
a) konishi.koki@m.ait.kyushu-u.ac.jp
b) kawatani.takuya@m.ait.kyushu-u.ac.jp
c) mine@ait.kyushu-u.ac.jp

©2021 Information Processing Society of Japan

パーパラメータなどを Web ブラウザで確認することができる．OSS としては他にも，DVC (Data Version Control)がある. DVC は Git を併用した実験管理ソフトウェアである．ソースコードだけでなくデータセットのバージョン管理も行うことができる．機械学習の一工程をステージと言う単位で分けて, 一連の工程（パイプライン）を定義している．パイプラインではステージの入出力関係を記録するため，ステージ間の依存関係を調べることができる．実験の過程の追跡・記録，実験の再現，リモートストレージ設定の三点に関する CLI コマンドを主に提供している．

本研究では，実験管理の機能を提供しつつ，データの検索・共有機能を有するフレームワークを提案する．先に挙げた実験管理ソフトウェアではデータの検索機能がなく，具体的にどのようなデータが記録・保存されているのか，他のユーザからはわかりづらい状況にある．本フレームワークは，データセット，ソースコード，学習モデルのメタデータを用いた検索・共有機能を提供して，データの再利用性を高め，類似作業の効率化を図る．

3. フレームワークが提供する機能

本研究で提案するフレームワークの概要図を図 1 に示す．フレームワークが提供する機能は，学習モデル構築の過程において生成されるデータ，プログラムを記録して，それぞれのデータにタグを付与し，管理することが主なものとなる．

前処理後のデータセットや生成された学習モデルなどは，個人のローカル環境で管理することが多く，第三者と共有を行う際に手間がかかる．そこで，本研究で提案するフレームワークにより，機械学習モデル構築における管理体制を提供することで，共有の手間を軽減できる．

フレームワークの構築には Python 言語，DBMS には MySQL を想定している．

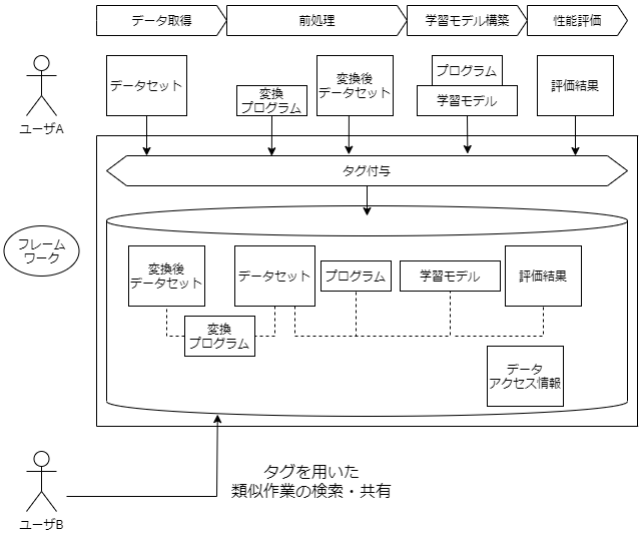

図 1. フレームワーク概略図

3.1 実験環境構築手順の登録・取得

Python 言語には，機械学習に関する豊富なライブラリが存在している．それらのライブラリは日々更新・バージョンアップしていくため，前回実行できたはずの処理が実行できなくなる，計算結果に齟齬が生じるなどの不都合が生じるときがある．こういった事態をあらかじめ防止するため，実験環境の構築手順をデータベースに登録する．構築手順についてはテキスト形式を想定しており，例えば，GitHub リポジトリのリンクを示す, Docker Hub で共有されているイメージの指定などを行う．

3.2 データセットの登録・取得

データセットがローカル環境で管理されている場合，第三者へのデータの譲渡や共有に手間がかかる．また，機械学習用に前処理されたデータセットがローカル環境にしかないという状況は日常的に起こっており，複数の人が複数の環境で同様の前処理を実施する無駄が生じている．

