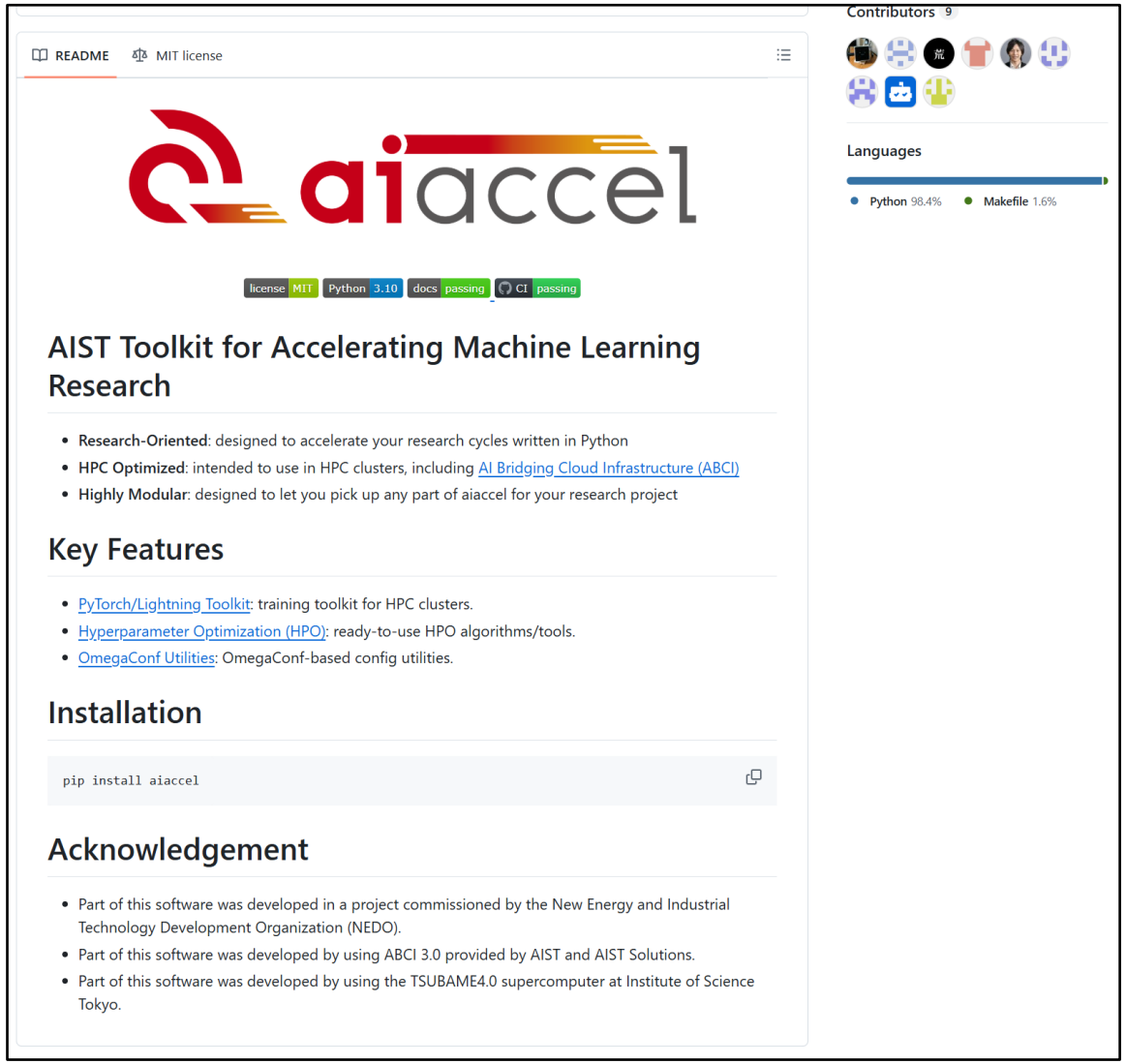
人工知能を実社会で利用するための基盤技術の紹介

～フィジカル領域の生成AI基盤モデル～

産業技術総合研究所 人工知能研究センター
社会知能研究チーム
大西 正輝

AIST Toolkit for Accelerating ML Research

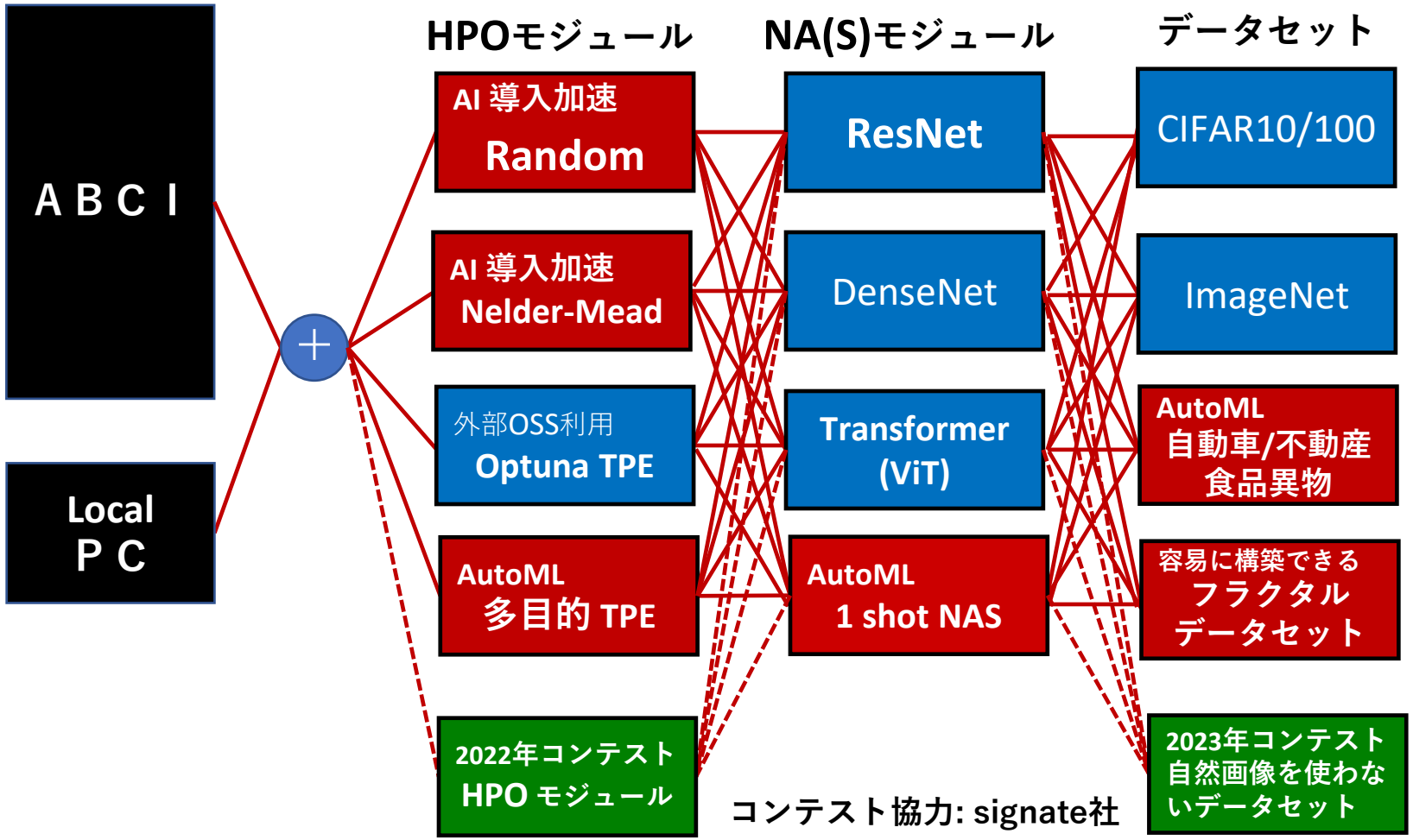


ML や AI の研究開発を
加速するための基盤技術として
Github で公開

https://github.com/aistairc/aiaccel

aiaccel 公開当時の構想

- NEDOプロ（AI導入加速・AutoML）で人工知能の現場導入時間を10%以下に
  2018～2022　2020～2023



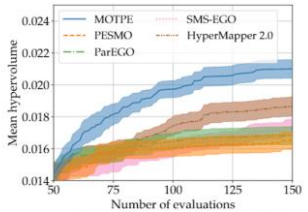
コンテストを利用したモジュール作成

2022年度に開催

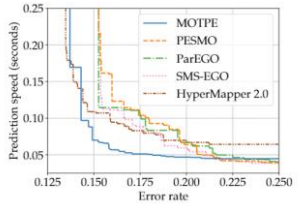
HPOモジュールコンテスト

◎ HPOの多目的拡張

HPOの一手法TPEである多目的最適化に拡張し CNN のモデル作成の多目的最適化で評価



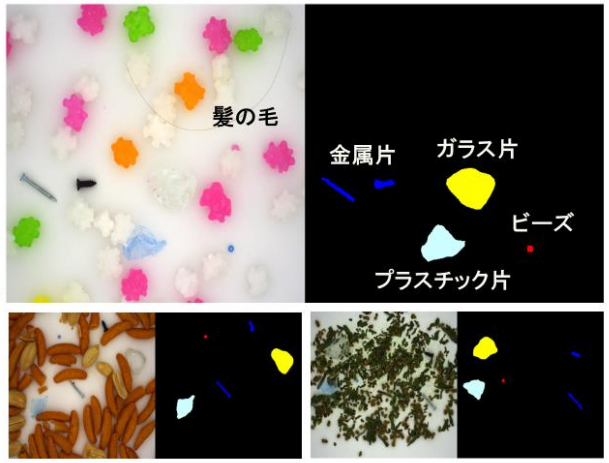
ハイパボリュームの推移



得られたパレート解集合

◎ コンテストの手伝い（データ利活用）

◎ 実データでの評価（異物検知）



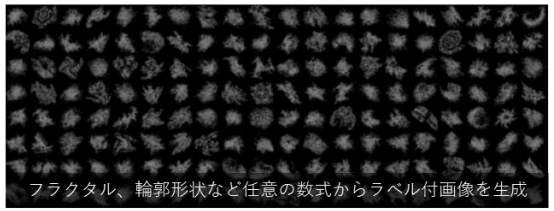
