An Exploratory Study of Deep Learning Supply Chain

In Proceedings of MSR '22: Proceedings of the 44th International Conference on Software EngineeringMay 2022
pp. 86–98

深層学習

- ►深層学習(以下, DL)が様々な分野で主流の方向性
  - コンピュータビジョン，自然言語処理など
  - 多くのDLフレームワークも存在

- ►DLエコシステム
  - 既存のフレームワークを独自で強化して使用
  - 強化したDLパッケージをリリース

- ►DLフレームワークを中心としたサプライチェーンを形成

背景

▶DLサプライチェーンのブラックボックス化

- サプライチェーンに関わるプロジェクトの多さ
- 複雑な依存関係
- 効果的なモデリング手法の欠如

▶DLサプライチェーンの問題点

- フレームワークの移り変わりが早い
  → 新たなフレームワークのエコシステムの構築をどうするか

⬇

成功したサプライチェーンの例から,
新たなフレームワークの開発の参考への知見を提供

RQ

- ►RQ 1（構造）
  - DLサプライチェーンの構造はどのようなもので,
    どのように進化し，その構造の脆弱性をどのように特定するか

- ►RQ2（領域分布）
  - DLサプライチェーンはどのような領域をカバーしているのか
    また，２つのサプライチェーンでの分布は異なるか

- ►RQ3（進化的要因）
  - DLサプライチェーンの下流プロジェクトにはどのような要因が関係しているか

データ収集

- ►WoC
  - 多くのGitプロジェクトを収集，依存関係を取得可能
  - 依存関係はすでに特定のファイルに格納済み
  - 直近の2ヶ月のデータは削除
    → データ量が少ないため

- ►Libraries.ioデータセット
  - 270万以上のパッケージ，3300のプロジェクト
  - 依存関係を追跡可能

- ►GitHub API
  - READMEファイルやプロジェクト情報を取得

サプライチェーンの構築

▶DLサプライチェーンの構築

- プロジェクトがノード，パッケージの参照関係がエッジ

Algorithm 1: Constructing DL SC

```
Input: R_i ;                    // url of projects on the i-th layer
Output: DL SC
initialization: i ← 1 ;                 // start from the 1st layer
IN_i ← {} ;     // import names of the packages released by the
  projects on the i-th layer
R_i ← {url of TensorFlow or PyTorch};
while true do
    IN_i ← {} ;
    for r ∈ R_i do
        search Libraries.io to obtain the released package p;
        if p then
            manually label the import name of p: i_n_p;
            IN_i.insert(i_n_p);
    if IN_i is null then
        return ;                        // no packages are released
    else
        i ← i + 1, R_i ← {} ;                          // next layer
        for i_n_p ∈ IN_i do
            search WoC to obtain projects that import i_n_p: r;
            R_i.insert(r);
    if R_i is null then
        return ;                    // no downstream projects exist
```

DLフレームワークが起点

↓

Libraries.ioを検索して関連パッケージを検索

↓

手作業でインポート名をラベル付

↓

各インポート名についてWoCを検索

RQ 1 (DLサプライチェーンの構造はどのようになっているか)

►TensorFlowのサプライチェーンを構築

- ２つのサプライチェーンの構造とその進化の特徴を示す

►結果

- サプライチェーンが際限なく続くわけではない (6次利用まで)
- パッケージを公開しているプロジェクトが少ない (利用のみ)

