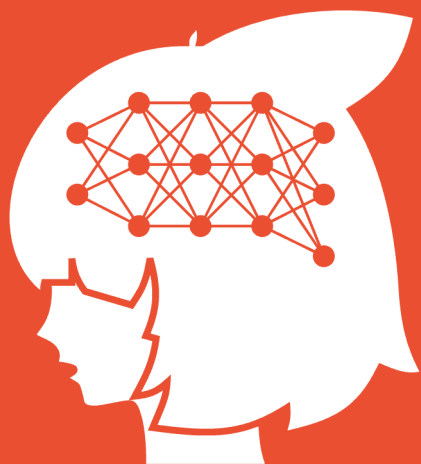




ML Shukai

Weekly P-AMI<Q>

2025/04/23 GesonAnko

自己紹介

- げそん (GesonAnko)

X(Twitter)@GesonAnkoVR

- ML集会 主催

- 自律機械知能の研究開発
- Pythonで機械学習のツール作る（機械学習より得意かもしれない）



自律機械知能に脳みそを焼かれた人。

今週の進捗

- PAMIQ-Core アルファリリース！！！！今日はその解説をするよ！

pamiq-core 0.1.1

`pip install pamiq-core`

✓ 最新バージョン

リリース日: 2025年4月22日

Framework for building AI agents with real-time adaptive learning capabilities.

ナビゲーション

- ≡ プロジェクトの説明
- 🕒 リリース履歴
- 📄 ファイルをダウンロード

プロジェクトの説明

pamiq-core

pypi v0.1.1 python 3.12+ License MIT docstyle google 🔄 Lint & Format / Tests / Type Check passing

pamiq-core is a framework for building AI agents. Developed for P-AMI<Q>, it enables train and inference in parallel, allowing agents to adapt continuously during interaction with their environment.

🌟 Features

- 🔄 **Parallel Architecture:** Simultaneous inference and training in separate threads
- ⚡ **Real-time Adaptation:** Continuously update models during interaction
- 🔒 **Thread-safe Design:** Robust synchronization mechanisms for parameter sharing and data transfers
- 🧩 **Modular Components:** Easy-to-extend agent, environment, and model interfaces
- 🔧 **Comprehensive Tools:** Built-in state persistence, time control, and monitoring

Verified details ✓
These details have been [verified by PyPI](#)

プロジェクトのリンク

- 🔗 Issues
- 📁 Repository

GitHub Statistics

PAMIQ-Core とは

- 深層モデル(AI)の学習と推論を同時並行して行うシステム

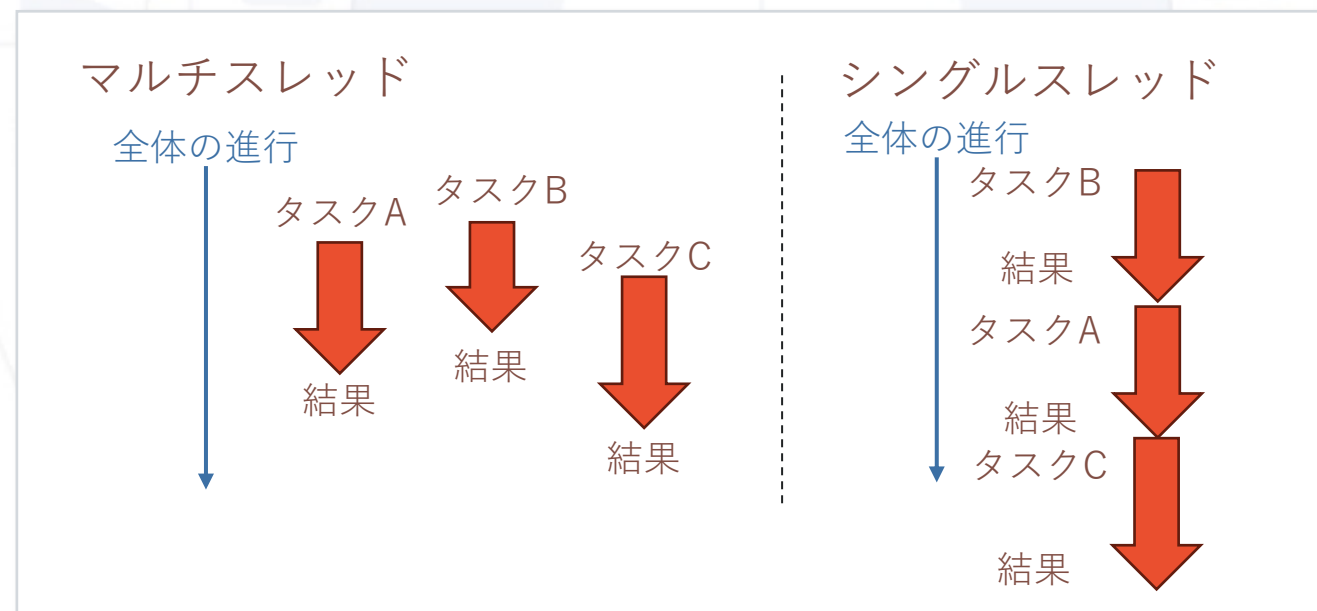
- 学習: ニューラルネットワークのパラメータを更新するフェーズ
- 推論: データを入力して求める結果を出力するフェーズ