HPOやNASで検出の困難な髪の毛を検出したい

2023年度に開催

自然画像を用いない事前学習用データセット作成モジュールの開発

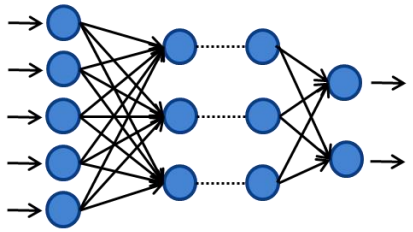
事前学習用データセット生成モジュール

作成



フラクタル、輪郭形状など任意の数式からラベル付画像を生成

事前学習



Fine-tuning

他データセット１

識別率〇%

他データセット２

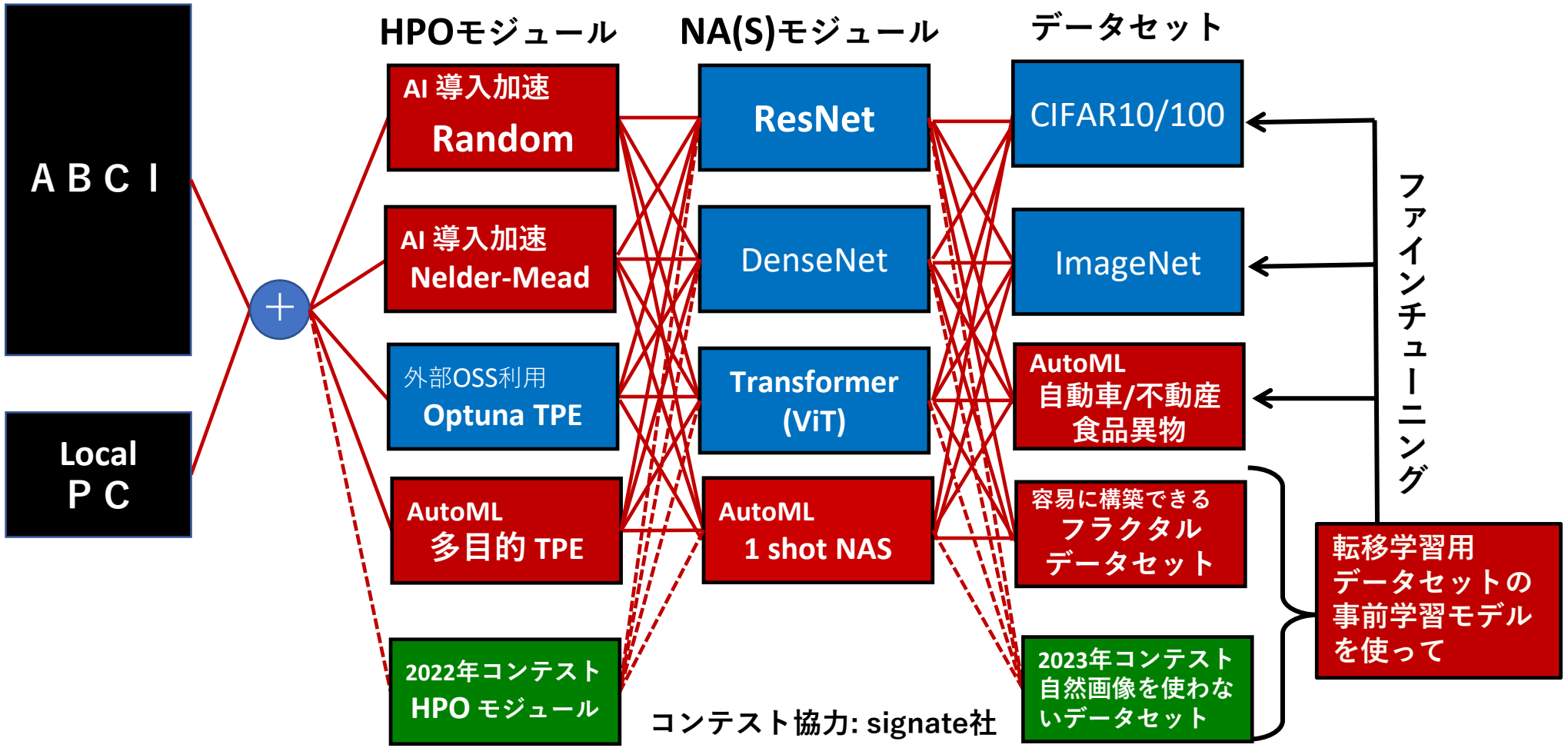
識別率〇%

自然画像を用いない事前学習用データセットを作成するためのモジュールを作成し転移精度を競う

aiaccel 公開当時の構想

2018～2022　2020～2023

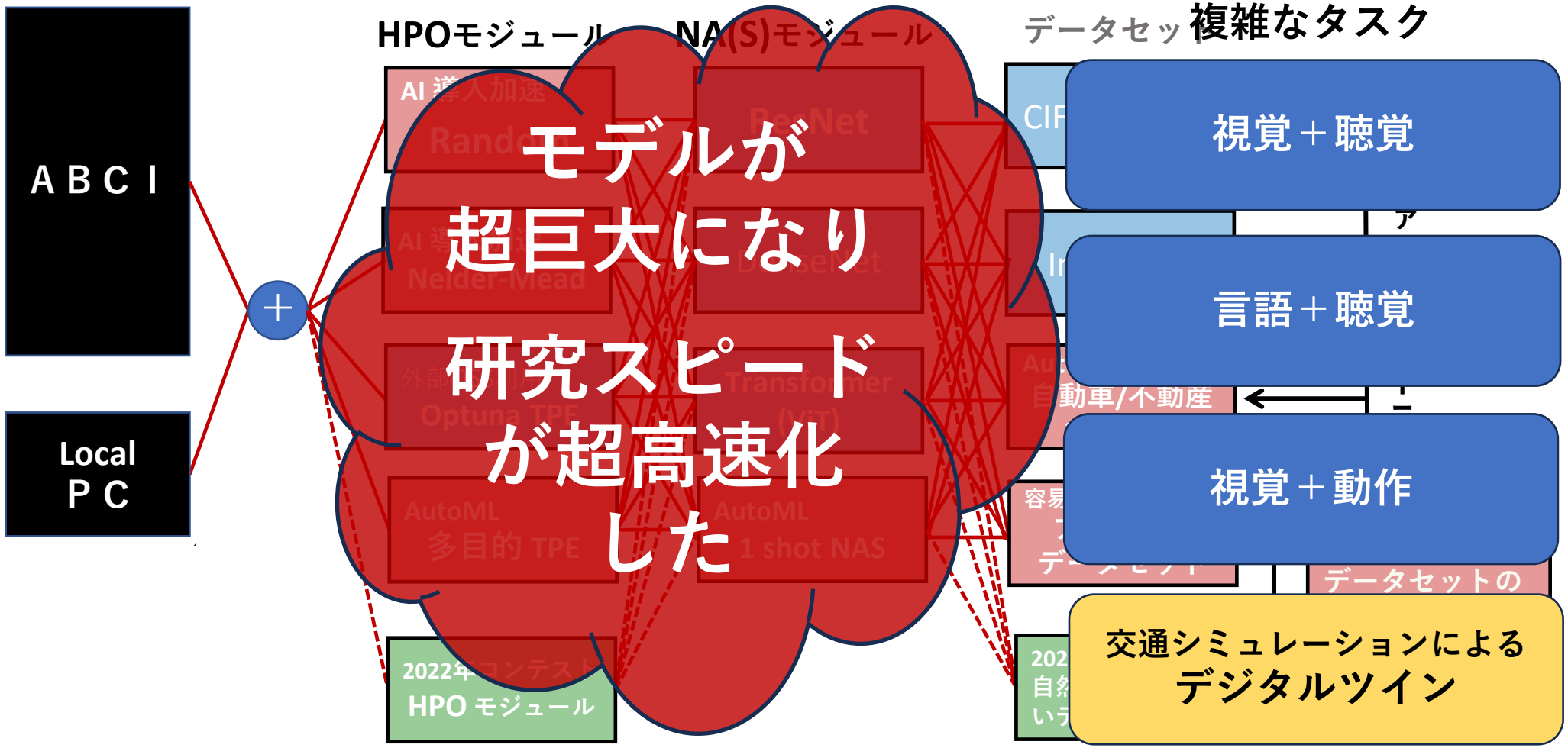
- NEDOプロ（AI導入加速・AutoML）で人工知能の現場導入時間を10%以下に



aiaccel 現在の構想

2018～2022　2020～2023

- NEDOプロ（AI導入加速・AutoML）で人工知能の現場導入時間を10%以下に



人工知能開発の最近の傾向

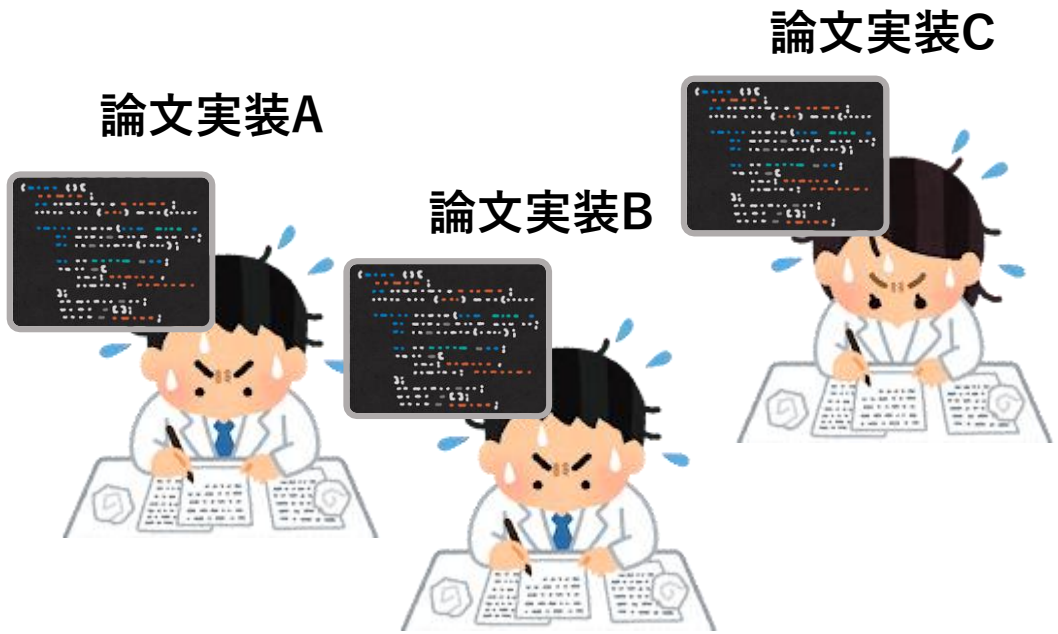
- 課題① 開発スピードが超高速化
  - 当初誰でも簡単にモデル開発できるようにを掲げていたが…
  - 研究者でさえ多くの試行錯誤の中から何が正解か分からなくなってきている
  - 公開コードやデータにさえ誤りが多くある（論文とソースコードの乖離）
  - 学習・評価パイプラインの統一化が必要

