AI Code Hackers

人工知能のコードを
ハックする会 #1

アジェンダ

- **概要説明**
  - 自己紹介
  - 本会の趣旨
- **ソースコード解析手法の共有/検討**
  - プロファイラの説明
  - 可視化ツールの説明
  - 実演
- **次回以降にハックするOSSの検討**
  - OpenNMT (MT) / Kaldi (ASR) / Merlin (TTS) …

自己紹介

神谷 亮平

株式会社LABBIZ 代表取締役
ソフトウェアエンジニア
「 THE STAGE .tech 」を企画・運営

▼開発経験

・アルゴリズム/ライブラリ開発（C / C++、Python）
・組み込みソフトウェア開発（C）
・WEBアプリケーション開発（Java、 PHP 、JavaScript、 Python、Go）
・Windowsデスクトップアプリケーション開発（C#）

▼略歴

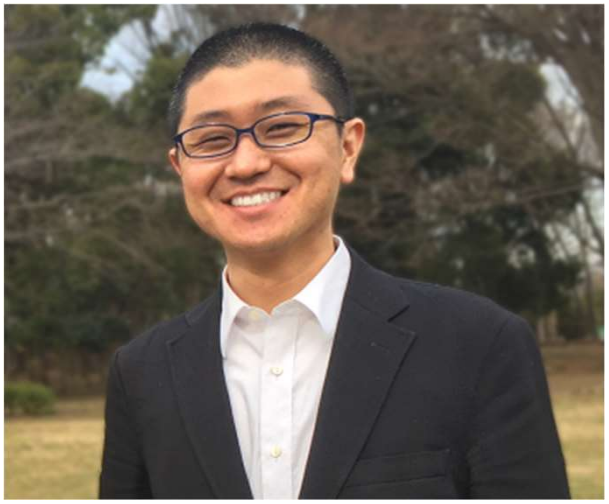
時計メーカー研究開発職 ⇒ データ分析会社新規事業開発職
⇒ IoTスタートアップ研究開発職 ⇒ 創業

AI Code Hackers

前職

指輪型ジェスチャーコントローラ「 Ring 」の開発者

現職



創り手の活躍にフォーカスしたインタビューメディア
「THE STAGE .tech」を運営

本会の趣旨

目的

- 人工知能(パターン認識/機械学習)のソースコードの解読方法を学び、共有する
- プログラムの改良や新しい応用を検討する
- 知的好奇心/探究心を追求して楽しむ

人工知能のコードを読んで
ハックして遊びたい！

ソースコード解析

解析手法

- **静的解析**
  - プログラムを実行せずに解析する
  - ソースコードを読む
  - 文法・スタイルチェッカーを使う
- **動的解析**
  - プログラムを実行して解析する
  - デバッガを使う
  - プロファイラを使う

今回はプロファイラを使う
動的解析手法を紹介

プロファイラ

- **プロファイラとは？** ※ Wikipediaより引用
  - プロファイラ（英: Profiler）は性能解析ツールであり、
    プログラム実行時の各種情報を収集する。
  - 関数呼び出しの頻度やそれにかかる時間を計測する。
  - 出力は記録したイベントの羅列（トレース）の場合と、
    観測したイベント群の統計的要約（プロファイル）の場合がある。

- **プロファイラの例**
  - gprof (対応言語: C/C++, Pascal, Fortran)
  - Java HPROF
  - Python cProfile / profile (Python モジュール)

今回はPython

Pythonのプロファイラ

- **Python cProfile / profile**

  cProfile と profile は 決定論的プロファイリング (deterministic profiling) を行います。
  プロファイル (profile) とは、プログラムの各部分がどれだけ頻繁に呼ばれたか、そして実行にどれだけ時間がかかったかという統計情報です。
  pstats モジュールを使ってこの統計情報をフォーマットし表示することができます。

  ※ 公式ドキュメント日本語訳より引用

cProfileとprofile
どっちにする？

Pythonのプロファイラの比較

- **Python cProfile**

  cProfile はほとんどのユーザーに推奨されるモジュールです。
  C言語で書かれた拡張モジュールで、オーバーヘッドが少ないため長時間実行されるプログラムのプロファイルに適しています。