機械学習に使用するデータセットはデータベースに登録して，他のユーザとの共有を行うことで，再現性を担保する．

3.3 データ変換処理の登録・取得

あるデータセットをもとに，新たなデータセットが生成されたとする．このとき，新しく生成されたデータセットに加えて，使用されたデータ変換プログラムについてもデータベースに登録する．これにより，データベースのストレージ不足などが原因で，変換後のデータセットを破棄したとしても，大元のデータセットが存在していれば，復元が可能となる．

3.4 学習モデルの登録・取得

MySQL がサポートしているデータ型として，BLOB (Binary Large Object) 型というバイナリデータを扱えるものがある．機械学習で構築されたモデルは，バイナリ形式に変換することで，データベースに登録できる．

3.5 評価の登録・取得

対象とする問題や使用する学習器によって評価指標は様々である．教師あり学習は分類問題と回帰問題に分けられ，たとえば，分類モデルの性能評価指標として混同行列が挙げられる．しかし混同行列は，分類するクラス数に応じて，行数および列数が変化するため，評価結果を登録するテーブルの形式を一意に定めることができない．また，回帰モデルにおいても,特徴量の有意性を評価するときに，特徴量がデータセットごとに異なるため，同様の問題が生じる．以上の理由により，評価結果を登録するテーブルの形式は，ユーザが定義できるようにする．

©2021 Information Processing Society of Japan

3.6 アクセス情報の記録

データベースに格納されるデータについて，利用頻度を計測する目的として，アクセス情報の記録を行う．具体的には，アクセス日時，データの登録，追加，取得，削除の操作のうちどれを行ったのかを示すアクセス情報，アクセスしたデータについて記録する．こうしたアクセス情報を記録することで，データベースに登録されたデータについて，再利用可能かどうかを判定し，必要となるデータの推薦の仕組みの開発に繋げていく．

3.7 変数検索

本フレームワークでは，同データベースに多数のデータセットが集まる．たとえば，高度交通システム向けのデータセットでは，走行履歴情報（プローブデータ）や路線情報などが含まれるが，そのようなデータセットの中で，タイムスタンプや位置情報など，同系統の特徴量をもつデータセットを組み合わせることで，新たな学習モデルの構築や拡張が期待できる．後述するタグ機能でデータセットを検索することで，組み合わせが可能なデータセットの推薦や提供も可能とする．

3.8 データのタグ付け

データベースに登録するデータセット，モデル，ソースコードについて，タグ付けすることで類似作業の分類を行う．タグ付けの方法については，ユーザが手動で行うものと，フレームワークが自動で行うものの二種類を想定する．タグには，データセットの列名やモデル構築に用いた学習器，処理の種類を想定している．

タグ付けの方法によって利点と欠点が存在する．まず，手動の場合，タグの名づけに，ある程度の自由度が与えられる．一方で，タグ名に統一性がなくなる点や，付けたタグ名の再利用が，忘却などにより難しくなる事も起こりうる．自動で行う場合，タグ付けの基準の規定が必要となるが，タグの付け忘れなどの人為的ミスは起こらず，タグ命名の統一性も担保される．

3.9 課題の記録・蓄積

機械学習モデルを構築する際に生じた課題について，ユーザに記録を促す．記録・蓄積された課題の関連性を調べることで，課題解決優先度の理解が進み，より効率的に作業を行うことができる．データベースに登録をすることにより，他ユーザと課題の共有も図ることができる．

3.10 データ追加時の自動データ変換

データセットには，更新・追加のない静的なものだけでなく，日ごとにデータが増加していく動的なデータセットも存在する．データベースにデータを追加するたびに，データ変換を手動で行うのは手間がかかるため，データセットテーブルに新たなデータを追加したとき，あらかじめ，欠損値の補完，除去，特徴量の標準化もしくは正規化などのデータ変換プログラムを設定することで，自動でデータ変換を行い，所定のテーブルに保存する．