- 課題② モデルが超巨大化
  - モデルが超巨大になって学習するのが手間
  - 自分達は超巨大データを持ってないので一から学習することができない
  - 高速な評価によるハイパパラメータ最適化技術が必要

AIモデル開発の課題①：開発スピードが超高速化

論文再現のための実装の多くは，学習・評価パイプラインを各自が個別で実装

- 多くは「論文執筆・実装公開のための書き捨てパイプライン」→ 保守性に課題
- 人の入れ替わりが頻繁な研究現場では，技術継承・引き継ぎのため標準化が不可欠



後続研究

- どの実装も拡張性が低いし統一されていない…
- 自分の拡張を導入するにはハードコードが多すぎる…

aiaccelをパッケージ化して開発

AIST
Create the Future, Collaborate Together

第三極の開発ではなく標準フレームワークの利便性を改善するヘルパーライブラリ

- 各パッケージは独立して実装 → 「使いたい部分を使いたいだけ」切り出せる設計

aiaccel.torch

HPC環境下での分散学習

- PyTorch Lightningのユーティリティ群
- ボイラープレートなコードの排除
- HDF5による統一的なデータセット管理
- データセットのキャッシュ機構を提供

aiaccel.hpo

ハイパーパラメータ探索の自動化

- OptunaのCLIラッパーおよびオプティマイザの提供
- 局所探索に優れるNelder-Meadオプティマイザの提供
- aiaccel.jobと併用しやすいCLIの提供
- 各モジュールはOptuna / Pythonコードから単体利用可能

aiaccel.config

効率的な設定ファイル管理

- OmegaConf に継承などを追加整備
- `hydra.utils.instantiate`による YAMLメタプログラミン

aiaccel.job

統一的な計算機環境の抽象化

- SGE・PBS・Slurmをサポート
- Kaldi/ESPnet-likeなCLI設計
- MPI並列計算をサポート

aiaccel.workflow

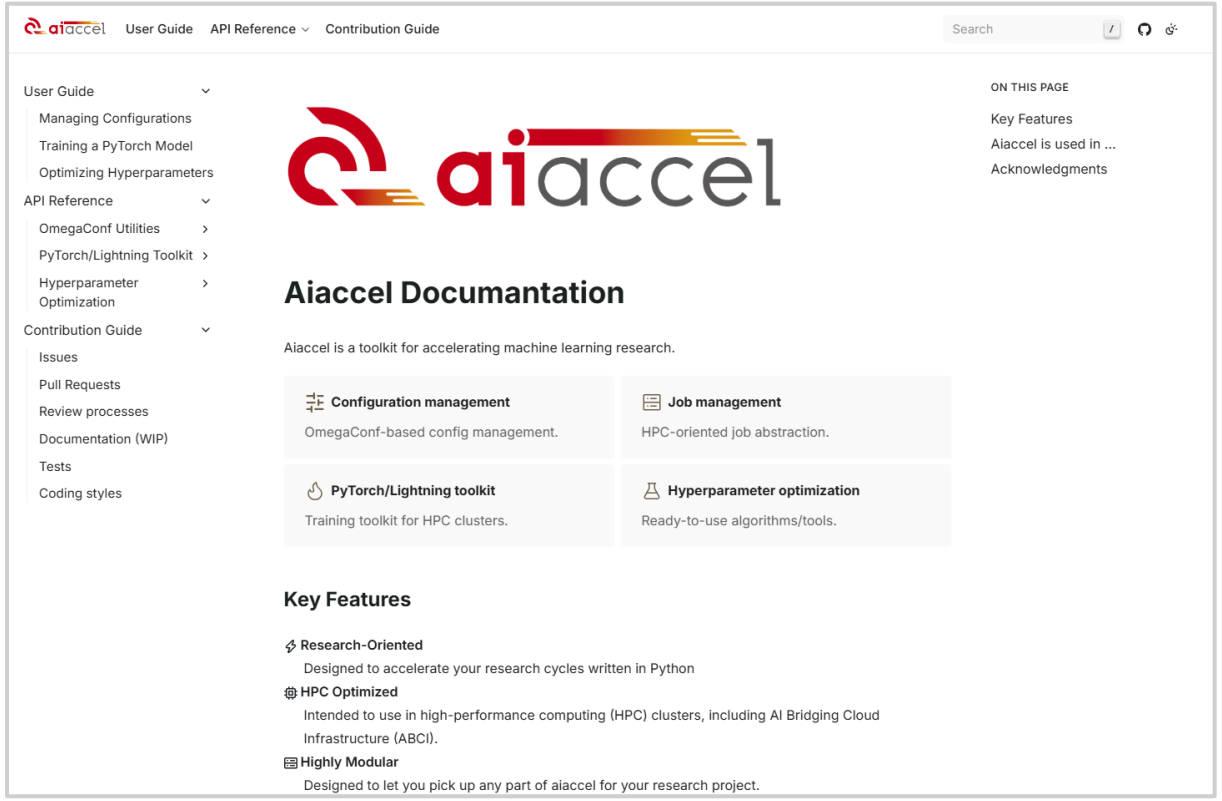
再現性を担保するワークフロー管理

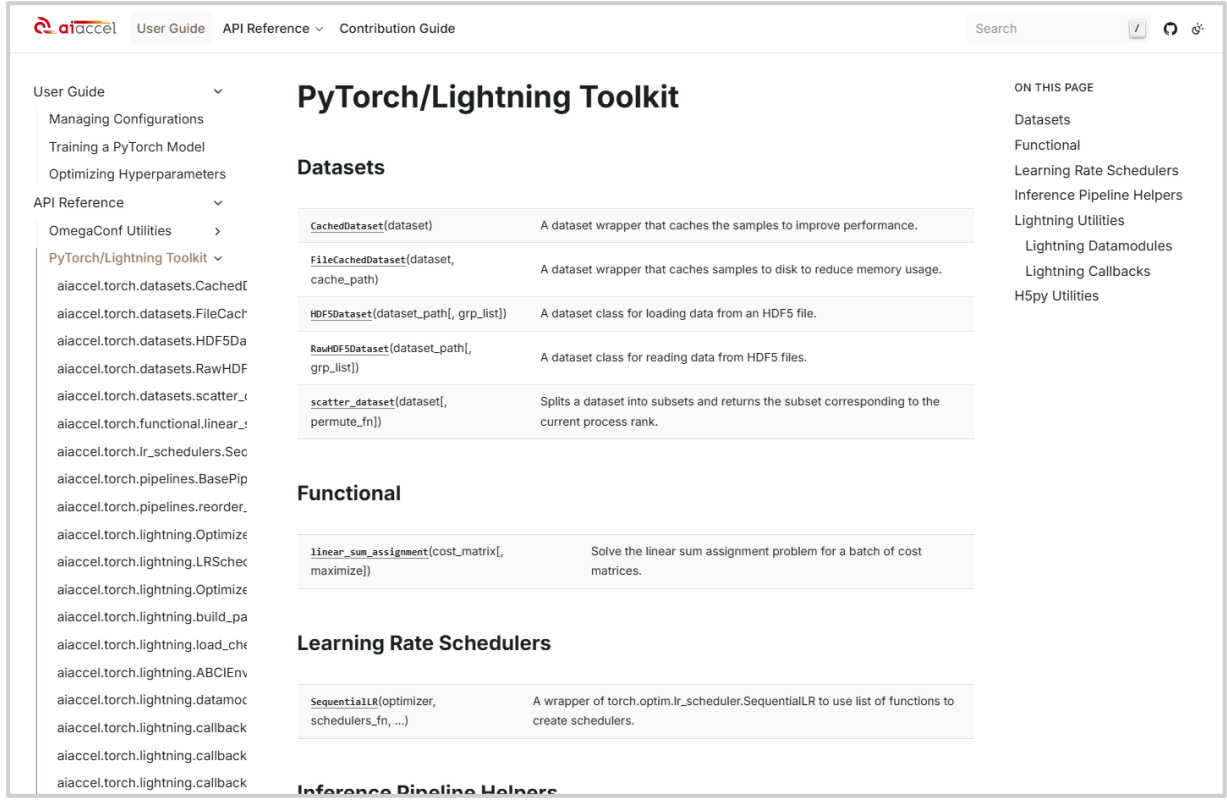
- GNU Makeベースのフロー管理
- ジョブテンプレートを整備
- aiaccel.jobとの併用で並列処理