- **Python profile**

  profile はピュア Python モジュールで、 cProfile モジュールはこのモジュールのインタフェースを真似ています。
  対象プログラムに相当のオーバーヘッドが生じます。
  もしプロファイラに何らかの拡張をしたいのであれば、こちらのモジュールを拡張する方が簡単でしょう。

※ 公式ドキュメント日本語訳より引用

とりあえず cProfile！

cProfileでOpenNMT-pyをプロファイリング

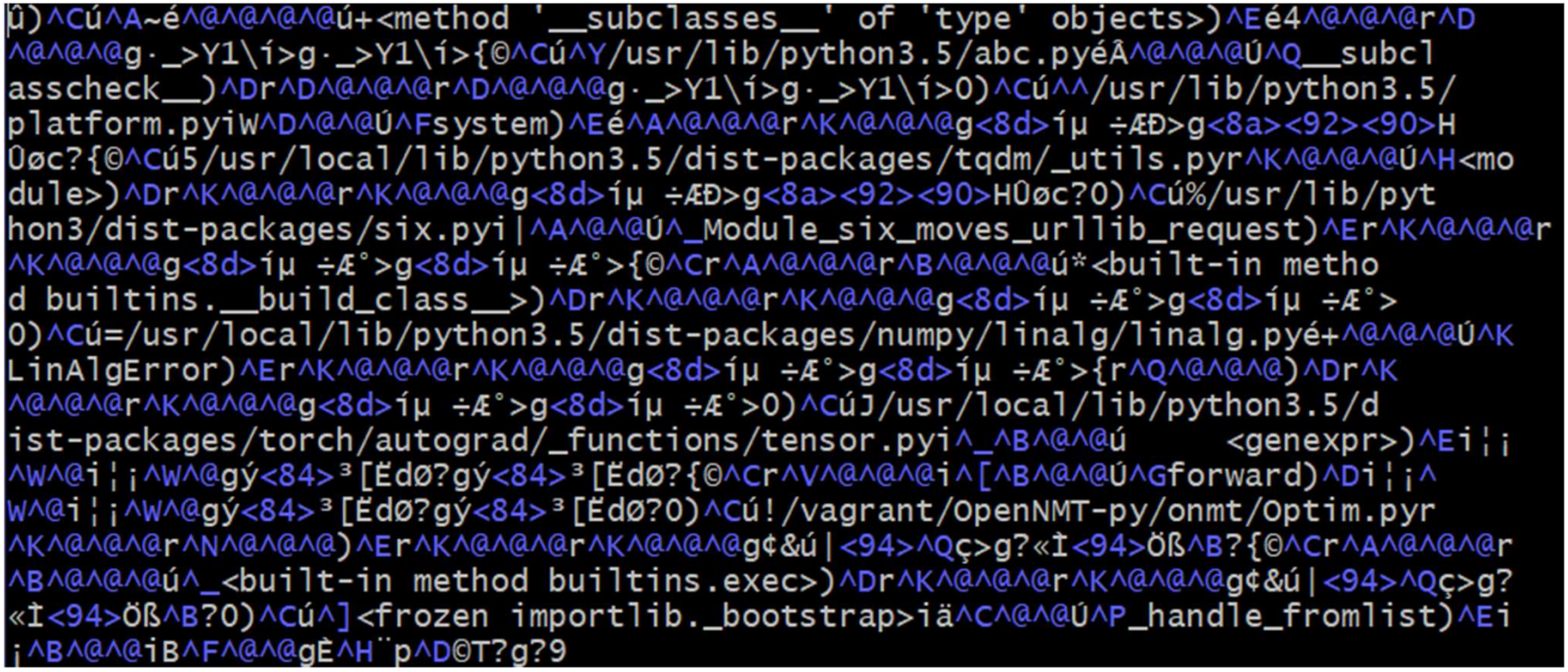
- **OpenNMT-pyとは？**
  - Open-Source Neural Machine Translation
  - オープンソースな機械翻訳システム
  - luaで書かれたOpenNMTのpython移植版（開発中）
  - 詳しくは後述
- **モデル学習のプロファイリング**
  - 「-m cProfile -o <output file name> 」というオプションをつけて実行

```
$ python3 -m cProfile -o train.prof train.py -data data/demo -save_model ./demo-model
```