4. データベーススキーマ

3．で述べたフレームワークが提供する機能を踏まえて，目的の処理を行えるようなデータベーススキーマを示す．

4.1 機械学習対象ごとのデータベーススキーマ

機械学習対象ごとのデータベースには，実験環境構築手順，機械学習モデル構築に使用されるデータセット，ソースコード，構築された学習モデル，評価結果についてのテーブルが作成される．また，それらの工程を関連付けるための，機械学習トレーステーブルも作成する．スキーマ図を図 2 に示す．属性名の後についている記号(FK)は，その属性が外部キーであることを示している．

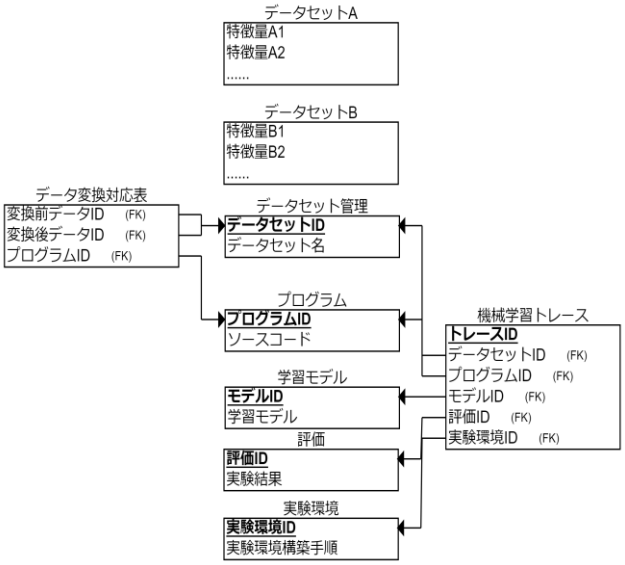

図 2. 機械学習対象ごとのデータベーススキーマ

4.2 アクセス情報記録用のデータベーススキーマ

アクセス情報記録用テーブルの基本スキーマは，アクセス日時，データの登録，追加，取得，削除を示すアクセス種類，外部キーとなっている．スキーマ図を図 3 に示す．

©2021 Information Processing Society of Japan

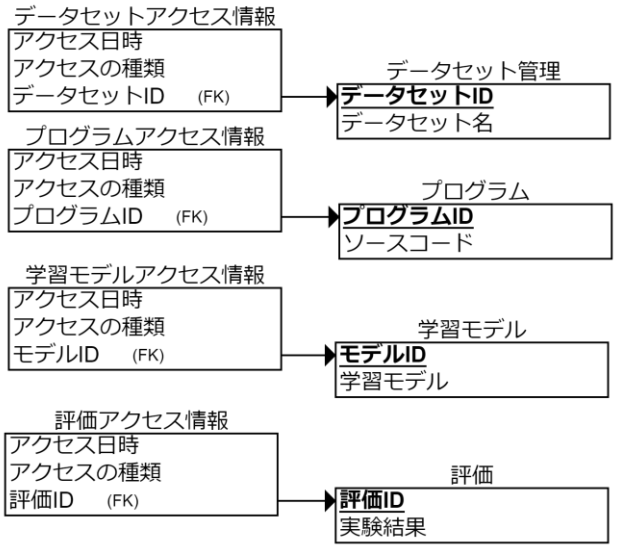

図 3. アクセス情報記録用のデータベーススキーマ

4.3 タグ用のデータベーススキーマ

タグ用テーブルの基本スキーマは，二十種類のタグと外部キーから構成される．機械学習モデル構築にあたって重要な要素である，データセット，プログラム，学習モデル，評価のデータにタグを付与する．データセットタグには特徴量，プログラムタグには前処理，訓練など処理の種類，学習モデルタグには学習器の種類，評価タグには，学習モデルの評価指標などを想定している．また，タグ数を二十としているが，タグ数に関する詳細な検討は今後の課題である．スキーマ図を図 4 に示す．