容易に利用開始するためのドキュメントを整備

API・利用方法および設計思想のドキュメントを整備しWeb上で公開

- 実装ノウハウをドキュメント化し，機械学習パイプラインの開発コストを低減





※ User Guideは現在整備中

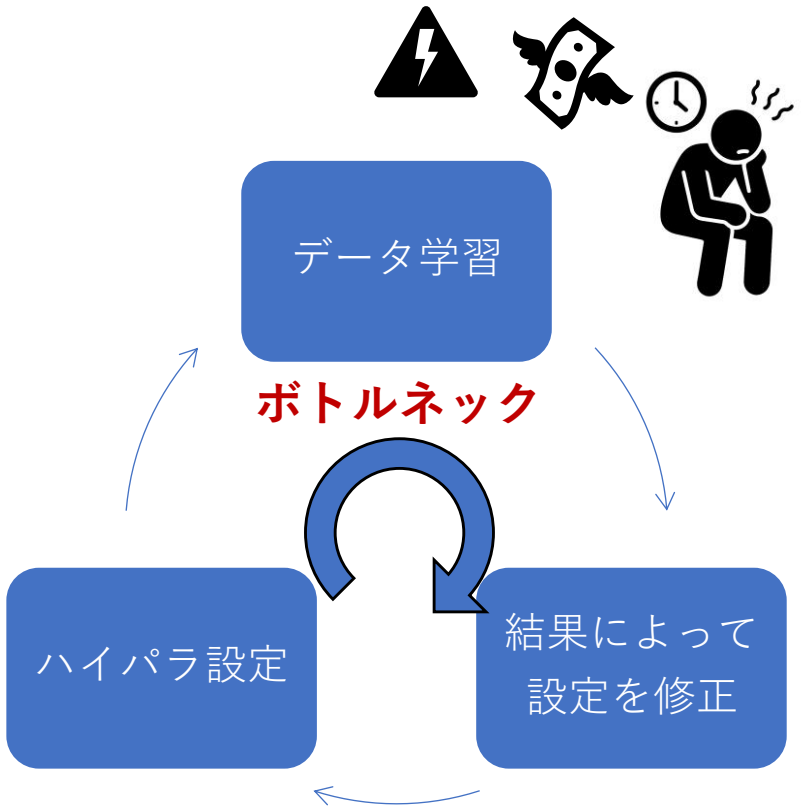
AIモデル開発の課題②：モデルが超巨大化

- 生成モデルにおける HPO は事前調整（キャリブレーション）に手間と時間がかかる、なぜか？

→計算が演繹的※だから

※決められたルールに基づき一方通行で計算を実行すること

- HPO→データ学習→HPO→学習→…
  - トライ＆エラーによる最適化はループ構造
  - 計算が時間的・費用的・エネルギー的に高コスト
- 最適な設定を見つけるまでの繰り返しが多く、プロセスのボトルネックとなる