- それを行うために…

- マルチスレッド化

- 深層モデルのパラメータ同期
 - データの受け渡し
 - その他 logging, 制御

- などを行うシステム

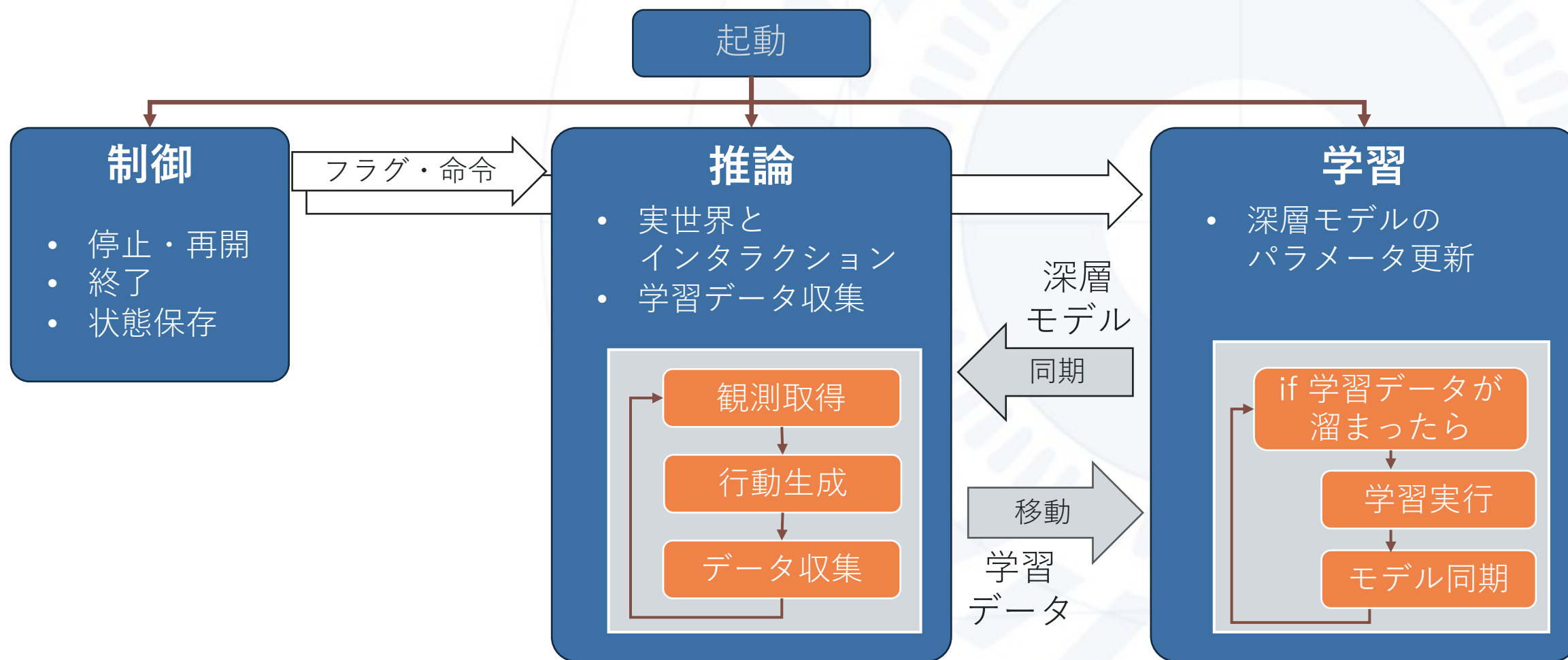


Systemの背景と目的

- リアルタイムに深層モデルを学習したい
 - 人も動物もリアルタイムに学習しているよね？
 - 学習と推論のフェーズが分かれているのは変だよね？
- 我々と共に過ごす中で現れる、**機械知能の変化**を観測したい。
 - 学習していく中で、どんな風に行動が変わるんだろうか？
- **非同期システムを実装するモチベーション**