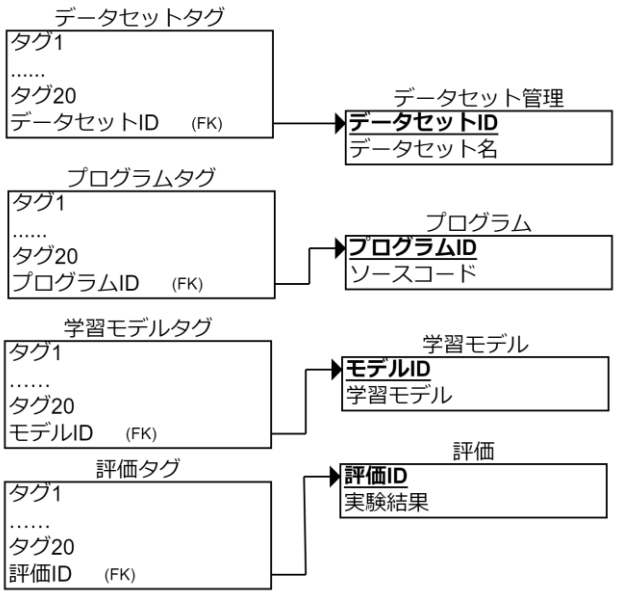

図 4. タグ用のデータベーススキーマ

5. 想定する使用例

これまで，機械学習モデル構築における過程，結果を記録するフレームワークの機能を示した．また，その機能を実現するためのデータベーススキーマも示した．本節では，フレームワークを利用した，新規学習モデル構築，再現実験，比較実験を行う際の手順を示す．

5.1 新規学習モデル構築

新たに機械学習モデルを構築する際には，他ユーザがその学習モデルについて再現・比較実験を行えるように，フレームワークを通して，過程，結果の登録をする．データベースに登録されたデータについては，順次検索タグの付与が自動で行われる．ユーザからのタグ付与も行うことができる．図 9 に新規モデル構築時のシーケンス図を示す．

5.2 再現実験

再現実験においては，論文等で示された工程に従い処理を進め，結果に齟齬が生じていないか確かめなければならない．そのためには，ソースコード，前処理後データセット，学習モデル，評価の検証を行う必要がある．図 10 には，再現実験時におけるシーケンス図を示す．また，具体例として，iris データセットを用いた機械学習について，再現実験を実際に行うところを示す．

5.2.1 iris（アヤメ）品種分類器の構築

iris（アヤメ）は花の品種である．iris データセットでは，このアヤメのうち，ヒオウギアヤメ，ブルーフラッグ，コバノズイナの三種類のガクの長さ・幅，花弁の長さ・幅を計測・記録したものを特徴量として保持している．このデータセットについて，ランダムフォレストを用いたモデルを構築する．木の深さは 2 として，データ数 150 個のうち，訓練データを 120 個，テストデータを 30 個に振り分けて，評価を行う．次から，再現実験までの手順を述べる．

まず，iris データセットをデータベースに登録する．図 5 に実際にデータベースに格納されているデータセットの一部分を示す．

sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	species
5.1	3.5	1.4	0.2	0
4.9	3	1.4	0.2	0
4.7	3.2	1.3	0.2	0
4.6	3.1	1.5	0.2	0
5	3.6	1.4	0.2	0

図 5. iris データセットの一部分

次に前処理として， iris データセットを訓練データとテストデータに分割する．この前処理のプログラムを，データベースに登録する．データセット管理テーブルとデータ変換対応テーブルを図 6 に示す．X_train，y_train は訓練データで，X_test，y_test はテストデータである．また，before_df_id は変換前のデータセットを表しており，after_df_id が変換後のデータセットを表している．code_id

©2021 Information Processing Society of Japan

は，変換に用いたソースコードを示している．

今回の例では，iris データセットを訓練データとテストデータに分割する処理を行ったので，一対多の対応となっている．

dataset_id	dataset_name
1	iris
2	X_train
3	X_test
4	y_train
5	y_test

before_df_id	after_df_id	code_id
1	2	1
1	3	1
1	4	1
1	5	1

図 6．前処理実行後のデータセット関連テーブル

前処理を終えた後，モデルの訓練を行い，その訓練したモデルとその評価結果，ならびに，全体のソースコードをデータベースに登録する．

最後に，データセットの登録から評価結果登録までを一つの工程としてデータベースに登録する．図 7 に一連の工程が記録されたテーブルを示す．

trace_id	dataset_id	code_id	model_id	result_id
1	1	2	1	1

図 7．iris（アヤメ）品種分類器の構築工程テーブル

5.2.2 iris（アヤメ）品種分類器の再現実験

trace_id を指定することで,モデル構築に関連するデータ一覧を取得することができる．そこで得られたソースコードをもとに再現実験を行い，実際にデータベースに登録されているモデル，または評価結果と比較を行う．図 8 に，ソースコード中のモデル構築処理と，データベースからモデルを読み込む処理を示す．