一晩待ちます。

プロファイルの結果

$ vi train.prof



AI Code Hackers

可視化ツールを使う！

プロファイルの可視化

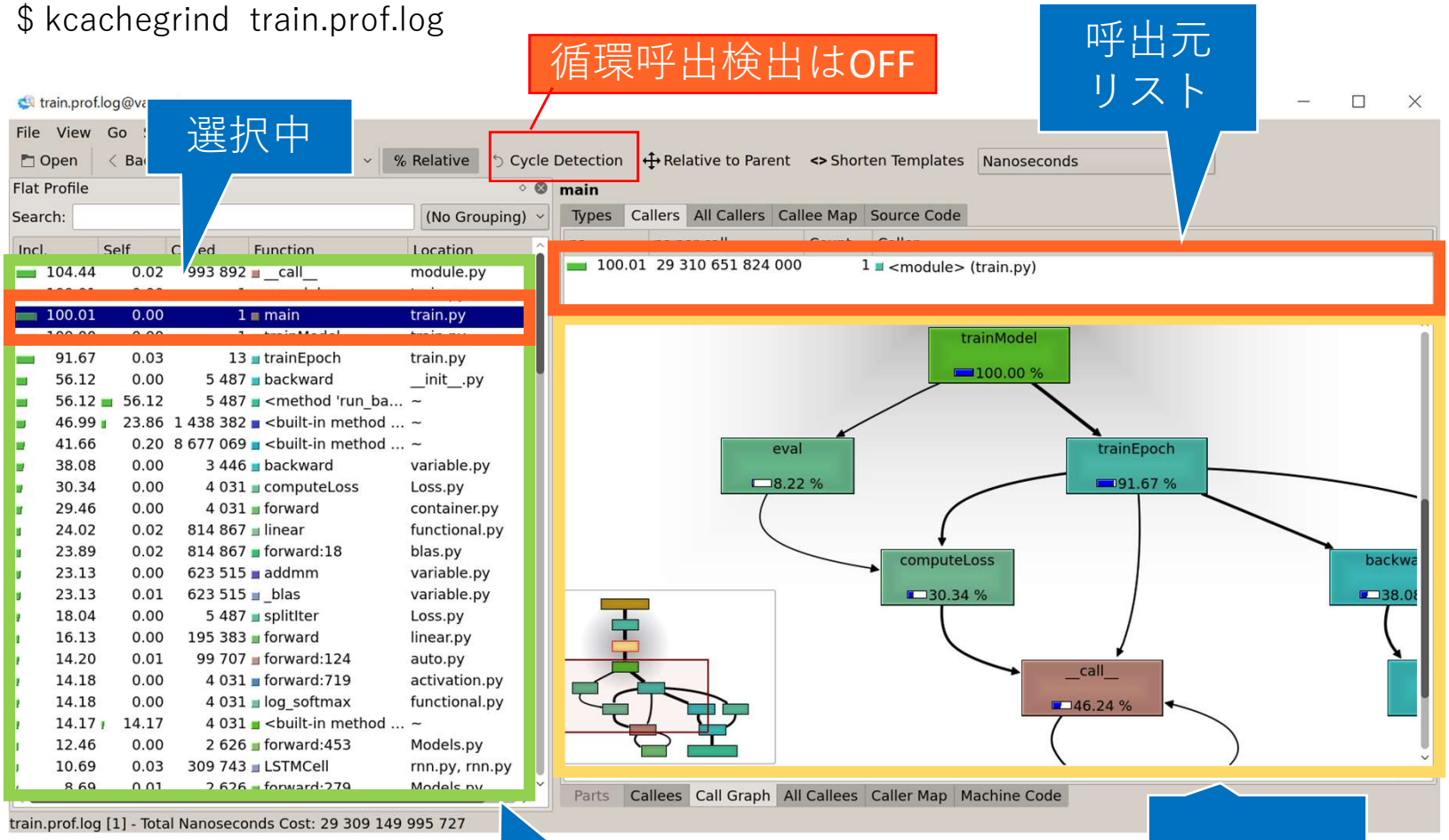
- **gprof2dot + dot**
  - プロファイルのデータからgraphvizのdotファイルを生成
  - dotファイルをdotに渡すことで、関数の呼び出し関係を可視化した「コールグラフ」画像を生成可能
- **gprof2dot + xdot.py**
  - dotファイルをxdot.py（GUIプログラム）でコールグラフをインタラクティブに可視化
  - 拡大・縮小したり、ハイライト表示が可能
- **pyprof2calltree + KCachegrind**
  - pyprof2calltreeで変換したデータをKCachegrind（GUIプログラム）でインタラクティブに可視化