このループが演繹的計算最適化の特徴

抜本的な解決法は？

- 人間ならどうする？ → 成功例に倣う
  - 過去の成功事例を集める
  - 似た事例であれば最適な設定も似ているはず！

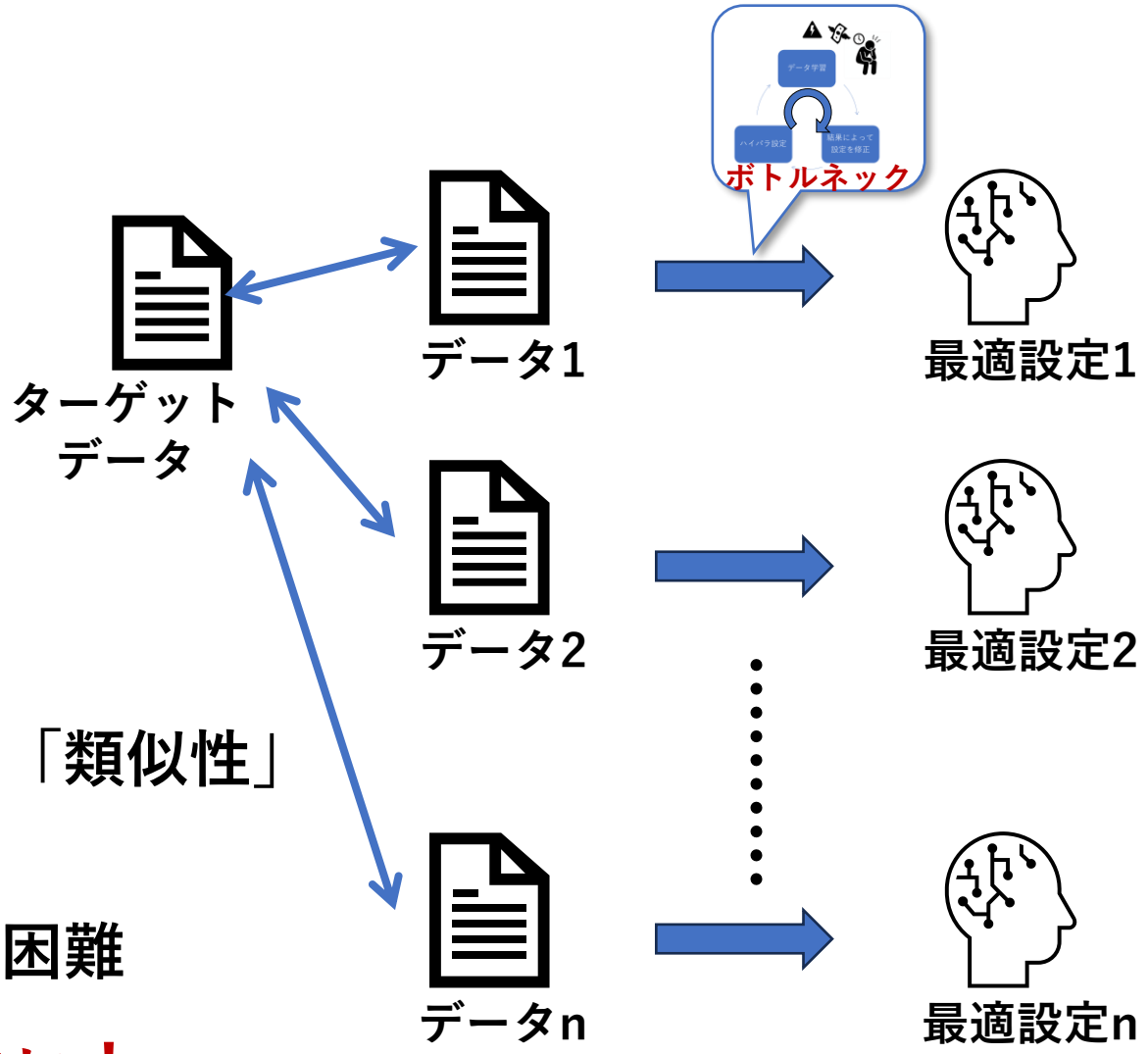
→テーブルルックアップ方式

※事例が離散的で少量であればこれで対処可能

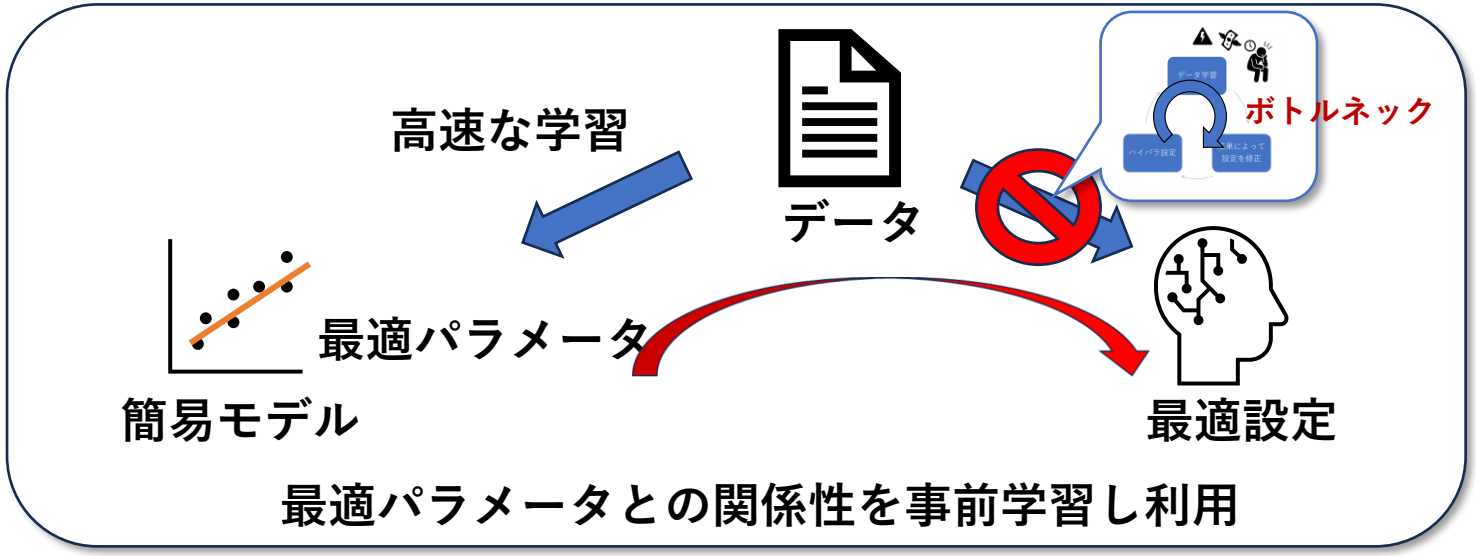
しかしながら・・・

- 各データサイズが大きいとき「似た事例」＝「類似性」の計算が困難
- 過去事例が膨大な時、ターゲットとの比較が困難

→データをコンパクトに定量化し比較すればよい！



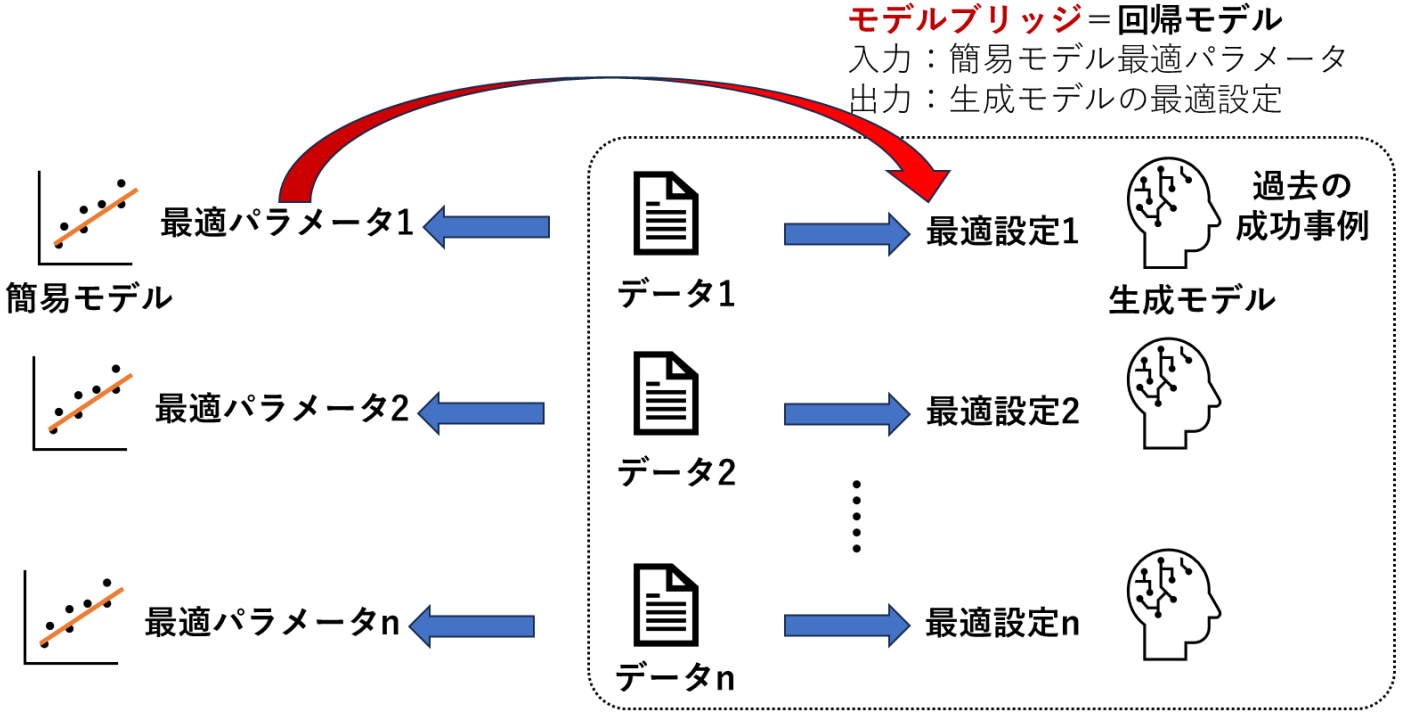
モデルブリッジによるHPOの高速化



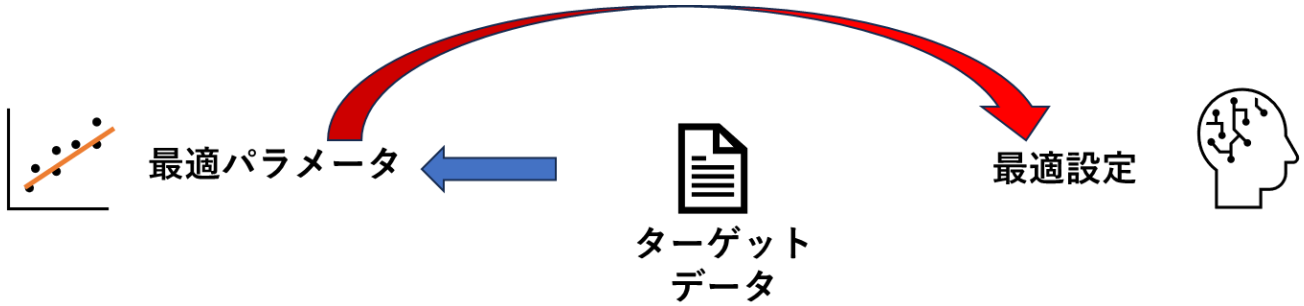
- 簡易な機械学習モデルでデータを学習：高速な学習が可能
  - 簡易モデルのデータ再現性が目的ではなく、最適パラメータとしてデータを定量化することが目的
- その「最適パラメータ」と元の生成モデルの「最適設定」の間の関係性を学習
- ２つのモデルの関係性を繋ぐモデルブリッジを過去事例から構築
- 「ターゲットデータ」から生成モデルの「最適設定」へのボトルネック計算を含むループを迂回し高速化を実現

モデルブリッジの構成

- 事前学習フェーズ：過去の事例（データと最適設定）を活用

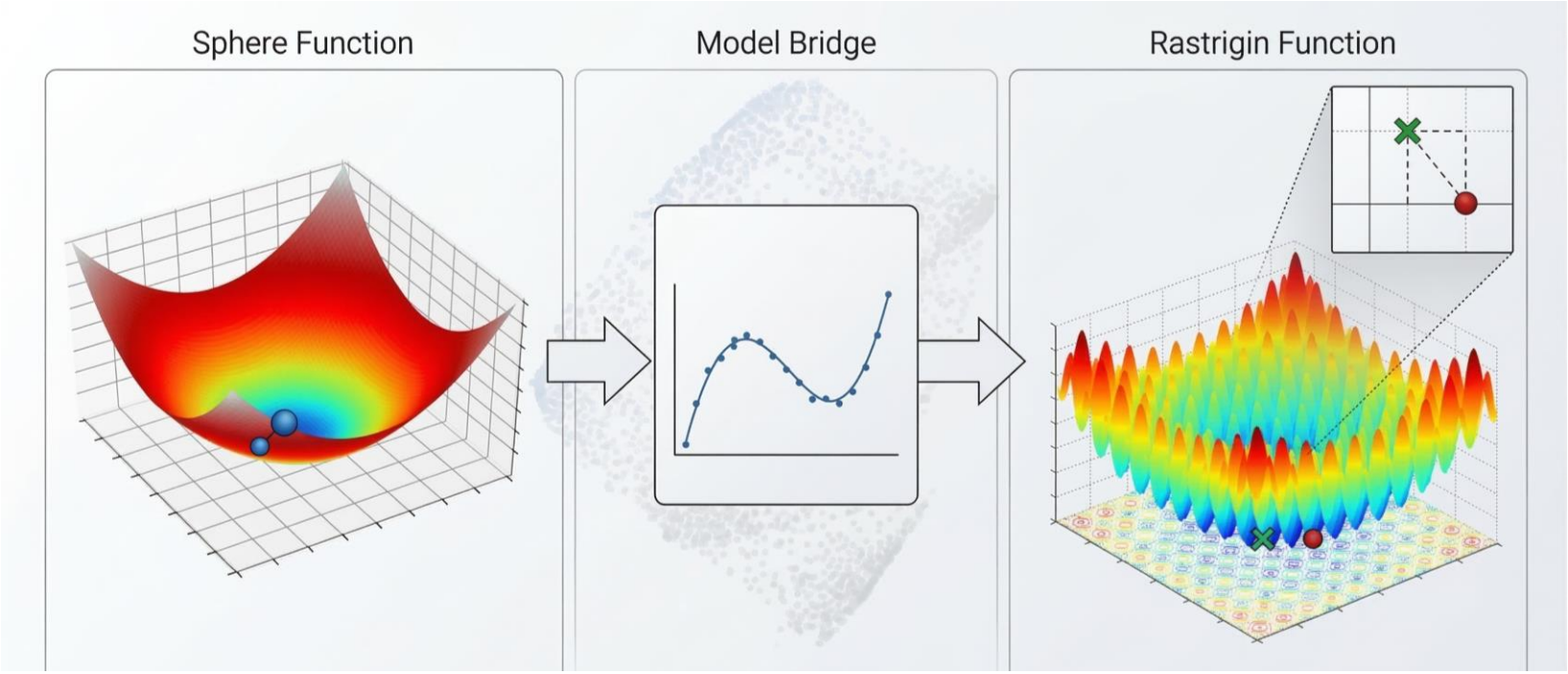


- 実行フェーズ：簡易モデルの学習結果から生成モデルの最適設定を直接推定



aiaccel に実装したモデルブリッジの例

- ベンチマーク関数間のモデルブリッジ
  - ベンチマーク関数からベンチマーク関数へのモデルブリッジを行うシンプルな例
  - examples/hpo/modelbridge/



aiaccelのモデルブリッジ機能

- aiaccel feature/modelbridgeブランチにて公開
- ４つのスタンドアロン機能: prepare, collect, fit_model, evaluate

1. prepare

モデルブリッジのディレクトリ構成にHPO実行に向けた設定ファイルを展開

2. aiaccel-hpo

深層学習モデルのHPOを実行。事前に実施済みの場合はスキップ可

3. collect

深層学習モデルのHPOの結果を収集

4. fit model

収集データに基づいて簡易モデルの結果を近似するサロゲートモデルを学習

5. evaluate

サロゲートモデルによって高速にパラメータを推定し、モデルのHPO結果を評価

実行例

1. git clone

```
❯ git clone -b feature/modelbridge https://github.com/aistairc/aiaccel.git
Cloning into 'aiaccel'...
remote: Enumerating objects: 8922, done.
remote: Counting objects: 100% (951/951), done.
remote: Compressing objects: 100% (621/621), done.
remote: Total 8922 (delta 608), reused 407 (delta 330), pack-reused 7971 (from 3)
Receiving objects: 100% (8922/8922), 5.17 MiB | 15.58 MiB/s, done.
Resolving deltas: 100% (5674/5674), done.
```

```
❯ cd aiaccel
```

2. pip install

```
❯ pip install ".[dev,modelbridge]"
Processing /Users/aistairc/Downloads/work/aiaccel
  Installing build dependencies ... done
  Getting requirements to build wheel ... done
  Preparing metadata (pyproject.toml) ... done
```

```
❯ cd examples/hpo/modelbridge
```

3. make all

```
❯ make all
========================================
  [Stage 01] Prepare
========================================
mkdir -p "/Users/aistairc/Downloads/work/aiaccel/examples/hpo/modelbridge/workspace/commands"
"/Users/aistairc/Downloads/work/aiaccel/examples/hpo/modelbridge/workspace/logs”
...
```

実行結果

```
❯ tree workspace
workspace
├── commands
├── logs
├── models
│   ├── model_meta.json
│   ├── regression_model.pkl
│   └── summary.json
├── pairs
│   ├── test_pairs.csv
│   ├── test_predictions.csv
│   └── train_pairs.csv
├── runs
│   ├── test
│   │   ├── macro
│   │   │   ├── 000
│   │   │   │   ├── config.yaml
│   │   │   │   ├── merged_config.yaml
│   │   │   │   └── optuna.db
```