これら新たに構築した学習モデルと，データベースから読み込んだ学習モデルについて，同一の評価結果が得られることを確認している．

```
# 学習モデル構築
clf = RandomForestClassifier(max_depth=2, random_state=0)
clf.fit(X_train, y_train)
# 予測
y_pred = clf.predict(X_test)
print(y_pred)

# 学習モデル読み込み
clf_restore = pydbm.load_model(model_id)
# 検証
y_pred = clf_restore.predict(X_test)
print(y_pred)
```

図 8．モデル構築・読み込み処理

5.3 比較実験

あるユーザが学習モデルを構築して，別のユーザがそのモデルをもとに比較実験を行う場合を考える.具体的には，データベースに登録されているデータセットについて，新たな変数を追加して，そのときの学習モデルの精度がどの程度向上したのかを比較する場合を考える．図 11 に，比較実験時におけるシーケンス図を示す．ユーザは，機械学習モデルの構築に用いられたデータベースに登録済のソースコード，データセットを取得する．得られたデータセットに新たな変数を追加して，前処理を実行し，学習モデルを構築する．そうして構築されたモデルの性能を評価し，前回の実験と比較する．

新たに変数が追加されたデータセットは，その変数を追加したプログラムとともに，データ変換対応テーブルに登録する．それから，新たなデータセットや使用したソースコード，構築したモデルとその評価結果について，新規の trace_id を発行し，機械学習の一工程として管理する．

この一連の作業について，フレームワークを使用することで，データセットをはじめとした様々なデータを，一元的に管理することができるため，データセットの移行，ソースコードの共有などの作業の効率化を図ることができる．また，機械学習モデル構築において，対象とする問題設定が同一であるため，類似作業が多数存在することも想定される．そこで，本フレームワークの機能の一つである，類似作業の検索を行うことで,さらなる効率化が期待できる．