pyprof2calltree + KCachegrind を選択

pyprof2calltree + KCachegrind

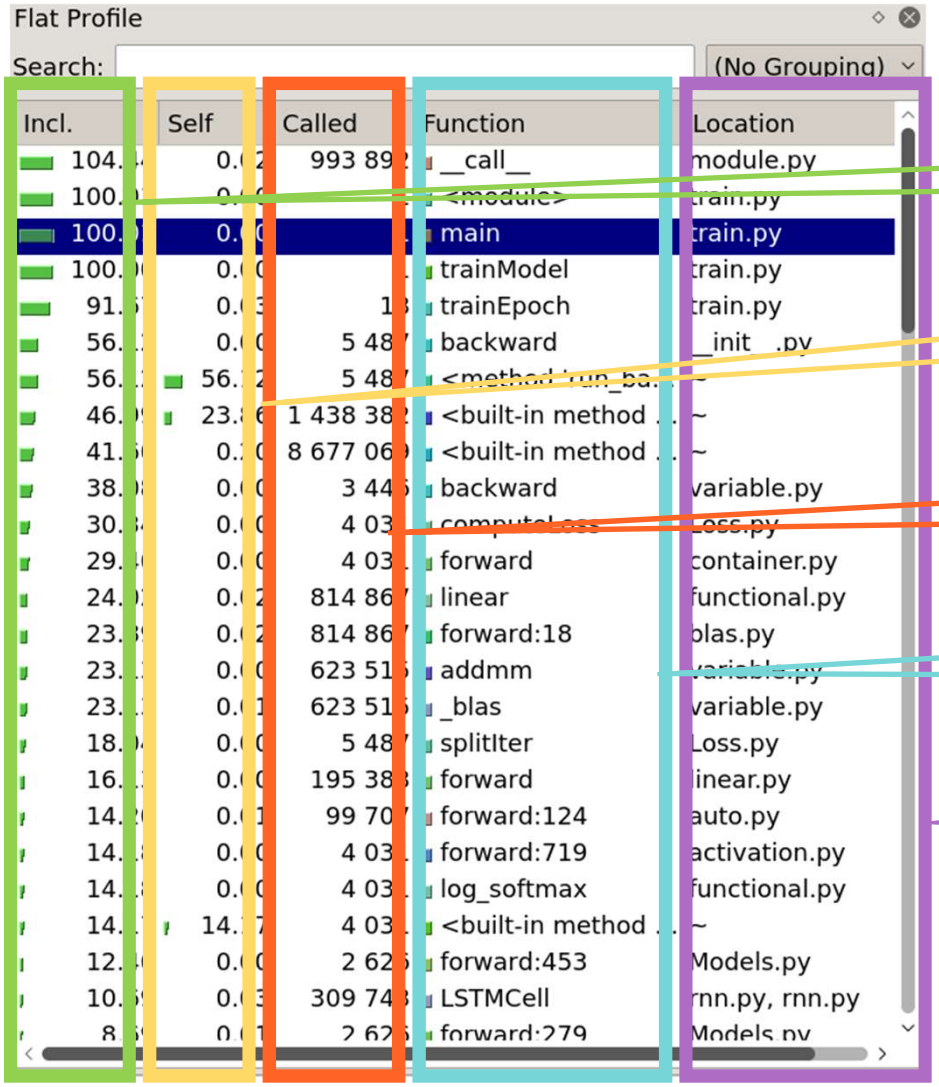
$ pyprof2calltree -i train.prof -o train.prof.log