← regression model

← evaluation result

← collected HPO results

← HPO runs

```
❯ cat workspace/models/summary.json
{
  "metrics": {
    "mse": 0.12345678901234567,
    "mae": 0.12345678901234567,
    "r2": 98.765432109876543
},
"n_test_samples": 20,
"model_type": "LinearRegression"
}%
```

応用例１｜視聴覚音響イベント物体検出基盤モデル

音を発する物体の位置とその種別を推定する大規模マルチモーダルAIモデルを構築

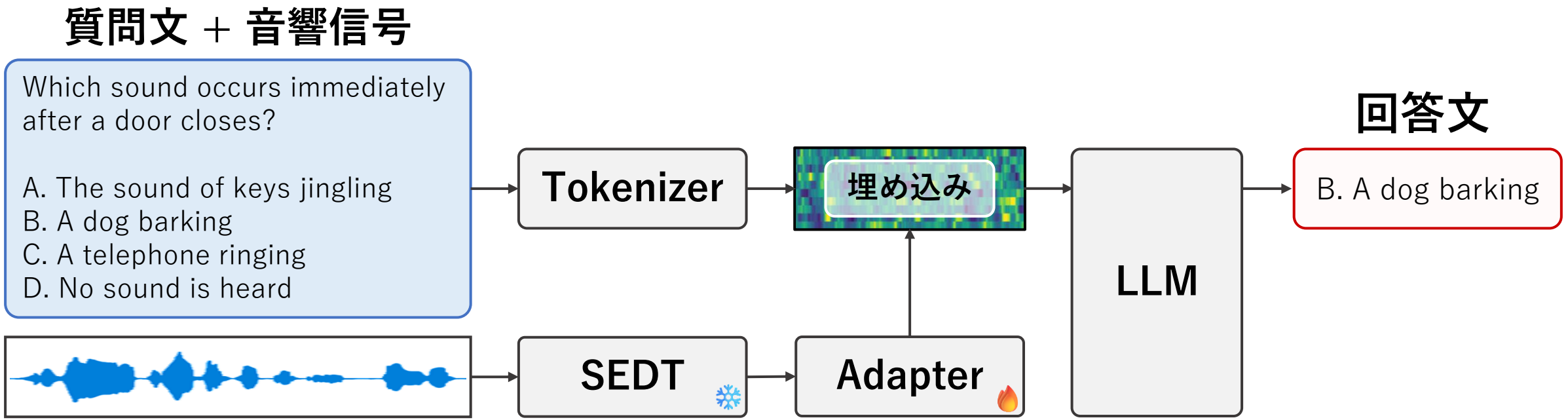
- 半自動アノテーションにより従来の80倍以上となる6.1Mフレームにラベル付与
- 膨大な半自動アノテーション・分散学習を aiaccel により可搬性高く実装

S. Sakurai+ “General-Purpose Audio-Visual Sounding Object Localization Based on Semi-Automatic Annotation,” IEEE SP Letters, submitted.

応用例２｜環境音分析のための音響言語モデルの開発

環境音に関する質問に答えるLLM（AQA）を音響イベント検出モデルを用いて構築

- 従来のフレーム単位の特徴でなくイベント単位の特徴で正答率を65.2 % → 74.4 %
- PyTorch Lightningに基づく aiaccel により，HPC環境下で容易に実装

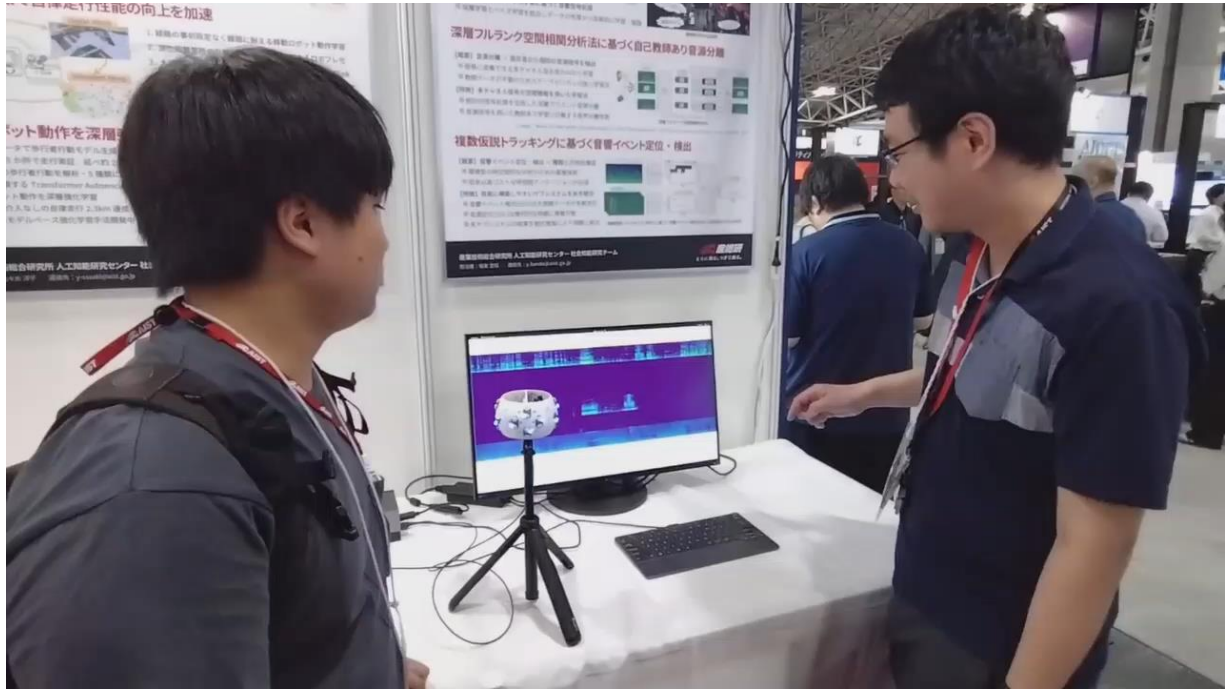


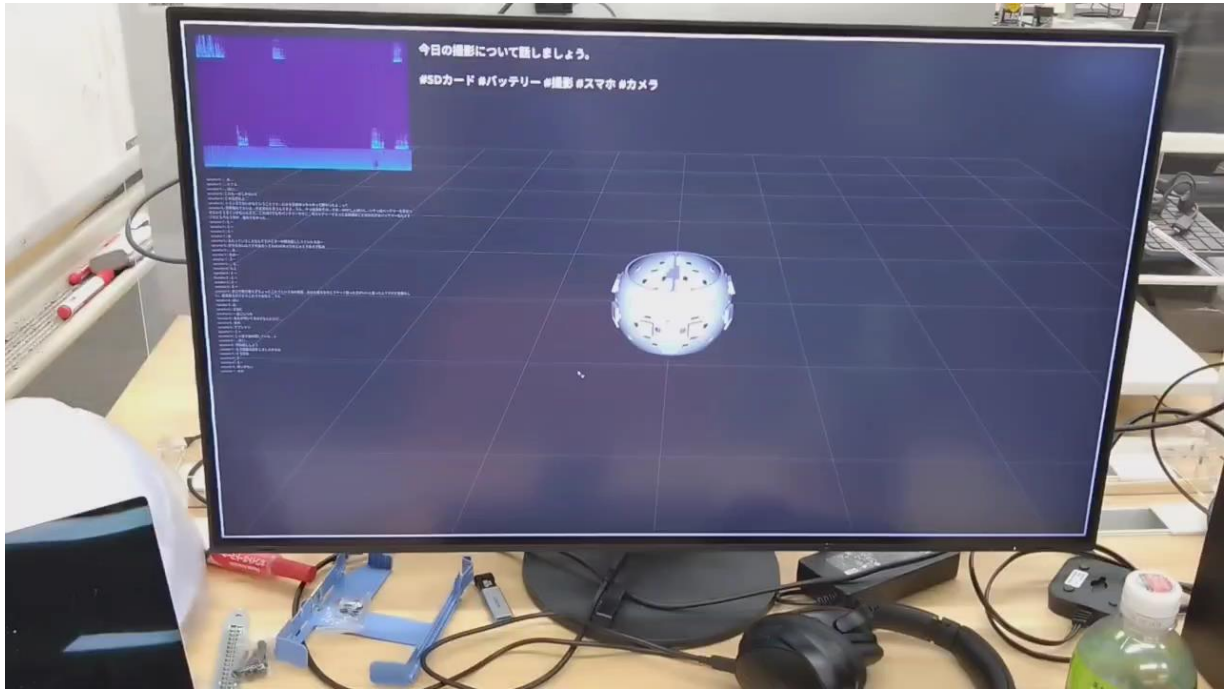
朴 潤花ほか, “イベント中心のシーン理解のための音響イベント検出 Transformerの再考,” 日本音響学会 春季研究発表会 2026

実用例３｜実時間音声対話分析システムの構築

詳細な教師情報（音源信号）を用いず学習できる遠隔音声認識フロントエンドを開発

- アレイ信号処理と深層学習を効果的に統合することで複雑環境でも頑健に動作
- 実アプリでの推論を前提とした aiaccel を活用することで容易にROS2と統合