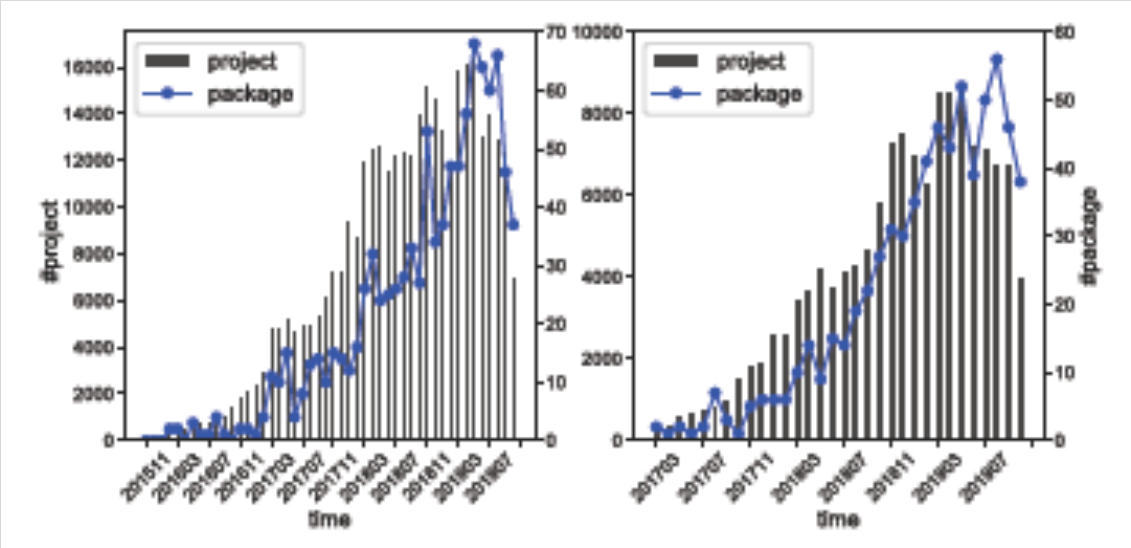
	TensorFlow	PyTorch
Start Time	2015.11	2017.03
End Time	2019.10	2019.10
#Projects (nodes)	355,392	136,507
#Packages	1,022	697
#Import relationship (edges)	369,201	152,059
#Layers	6	5
#Projects with downstream	614	418

RQ 1 (DLサプライチェーンの構造はどのようになっているか)

▶結果

- ほとんどのプロジェクトがTensorFlow, PyTorchを直接利用
  → 下流のプロジェクトを探し出すのが難しい可能性
- すぐ上のプロジェクトのパッケージのみをインポートする傾向
- プロジェクトへの追加数は日ごとにどちらも増加傾向

TensorFLow			PyTorch		
Project Layer	Layers of Dependent Projects	Count	Project Layer	Layers of Dependent Projects	Count
1	/	1	1	/	1
2	1	317,115	2	1	112,516
3	2	23,323	3	2	8,072
	1,2	12,252		1,2	15,255
4	3	1,268	4	3	422
	1,3	500		1,3	28
	2,3	348		2,3	104
	1,2,3	321		1,2,3	81
5	4	211	5	4	24
	1,4	3		1,4	1
	2,4	3		3,4	3
	3,4	34			
	1,2,4	6			
	1,3,4	1			
	2,3,4	1			
	1,2,3,4	4			
6	5	1			



RQ2 (サプライチェーンはどんな領域をカバーしているか)

▶トピックモデリング分析

どのような領域で使用されているのかのラベル付
README.mdを基に行う

⬇

- サプライチェーンの進化状況の把握に役立つ

▶対象

- 10以上の下流プロジェクトをもつパッケージを分析
- → TensorFlow : 215パッケージ， PyTorch : 142パッケージ
- ラベル付は手間がかかるため

RQ2 (サプライチェーンはどんな領域をカバーしているか)

- ▶サプライチェーンにおけるパッケージ分布
  - 合計51種類のパッケージが存在
  - 領域関連パッケージは様々な分野へ応用
  - 利用のみではなく,
    サービスやサポートへの拡張も多い

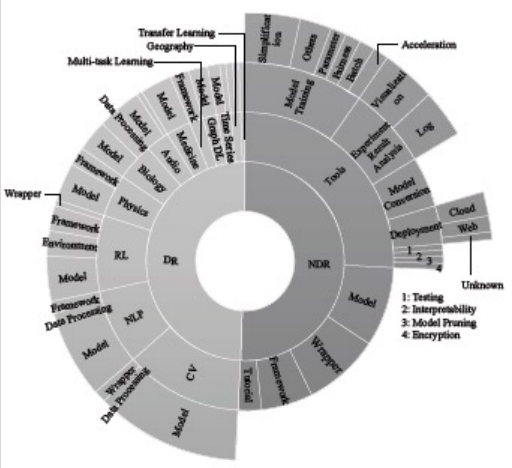


Figure 7: Packages Distribution of TensorFlow SC.