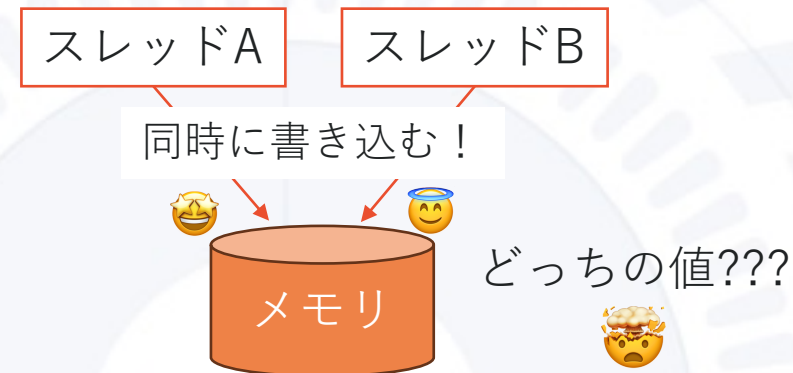
動作システムの概要

- 制御・推論・学習を並行処理（マルチスレッド化）



具体的な機能

- 安全な非同期処理
 - リソースへの同時アクセスの保護
 - 非同期処理をユーザーから隔離
- 拡張性の高い設計
 - 様々な機械学習ライブラリをバックエンドに使用可能
 - PyTorch (対応済) , Flax
- その他実験に便利なツール群
 - 状態保存・読み込み
 - 実行時間の加速・減速
 - PAMIQ Consoleによるシステムの制御



ドキュメンテーションサイト



- mkdocsで記述
 - ユーザーガイドやAPI Referenceが読める！

The screenshot shows the PAMIQ-Core documentation homepage. The header includes the PAMIQ-Core logo, a search bar, and the repository name MLShukai/pamiq-core. The main content area has a green header with navigation links: Home, User Guide, and API References. The main text welcomes users to the PAMIQ-Core Documentation and describes the framework as a tool for building AI agents with real-time learning capabilities. A 'Features' section lists several key capabilities: Parallel Architecture, Real-time Adaptation, Thread-safe Design, Modular Components, and Comprehensive Tools. A 'Table of contents' sidebar on the right lists: Features, Installation, Basic Example, Remote CLI Control, Documentation, and Contribution. At the bottom, a diagram illustrates the system architecture, showing the flow from Launch to Inference and Training, with sub-components like Control, Inference (Get Observation, Generate Action, Collect Data), and Training (Update Deep Model Parameters, Execute Training, Sync Parameters).

The screenshot shows the 'Launch' page in the PAMIQ-Core documentation. The header is consistent with the previous page. The main content area has a green header with navigation links: Home, User Guide, and API References. The main text explains that the 'launch' function is the entry point for starting a PAMIQ-Core system. A 'Basic Usage' section provides a code snippet for launching the system. A 'Table of contents' sidebar on the right lists: Basic Usage, Common Configuration Scenarios, Accelerated Learning, Resumable Training, and API Reference.

```
from pamiq_core import launch, LaunchConfig, Interaction

# Create your agent and environment
agent = YourAgent()
environment = YourEnvironment()

# Create an interaction between them
interaction = Interaction(agent, environment)

# Launch the system
launch(
    interaction=interaction,
    models={"model_name": your_model},
    data={"buffer_name": your_data_buffer},
    trainers={"trainer_name": your_trainer},
    config=LaunchConfig(
        states_dir="/.saved_states",
        max_uptime=3600, # Run for 1 hour
        time_scale=2.0 # Run at 2x speed
    )
)
```

チームメンバーへの感謝



- Zassouさん
 - PAMIQ Coreのモデルプロトコル、学習プロトコルの定義
 - PyTorchバックエンドの実装
 - PAMIQ Consoleの実装
- Klutzさん
 - 基本的なマネジメント
 - メンバーの間の調停
- Karudanさん
 - 時間管理モジュールの実装
 - 途中でリタイアしましたが、初めてのチーム開発にPAMIQ-Coreを選ぶという素晴らしいチャレンジでした！
- Alciaさん
 - Thread制御と厳密なテスト
 - 基本スレッドクラスの実装
- Myxyさん
 - 広範なシステムコードのレビュー（主に私のコード）

これからももう少し続くよ



- テストツールの拡充
- サンプルプロジェクトの実装
- ドキュメンテーションの整備
- 実際にPAMIQ-Coreを使って開発を行い、バグを見つける…