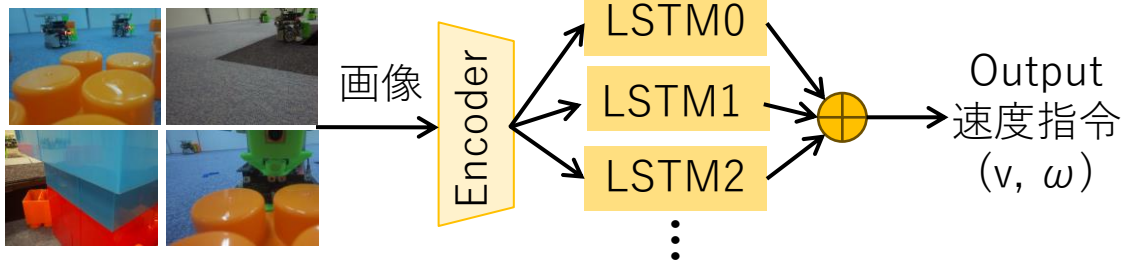


※ 本システムの一部は内閣府 BRIDGEに支援されたCMUとの共同研究の成果を含む

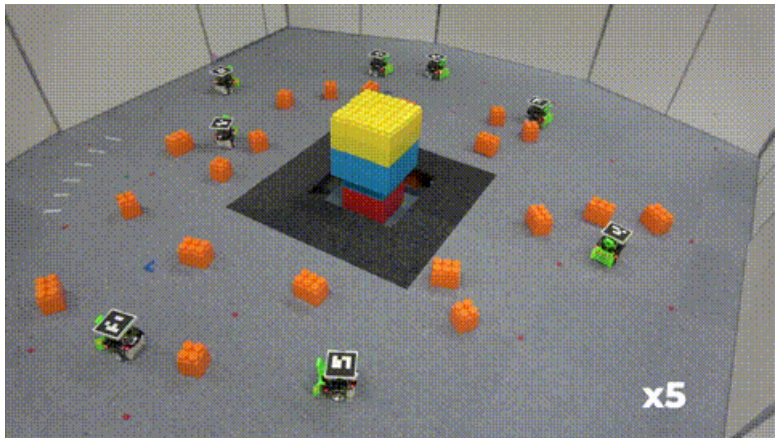
応用例４ |マルチロボットでの自律分散システムの開発

- Box Pushingタスクで少データ模倣学習による協調動作獲得
- LSTM Mixture of Expertsで，探索・運搬・収納の多様な動作を実現

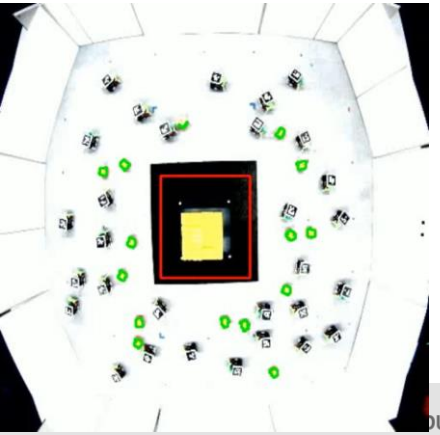
手動操縦し，
画像と
速度指令
を収集



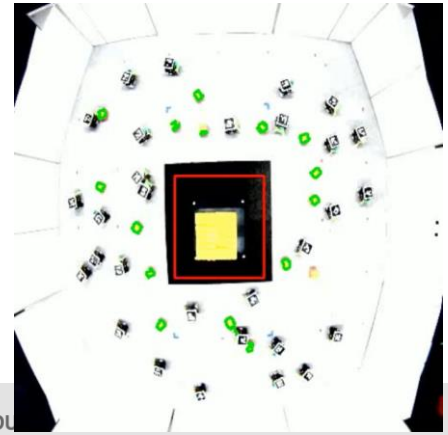
中央に集める



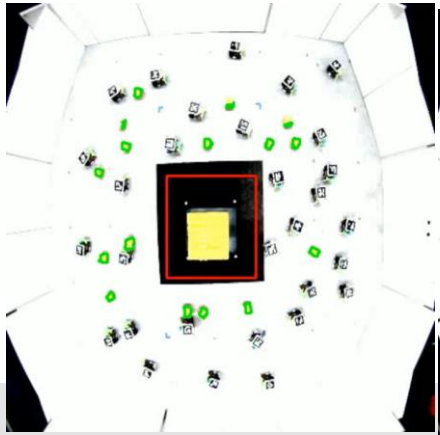
Round0
1台の手動操作で学習
全機にコピー



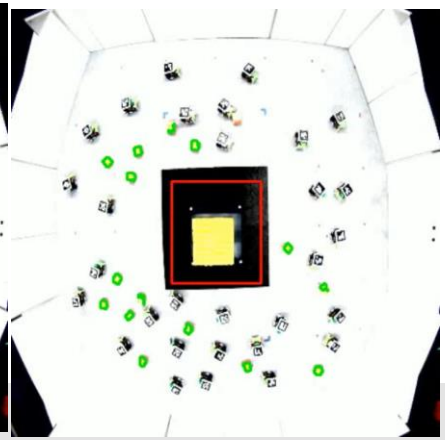
Round1
静止マウス置いた状態
で1台手動操作



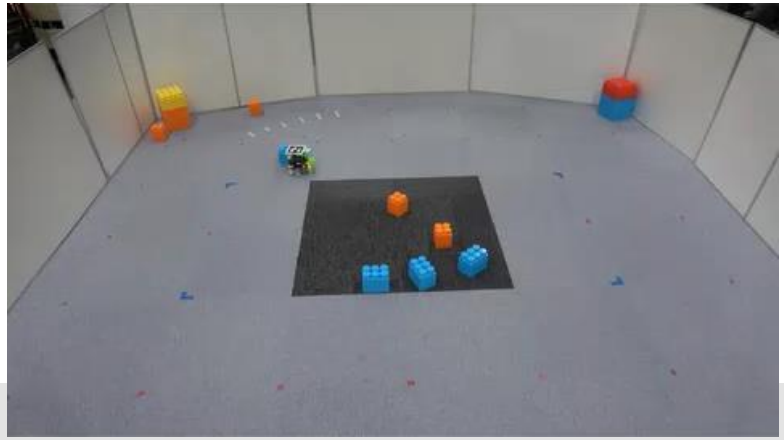
Round2
Round1環境で
1台手動操作



Round3
round2環境で
1台手動操作

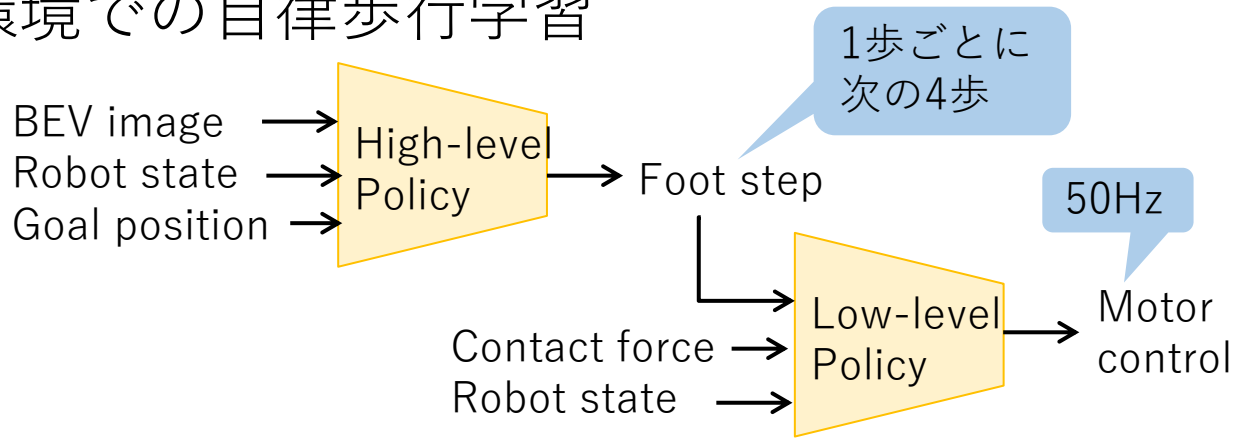


色分け（オレンジは左，青は右）

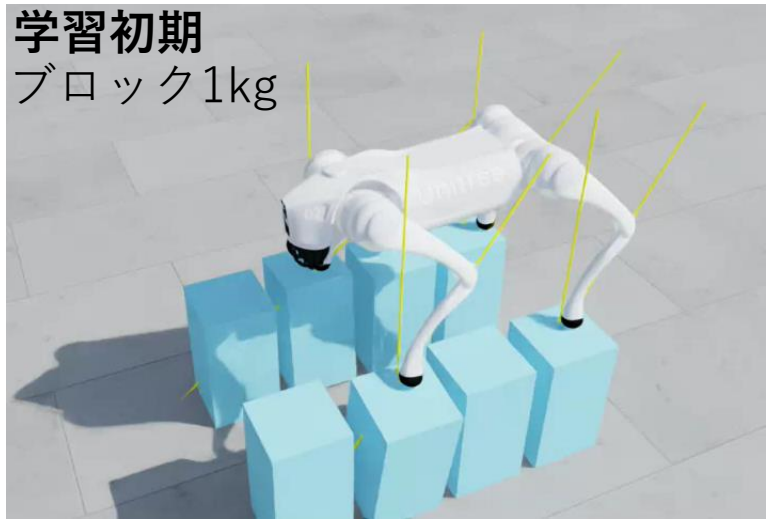


応用例５｜4脚ロボットによるマルチモーダル歩容獲得

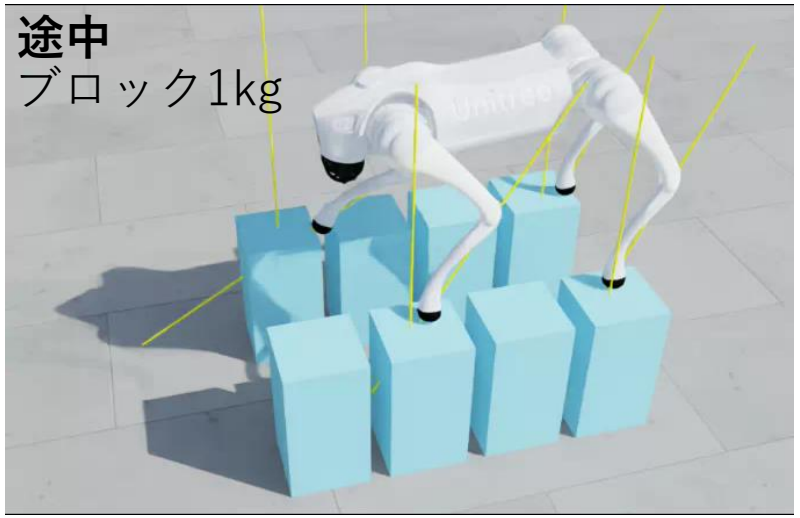
- Fast-slow 2階層モデルによる不安定環境での自律歩行学習
- (slow)視覚ベースfoot step plan
- (fast) 力覚ベース歩容生成