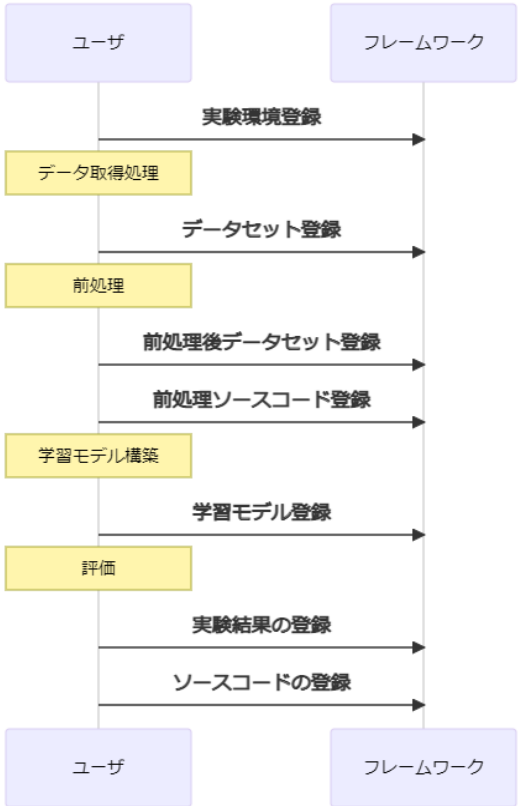

図 9．新規モデル構築時におけるシーケンス図

©2021 Information Processing Society of Japan

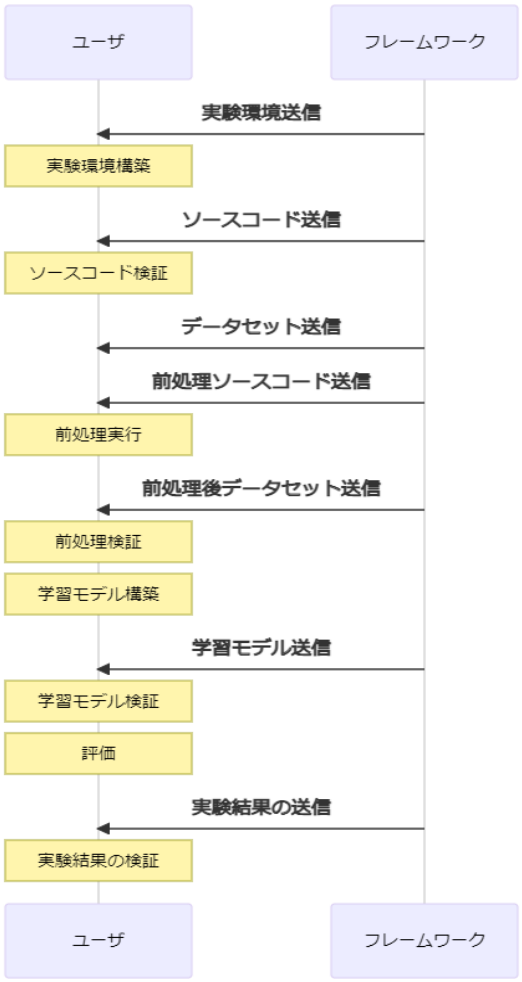

図 10. 再現実験時におけるシーケンス図

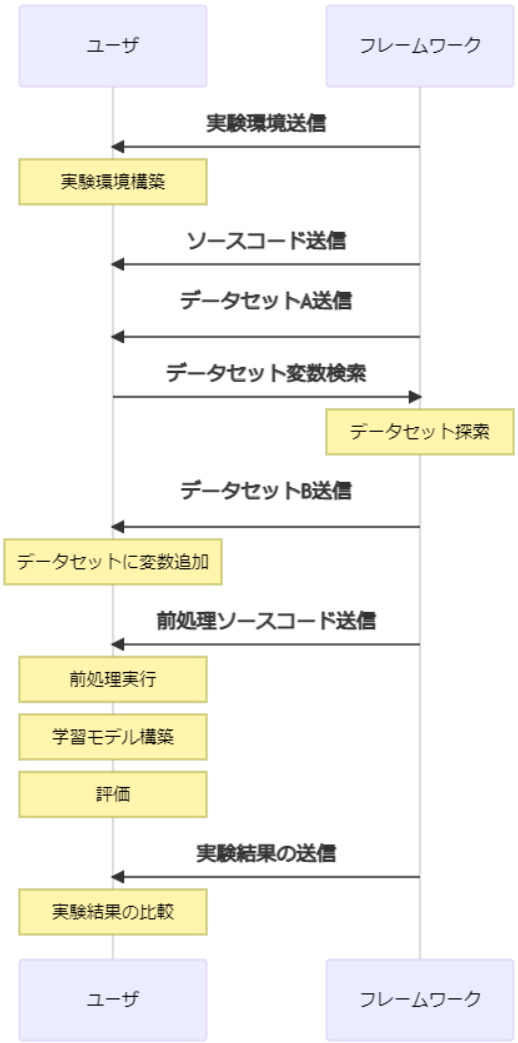

図 11. 比較実験時におけるシーケンス図

6. 具体的な実用例

5. 想定する使用例では，学習モデルの構築に焦点を当てて，フレームワークの使用例を示した．このセクションでは，バス運行プローブデータを対象としたフレームワークの，実際の運用例について述べる．

