1B-04

機械学習ソフトウェア時代のバージョン管理システムの提案

千々和大輝　　馬越健治　　井上知洋

日本電信電話株式会社

{daiki.chijiwa.mk, kenji.umakoshi.xb, tomohiro.inoue.ak}@hco.ntt.co.jp

1　概要

近年，機械学習ソフトウェアの開発は著しく進展している．とくに深層学習技術の発展により，画像や音声，自然言語の高精度な認識および自動生成が可能となり，実用的なソフトウェア開発においても強力な技術要素となっている．一方で，通常のソフトウェア開発とは異なる側面として，学習データの変化や学習時のハイパーパラメータの設定によって学習アルゴリズムの性能が大きく変化するという性質が知られている．このため学習アルゴリズム・学習データ・ハイパーパラメータなどの組み合わせと，その結果として得られる学習済みモデルや評価結果などの学習結果を長期間にわたって記録し，最適な組み合わせを見つける必要がある．

本稿では，このような機械学習アルゴリズムの開発フローを支援するためのバージョン管理システムに求められる要件について議論し，それを満たすバージョン管理システムを提案する．

2　要件

本節では，機械学習アルゴリズムの開発フローを支援するためのバージョン管理システムに求められる要件について議論する．

まず機械学習アルゴリズム開発においてバージョン管理の対象とするデータは，学習プログラムと**実行時データ**の２種類に分類することができる．ここで実行時データとは，学習データや前処理済みデータ，実行時に読み書きされる設定ファイル，モデルの中間保存ファイルなど，プログラム実行中に入出力される，プログラム自身を除いたデータを指すこととする．機械学習開発においては，たとえば異なる学習アルゴリズムと異なる学習データやハイパーパラメータを様々に組み合わせて学習を繰り返すため，バージョン管理システムはこの２種類のデータを**独立管理**し，自由に組み合わせられる必要がある．これが要件の１つ目である．

次に要件の２つ目として，記録された学習結果の再現性・冪等性を保証するために，学習プログラムを**隔離実行**できることが挙げられる．これにより実行中の不用意なプログラム改変，あるいは中間出力データの不用意な改変などによる実行結果への干渉を防ぐことができ，実行時データのバージョン管理機能における再現性を保証できる．仮に隔離実行を行わない場合，同一レポジトリ内で複数個の学習プロセスを並列に実行すると一方がもう一方の中間出力を書き換えてしまうなどにより容易に干渉が起こり，もう一度学習をやり直さなければならず開発効率の低下に繋がる．さらにこのような他プロセスからの干渉による学習結果の変化に気づけなかった場合には，学習の再現性が損なわれることになる．

３つ目の要件として，実行時データの**自動記録機能**がある．実行時データの記録が必要となるタイミングとしては，

(a) 学習プログラム実行前にユーザが学習データや学習のベースとなる学習済みモデルなどを記録する場合
(b) 学習プログラムを実行して終了するまでの間に入出力の対象となったデータを記録する場合

の２パターンあるが，(a) については，ユーザがデータを追加したいタイミングで手動で記録するのが自然である．一方 (b) の場合については，仮に学習終了時にユーザが手動で記録するという運用方法を取ってしまうと，一度の学習には一般に数時間から数日にわたる長時間の計算を要するため，記録忘れや記録内容の誤りなどのヒューマンエラーが生じる可能性がある．その結果，後から学習結果に参照できず再実行が必要となったり，不正確な結果を参照してしまい再現性を保証できないということにつながってしまう．したがって，実行時データの入出力はリアルタイムに記録し，このようなヒューマンエラーが入り込む余地を排除することが求められる．

またこれらの要件を実現する上で，空間効率を損なうデータ配置なども避ける必要がある．学習データや学習済みモデルなどのデータ容量は数百 MB から数百 GB と非常に大きいため，仮に同じデータを重複して保存するような設計ではすぐにストレージ領域を圧迫してしまう．

3　提案システム

3.1　提案システムの概要

本システムは，機械学習プログラムのソースコードだけでなく学習に関わる実行時データ及び実行履歴を記録し，任意時点での結果を参照できるようにするためのバージョン管理システムである．本システムは次の機能を持つ．

- 学習プログラムと実行時データの独立管理機能
- 学習プログラムの隔離実行機能
- 実行時データの自動記録機能

またこれらの機能を実現するにあたって **Unionfs+** というファイルシステムを実装することで，学習プログラム側に一切の変更を加えることなく本システムの導入を可能にした．以下で詳細について説明する．

3.2　Unionfs+

Unionfs+ は，Unionfs [3] をベースとしたファイルシステムであり，複数ディレクトリを重ね合わせて単一のディレクトリとしてアクセスできるようにする重ね合わせ機能を提供する．