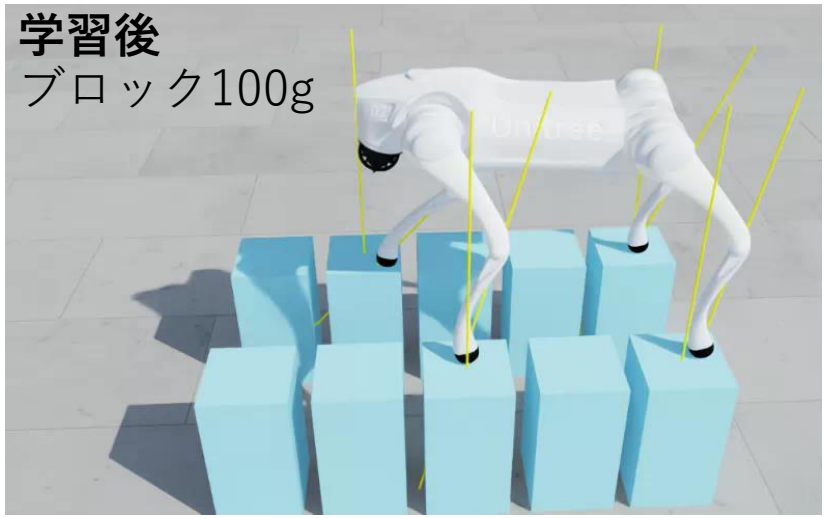
ブロック1kg，脚4本学習途中



ブロック1kg，脚4本学習後



ブロック100g，脚8本学習後

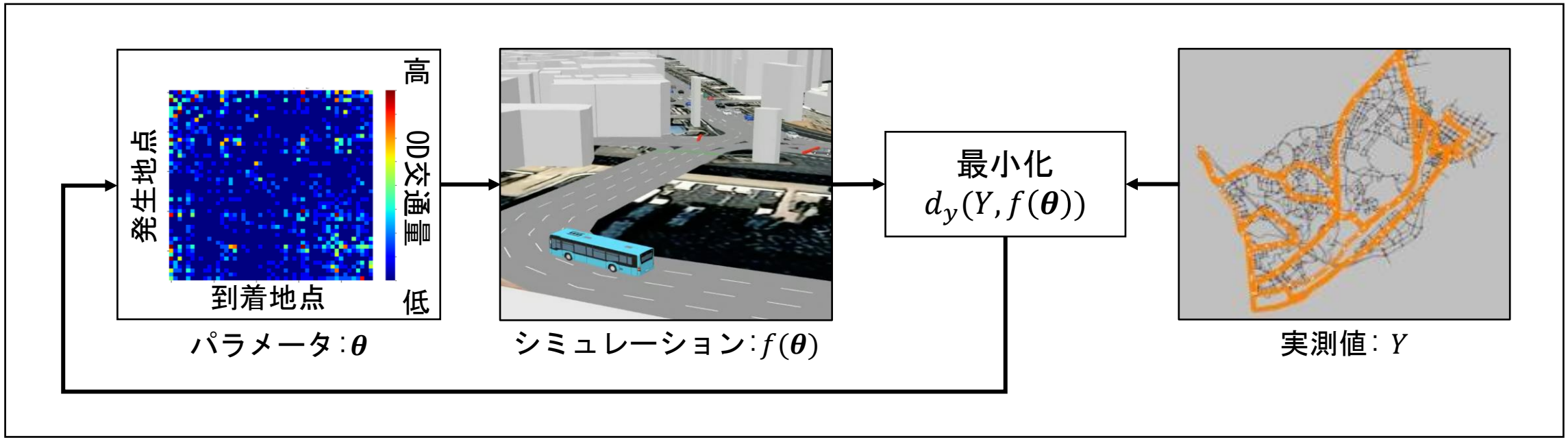


応用例６｜交通シミュレーションデータ同化

シミュレーション出力$f(\boldsymbol{\theta})$と実測値Yの差$d_y(Y, f(\boldsymbol{\theta}))$からパラメータ$\boldsymbol{\theta}$を調整（or 推定）する手法

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmin}}\, d_y(Y, f(\boldsymbol{\theta}))$$

パラメータθの調整には，シミュレーションの繰り返し実行が必要！



データ同化の流れ

応用例６｜交通シミュレーションデータ同化

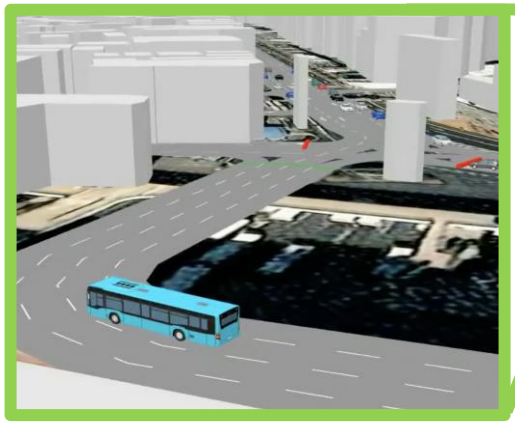
パラメータ調整に必要な計算時間 ＝ シミュレーション実行時間 × パラメータ探索回数 × シナリオ数

高速かつ高精度にパラメータ調整できる手法が必要！

約30分 → 約1500分 → 約22500分

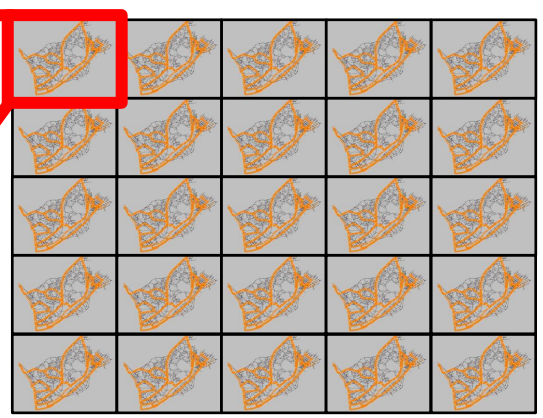
→約1日

→約2週間



シミュレーション1回
(実行時間約30分)

シミュレーション50回
(1シナリオのパラメータ調整)

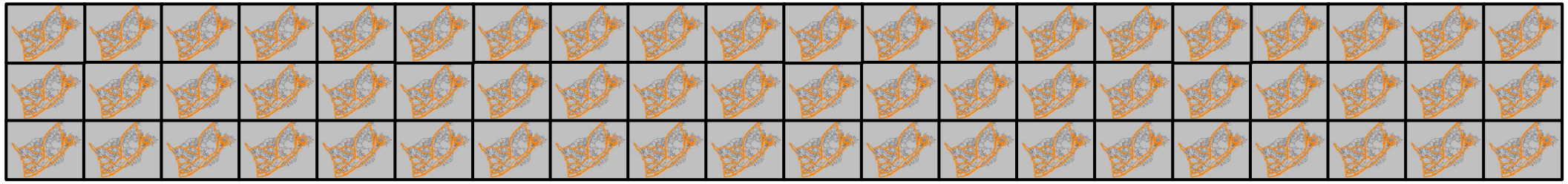


15シナリオのパラメータ調整

応用例６｜交通シミュレーションデータ同化

同一シナリオiを異なるモデル$g(\boldsymbol{\xi}_i), f(\boldsymbol{\theta}_i)$で調整した結果から関係性を表現する手法

簡易モデルを用いることで高速化が期待・シナリオ数の増加で高精度化が期待

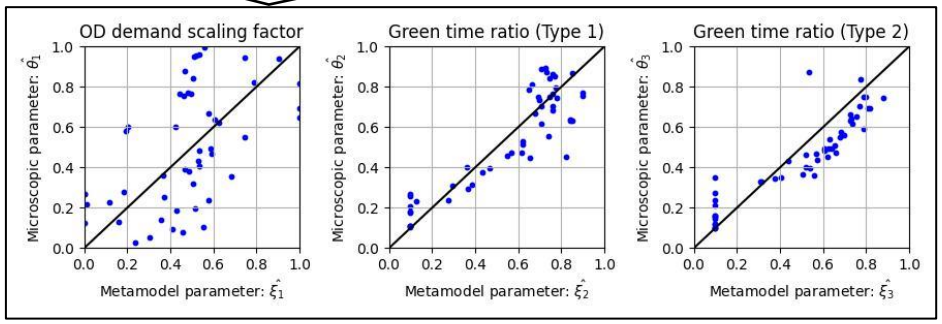


iシナリオの実測値Y_i

パラメータ調整
$d_y(Y_i, g(\boldsymbol{\xi}_i))$

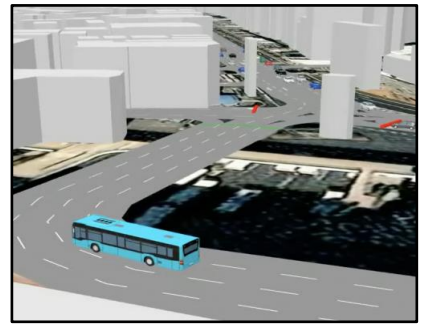
簡易モデル：$g(\boldsymbol{\xi}_i)$

$\{\widehat{\boldsymbol{\xi}_i}, \widehat{\boldsymbol{\theta}_i}\}$から$\boldsymbol{\theta} = h(\boldsymbol{\xi})$を表現する式を構築



各シナリオi毎の最適値ペア$\{\widehat{\boldsymbol{\xi}_i}, \widehat{\boldsymbol{\theta}_i}\}$

パラメータ調整
$d_y(Y_i, f(\boldsymbol{\theta}_i))$



シミュレーション：$f(\boldsymbol{\theta}_i)$

モデルブリッジを利用することで約2500倍の探索速度を実現

まとめ

- AI 導入を加速化する aiaccel の紹介
- 開発スピードの超高速化に対応
- モデルの超巨大化に対応（モデルブリッジ）
- 応用例の紹介

- 今後の課題
  - さらなる応用例（LLMのプロンプト最適化や量子コンピューティング）に適用しながら基盤技術を評価・洗練化
  - 充実したドキュメントや利用ノウハウの作成