$ kcachegrind train.prof.log



AI Code Hackers

KCachegrind - Flat Profile



計算量積算値

関数毎の計算量

関数のコール回数

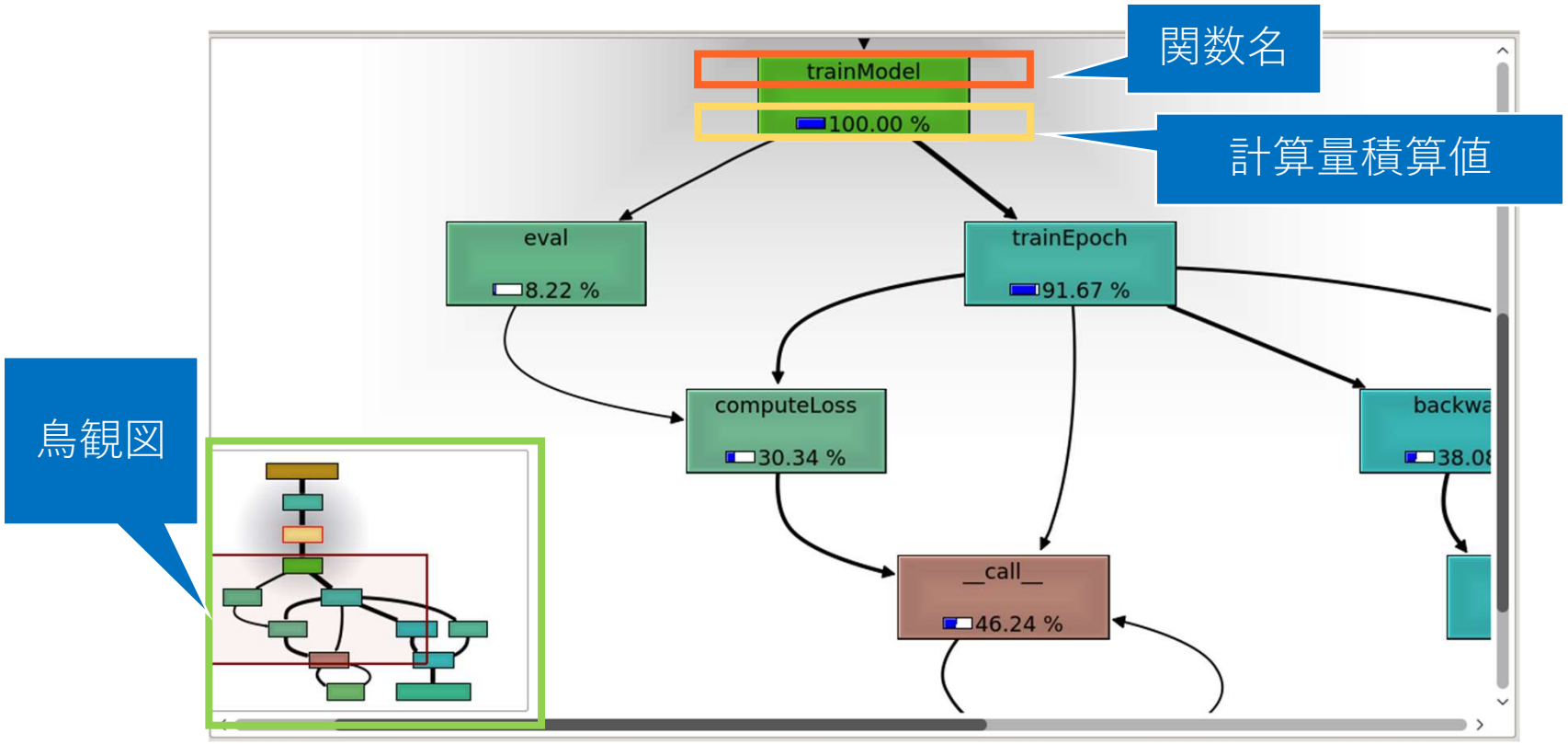
関数名

ファイル名

AI Code Hackers

KCachegrind - Call Graph

関数の呼び出し・依存関係のグラフ



AI Code Hackers

実演

どうやってハック？

ハック例（案）

- プロファイルでボトルネックを検出し、コードを効率化
- コードと論文を読み比べて、他の論文の実装に置換
- 入力データのクレンジング・フィルタリングを実施して性能UP
- 何か面白い応用を実施

Q.どのコードをハックしたい？
（次回以降の相談）