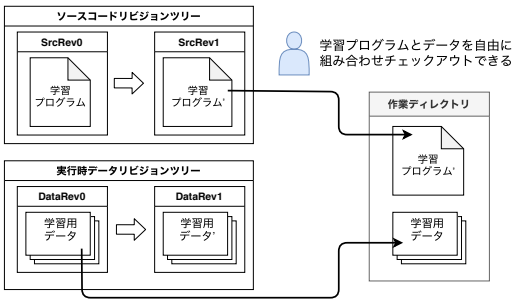

図 1: 学習プログラムと実行時データの独立管理

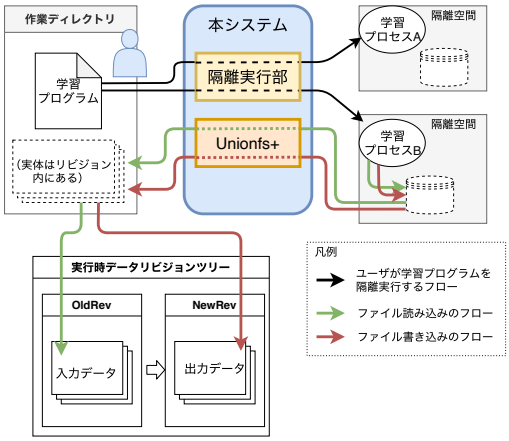

図 2: 隔離実行および自動記録機能

Unionfs との相違点として，書き込み要求だけでなく読み込み要求もフックし，学習中に参照されたデータ・されていないデータを記録する機能やディレクトリ重ね合わせ時にアクセスできるファイル・ディレクトリを制御する機能など，機械学習における再現性を保証するバージョン管理機能を実現するための機能群を追加している点が挙げられる．

3.3 学習プログラムと実行時データの独立管理機能

学習プログラムに対するバージョン管理ツリーと実行時データに対するバージョン管理ツリーの二つのツリーを持ち，それぞれ独立に管理することで，互いに異なる時点で得られた学習プログラムと実行時データを組み合わせた学習の実行を可能とする (図 1)．このとき，組み合わせてチェックアウトする操作も Unionfs+ のディレクトリ重ね合わせ機能を使用するためコピー処理は生じず，ストレージを効率よく利用できる．

3.4 学習プログラムの隔離実行機能

学習プログラムは本システムを経由することで隔離実行することができる (図 2)．内部的には，各プロセスごとに異なるマウント名前空間を用意し，その中で本システムの Unionfs+ をマウントすることで，学習データなど共通の入力データをプロセス間で共有しながら互いにファイル出力が干渉し合わないような実行環境を実現している．

3.5 実行時データの自動記録機能

Unionfs+ により，過去に記録された実行時データには Read-only でアクセスできるようにし，学習プログラムから新たな書き込み要求がきた際には Copy-on-Write 方式で新たな領域に書き込むことで，入力データや過去のデータが改変されることなくリアルタイムな実行時データの自動記録を実現できる（図 2)．このとき変更が生じなかったデータは複製されないため，ストレージ使用率を最小限に抑えることができる．また前述の隔離実行機能と連携することで，どの実行コマンドによってどの実行時データが入出力されたのかということも自動的に記録できる．

4 既存のシステムとの比較

DVC [1] はソースコードバージョン管理システム Git [2] と併用することを前提に作られた機械学習向けのバージョン管理システムである．主に学習のワークフロー化を支援する機能を備えているが，本提案システムのような自動記録機能および隔離実行機能を持たないため，どのようなデータが入出力されるかを事前に指定する必要があったり，学習プロセス同士が干渉し合わないようユーザが注意しておく必要がある．

MLflow [4] は機械学習の開発サイクル全体を支援するプラットフォームであり，とくに学習に用いたデータやハイパーパラメータ，学習済みモデルを管理するための機能を備えている．しかし MLflow をプロジェクトに導入するためには学習プログラムやデータ前処理プログラムなどに MLflow API に対応するよう変更を加える必要があり，透過性・再利用性が低く導入コストやヒューマンエラーリスクが依然として残っている．一方，本提案システムでは Unionfs+ が通常のファイルシステムと全く同じように振る舞うため，そのような変更無しに導入できる．

5 結論

本稿では，機械学習ソフトウェアの開発における 3 つの要件 (1) 独立管理 (2) 隔離実行 (3) 自動記録について議論し，それを満たすバージョン管理システムを提案した．とくにファイルシステム Unionfs+ を実装し，学習用データや前処理済みデータ，モデルの中間保存ファイルへの読み書きなど，学習に関わる全てのデータ読み書きの自動記録，および学習プロセスのファイルシステムレベルでの隔離実行機能を，空間効率を損なうことなく実現した．

また Unionfs+ の透過的な振る舞いにより，本システムは学習プログラム側に一切の変更を加えることなく導入できる．本システムを導入することで，学習の試行錯誤の各ステップでどのような学習アルゴリズム・学習データ・ハイパーパラメータを使用しどのモデルが出力されたか，などの記録を自動的に行えるようになり，学習プログラムに本来不要なコードを加えることなく学習の再現性を確保することが可能となった．

今後の展望としては，ストレージ使用効率の更なる改良などを行い，本システムの実用化を目指す予定である．

参考文献

[1] “Data Version Control - DVC”, https://dvc.org/
[2] “Git SCM”, https://git-scm.com/
[3] D. P. Quigley, et al. UnionFS: Userand Community-oriented Development of a Unification Filesystem. Proceedings of the 2006 Linux Symposium (July. 2006).
[4] Zaharia, M., et al. Accelerating the machine learning lifecycle with mlflow. Data Engineering, pp. 39, 2018.