6.1 バス運行プローブデータの性質

本研究で述べるバス運行プローブデータとは，昭和自動車株式会社の協力のもと，九州大学線を走行する路線バス車両 42 台から収集される走行記録データのことである．走行記録データは，車両識別 ID，時刻，緯度，経度，速度で構成されており，200m 走行するたびに記録される．記録データの特性として，速度 9[km/h]未満で走行していること，また停止したことをデータから直接確認することができない点が挙げられる．急ブレーキや急ハンドルを検知すると，加速度情報を追加した走行データを記録する[3]．緯度，経度の計測には GPS を用いているが，時折，無視できないほどの外れ値を記録していることがある．

表 1. 特徴量付加・算出にかかる時間

	変換時間 (1 日)[s]	変換時間 (一ヶ月)[s]
データ読み込み	0.28	142.09
測地系変換	0.0043	0.040
バス停情報付加	0.20	5.32
通過･停車時間算出	7.72	1894.04
所要時間算出	10.46	3245.76

6.2 バス運行プローブデータの変換

バス運行プローブデータは，前述したとおり，バスの速度や位置情報など，バスの状態に関するものしか記録されない．そのため，実際にプローブデータを活用するには，最寄りのバス停情報をユーザ側で追加するなど，周辺環境に関する特徴量を付加しなければならない．川谷ら[5]の研究では，プローブデータについて以下の処理を行い，特徴量を付加・算出している．

・GPS データを日本測地系から世界測地系に変換

©2021 Information Processing Society of Japan

・バス停近傍の通過情報を付加
・各バス停通過・停車時刻の算出
・バス停間の移動時間算出
・各時刻における天候データの追加

バス運行プローブデータは，バスが運行するたびにデータが記録・蓄積されていく．表 1 に，2021 年 4 月 1 日から 2021 年 4 月 30 日までの一ヶ月間のプローブデータならびに，2021 年 4 月 1 日の一日間プローブデータのそれぞれに対して特徴量を付加・算出する際にかかる時間を示す．

バス運行プローブデータは，各バス，日付ごとに CSV ファイルで管理されているため，データ読み込み時間が大きくなる．また，バス停通過・停車時刻算出，バス停間所要時間算出にもかなりの時間がかかっている．

以上のことから，こまめなデータ変換の必要性がわかる．しかし，3.10 で述べたように，データが追加されるたびに，データ変換プログラムを実行することは手間である．そこで本フレームワークの機能の一つのデータ自動変換を利用し，プローブデータが保存されるデータベーステーブルに変換プログラムを設定することで，自動で上記の特徴量付加・算出を行う．

6.3 バス運行プローブデータの抽出

バス運行プローブデータは，バスの走行本数として一ヶ月間におよそ一万四千本増加する．大量のデータが存在する中，目的に応じたデータ解析を行うためには，データの条件抽出が重要である．フレームワークが提供する機能である，データの自動変換を活用することで，効率的な条件抽出が可能となる．

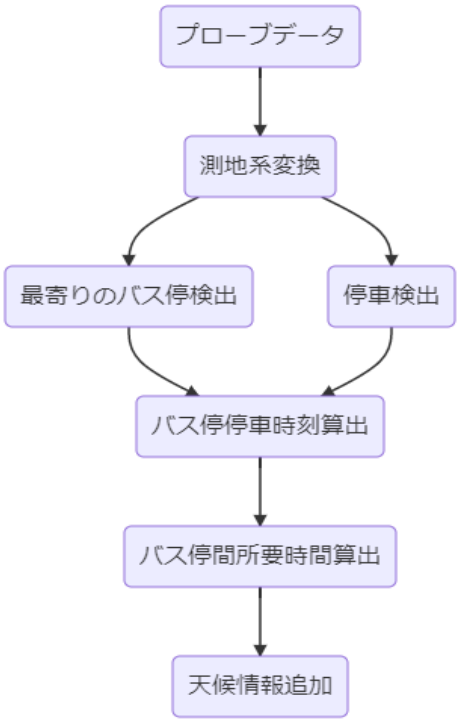

図 12. プローブデータ特徴量付加手順

プローブデータに対して条件抽出を行う例として，バスの通常運行において障害となる事例の抽出が挙げられる．たとえば，バスの遅延や道路の渋滞，GPS で計測した位置情報に外れ値が発生した場合などが相当する．このような障害となる事例の解析と解析結果に基づく対策や改善案の策定を行うため，障害となる事例を検出するための条件定義を行い，プローブデータから目的の事例に関連するデータを抽出する．

