AI システム開発における再利用の検討

大木 憲二[†1]

概要：AI を活用するシステム開発の案件が増えてきており，成功事例などの資産の再利用を促進していきたい．しかしながら，AI システム開発は従来のシステム開発に対して開発プロセスや生成される成果物などが大きく異なっており，従来と異なる再利用のあり方が必要になるのではと予想する．本稿では，著者が携わった開発事例に基づいて再利用性の検討を行う．

1. はじめに

近年，様々なシステムに AI が適用されつつあるが，これはディープラーニングの進展により精度が飛躍的に向上しただけでなく，AI，特に機械学習を開発者が手軽に扱えるソフトウェアが普及してきたことが大きい．しかしながら，このようなソフトウェアは汎用目的に作られているため，実際に現場に適用する際には属人性が出てしまいがちであり，解決手法が求められているところである．

著者は社内の AI システム開発案件に携わっており，本番環境での試行を開始したところであるが，今後，本開発で得られた知見や成果物を他へ展開していくことも求められており，再利用についての検討を行っている．本稿では事例の概説と再利用における検討内容について報告する．

2. 事例

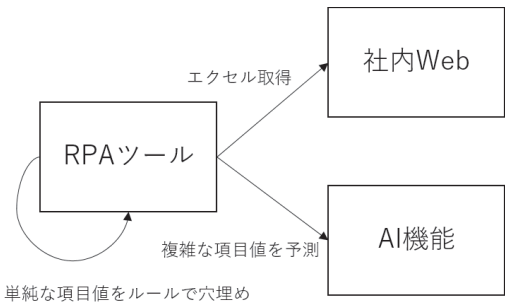

図 1: 本事例の概略図

今回開発した AI は RPA(Robotic Process Automation)システムの一部である．RPA とは Web やデスクトップ操作を自動化するソフトウェアであり，ホワイトカラー業務への適用が進んでいるが，現状では単純な定型作業の自動化に限られている．本事例の自動化対象は Web システムを通じて申請されたデータに含まれている Excel ファイルの中身を埋めるというものであるが，その中の一部項目について埋めるための条件をルールで表す事ができない．そこで過去のデータから学習したモデルに基づいて値を予測する機能を開発し，RPA ツールから呼び出すことで自動化を達成した．手法としては教師あり機械学習の多クラス分類に属するものである．概略を図 1 に示す．

2.1 開発プロセス

本事例の開発プロセスは大きく 4 つに大別される．

1. 企画
2. AI 機能開発・評価
3. システム開発
4. 運用・保守

従来型システム開発との差異は，AI 機能開発を先行して行い，業務適用すべきかどうかの評価を行った点である．AI については評価結果次第で採用を見送る懸念があったが，AI の利用有無によって全体のシステム構成が異なるため，AI の評価を前倒しで行うことでシステム開発への影響を抑えた．

また，早期に運用を開始して効果がある事を示していくために，企画の段階で比較的効果が高そうな業務を絞ったうえで開発を行った．1〜4 のフェーズはサイクルとして捉えることができ，各サイクルにおける運用が安定した後で企画へ立ち返って次のサイクルでさらに自動化対象を拡充していくことにした．

2.2 再利用に対する期待

AI と RPA の開発はそれぞれ異なるスキルが必要であるため，本事例の第一サイクルでは AI と RPA で体制を分けて開発を実施した．一方，次サイクルにおける自動化対象は，扱う入力データ項目は同一ながら違う項目の多クラス分類の自動化を実現するものであった．前のサイクルと開発プロセスや成果物の類似性が予想されることから，すでに得られている資産を流用して，初回より開発工数の短期化が求められていると共に，RPA 側のメンバが中心に双方を開発できるように AI に関わる手順やノウハウの伝達を行う事も求められている．

3. 再利用の検討

3.1 再利用を支援するソフトウェア

再利用の方法を検討する上で不可欠となった 2 つの OSS について紹介する．

(1) **Jupyter Notebook による実行履歴や結果の共有**

†1 (株)富士通研究所
Fujitsu Laboratories Ltd.

Jupyter Notebook[1](以下，Jupyter)とは，ノートブックと呼ばれるエディタ上でプログラムの実行履歴と結果を残しながらインタラクティブに開発作業を行えるオープンソースの Web アプリケーションツールである．グラフの可視化機能も優れており，データの分析や学習，評価を試行錯誤する環境として適している．また，markdown 形式でドキュメントも併記できるため，文芸的プログラミングも模倣する事が出来る．