- ▶人気があるパッケージ
  - 非領域パッケージが人気
  - RL (強化学習) の人気が高い

Table 3: Top 10 Most Popular Package Types

	TensorFlow SC		PyTorch SC	
Rank	Type	Count	Type	Count
1	NDR, Tools	18,372	NDR, Tools	14,080
2	DR, RL	3,902	NDR, Framework	2,930
3	DR, CV	3,138	DR, CV	2,412
4	DR, NLP	1,437	DR, NLP	821
5	NDR, Wrapper	1,159	DR, Audio	782
6	NDR, Model	829	DR, RL	376
7	NDR, Framework	693	DR, Medicine	336
8	DR, Biology	619	DR, Physics	315
9	DR, Physics	563	NDR, Model	261
10	DR, Medicine	353	DR, Graph DL	242

RQ2 (サプライチェーンはどんな領域をカバーしているか)

▶サプライチェーンにおけるプロジェクト

- CV, NLP, 自動運転やロボットなど様々な分野のプロジェクト

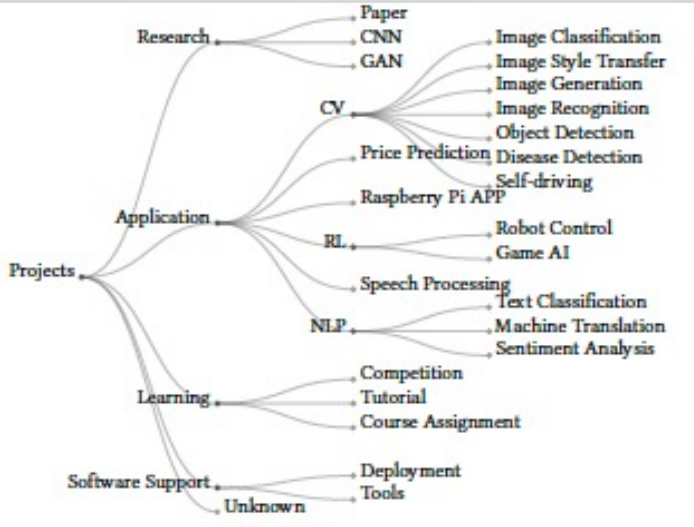


Figure 9: Project Types in the TensorFlow SC

- 2次利用のパッケージについても同様の下流プロジェクトをもつ

RQ3 (サプライチェーンの進化にはどのような要因が関係するか)

▶下流プロジェクトが10個以上あるパッケージを分析

- 下流プロジェクトの数に関連する可能性のある３つのカテゴリ
  下流プロジェクトの特徴 : コミット数, 貢献者数, 年齢, スター数
  領域 : 領域に関連しているか
  サプライチェーン特徴 : レイヤ数，依存パッケージ数

- GAM (一般化加法モデル)を適用
  説明変数と目的変数の間の非線形・非単調関係を解析するため

- 最終的なモデルは
  領域，著者数，依存パッケージ数の３つの説明変数を持つ

RQ3 (サプライチェーンの進化にはどのような要因が関係するか)

►結果

- 下流プロジェクトへの影響は非線形
  → 著者が少なければ負の相関
  → 著者が多ければ (7人以上)の時は正の相関

- 依存パッケージ数が多い
- → 下流プロジェクトは少ない傾向

まとめ

- ▶DLコミュニティの発展を支援するための知見を提供

- ▶RQ 1
  - サプライチェーンは階層構造
  - 構造を可視化することでサプライチェーンの状態を把握しやすくなる

- ▶RQ2
  - 領域分布を分析
  - パッケージを利用する際の参考になる

- ▶RQ3
  - 下流プロジェクトの数に関連するいくつかの要因を明らかに
  - 著者数や依存パッケージ数が下流プロジェクトに影響する

所感

- ▶言い回しをいちいち変えていることが多く，読んでいてしんどかった

- ▶分析手法が難しい！
  - 特にRQ3はほとんど理解できない