6.3.1 バスの遅延状況

バスの遅延状況を表す特徴量として，各バス停間にかかる移動時間を考える．まず，各バス停間の移動時間について，平均値や分散などの基本統計量を求める．次に，バスに遅延が発生したと判定する基準として，標準偏差 σ を用い，移動平均時間に σ を加えた移動時間を閾値とする．設定した閾値以上の移動時間が生じた場合，バスに遅延が発生したと判定して，その便のプローブデータを抽出し，別のデータベーステーブルに登録を行う．

6.3.2 渋滞の検知

バス運行プローブデータの性質として，バスが 200m 走行するたびにデータが記録される，という点を挙げた．この性質のため，渋滞によりバスが長時間停車していたとき，プローブデータのレコード間に大きな時間差が生まれる状況が考えられる．渋滞の検出にはこの特性を利用して，レコード間に一定時間以上の差がある場合を，渋滞が発生したプローブデータとして抽出する．

6.3.3 GPS 外れ値の除去

各路線の緯度・経度情報と照らし合わせて，GPS データの外れ値が生じたかどうかを判定，除外する．しかし，こういった外れ値により除外されたデータを収集することで，何かしらのパターンの解析に繋がる可能性があるため，別のデータセットテーブルに保存することも考える．

6.4 主格データインターフェースの提供

プローブデータから生成される新たに生成されるデータについて，バスの位置情報，速度，移動時間など主格的な役割を果たすデータについてのインターフェースを提供する．具体例としては，ある便のバスの位置情報・渋滞発生地点をマッピングする，などである．こうしたインターフェースを提供することで，GPS の外れ値を視覚的に捉えることも可能となる．

7. まとめと今後の課題

本研究では，機械学習モデル構築の過程，結果を記録し，

©2021 Information Processing Society of Japan

再現実験や比較実験を手助けする機能，また，類似作業の検索・共有が可能となる機能を提供するフレームワークについての提案を行い，再現・比較実験を行う際のシーケンス図を示した．さらに，バス運行プローブデータを対象としたフレームワーク活用の具体例を示し，本フレームワークを利用することによるメリットを述べた．

今後の課題としては，次が挙げられる．

まず第一点として，データベーススキーマの検討である．本フレームワークでは DBMS に MySQL を採用しているが，MySQL は NoSQL をサポートしていないため，キーバリュー型のスキーマについて検討していない．今後のフレームワーク開発において，十分な検討を進めなければならない．

第二点として，検索機能が挙げられる．現在提案している検索方式では，各データにタグを付与するという簡素な方式をとっているが，類似作業の検索と共有という本フレームワークの目的には，十分ではないように思われる．

第三点として，OSS の導入がある．関連研究に OSS として MLflow と DVC を挙げた．これらの実験管理ソフトウェアを本フレームワークに組み込むことで，データの検索・共有機能開発に注力することができる．

これらを今後の課題として，解決を図っていきたい．

謝辞

本研究の一部は科研費 JP21H00907，JP19KK0257 の支援を受けました．ここに感謝いたします．

参考文献

[1] D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips, Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-Francois Crespo, Dan Dennison, *Hidden Technical Debt in Machine Learning Systems*, NIPS(2015)

[2] Matei Zaharia, Andrew Chen, Aaron Davidson, Ali Ghodsi, Sue Ann Hong, Andy Konwinski, Siddharth Murching, Tomas Nykodym, Paul Ogilvie, Mani Parkhe, Fen Xie, Corey Zumar, Databricks Inc. *Accelerating the Machine Learning Lifecycle with MLflow* IEEE Data Engineering Bulletin, December 2018.

[3] 川谷卓哉，佐藤祐大，峯恒憲，高予測精度と低収集コストを実現するためのバスプローブデータ収集地点の選定，第85 回 ITS 研究会(2021)

©2021 Information Processing Society of Japan