本開発では，このドキュメント付きで実行履歴や結果を残せるという特性を活かすために，後述する試行錯誤以外のフェーズでも利用した．

(2) **Docker によるサービス実行環境のコンポーネント化**

Docker[2]とは，コンテナ型の仮想化環境を提供する OSS である．Docker は差分ベースでディスク容量を抑えて大量のコンテナを管理できる仕組みや，Dockerfile の記述による環境構築の再現性を確保しやすい仕組みが備わっており，単一サービスやアプリケーションをコンポーネント化するための手段として適している．また，複数のコンテナを組み合わせたシステムを構築することも容易に行える．

本開発では，システム開発や運用で使用するサービスを適宜 Docker のコンテナ化しており，このコンテナを単位とした再利用を促進していくことを意識している．

3.2 本事例への適用

前節で述べたソフトウェアを本事例に適用することでどのように再利用が可能になるかの例を 2 つ提示する．

(1) **AI 開発時の実行環境**

図 2 は AI 開発者がデータの分析を行い，様々な機械学習アルゴリズムを用いてモデルを構築するための環境の構成図である．

本開発では，最初のサイクルで行った分析や学習評価の履歴をノートブックとして残していたため，これをベースに体裁を整えてテンプレートとして Docker イメージの中に含めた．次サイクルにおける入力データの形式は同一であるため，まず開発者はこのテンプレートを手元のデータでそのまま実行し，最初のサイクルでどのように分析や学習評価を行ったかを体感する．そして，今回のサイクルで求められている項目に合うようにテンプレートをカスタマイズしながら，新たな知見を得るために分析を進め，学習評価を行っていくことが出来る．

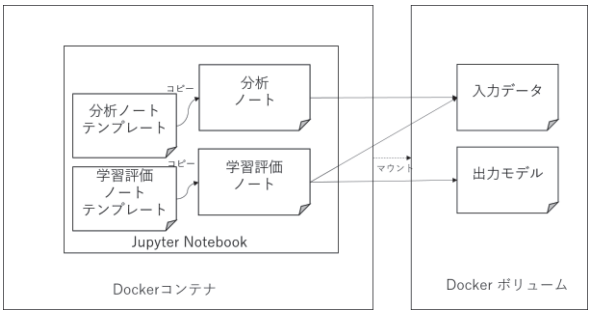

図 2: AI 開発時の実行環境構成

(2) **運用中のモデル再生成環境**

図 3 は運用・保守フェーズでデータが変化して精度が下がってきた際に，新たな学習データを用いてモデルの再構築や配備を行うための環境の構成図である．

Jupyter 側の Docker イメージは AI 開発時に利用したものからノートを入れ替えて別コンテナとして動かしている．モデル構築や配備の処理もノートブック形式で作成し，中身の一部をパラメタ化して，外からパラメタを渡してバッチ形式で実行できるようにしている．従来型のスクリプトを組む方法と比較して，実行履歴をログ代わりに残せるという特性があるため問題発生時の解析などが容易になる．

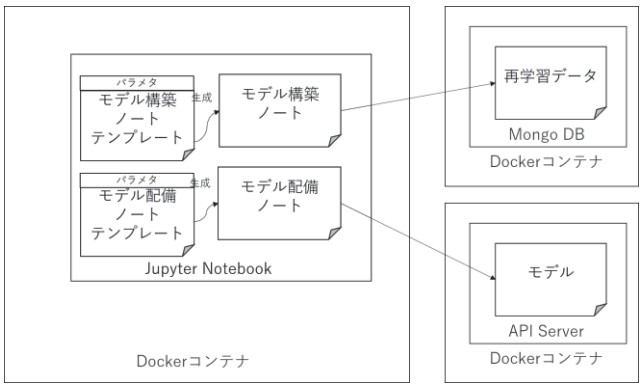

図 3: 運用中のモデル再生成環境構成

3.3 課題

Jupyter や Docker を再利用に積極的に活用していくには以下のような従来の再利用資産と同様の切り出しや粒度に関する検討課題があり，適切に設計しないと再利用による十分な効果が得られない可能性がある．

- コードをノートブックとして再利用可能にする場合における，クラスなど従来型のモジュール化との切り分け
- システムをコンテナ化する場合のコンポーネント粒度の切り出し方法

4. おわりに

本稿では AI システムを事例とした再利用の方式について説明した．尚，本稿では触れていないが，AI 開発一般ではデータや学習モデルの再利用性もまず重視されていることであり[3]，本稿の内容と併せて検討する必要がある．

今回実践した方式は再利用技術として必ずしも新しいわけではないが，OSS の活用によって知見や環境といった従来見逃されやすい側面の再利用が行いやすくなってきたとも考えている．本稿で述べたような方式を AI システムに限らず応用していきたい．

参考文献

[1] “Jupyter Notebook”, https://jupyter.org/
[2] “Docker: Enterprise Container Platform”, https://www.docker.com/
[3] 丸山宏．機械学習工学に向けて．日本ソフトウェア科学会大会論文集. 2017, vol.34, p.297-304.