課題：ラジオの観測者

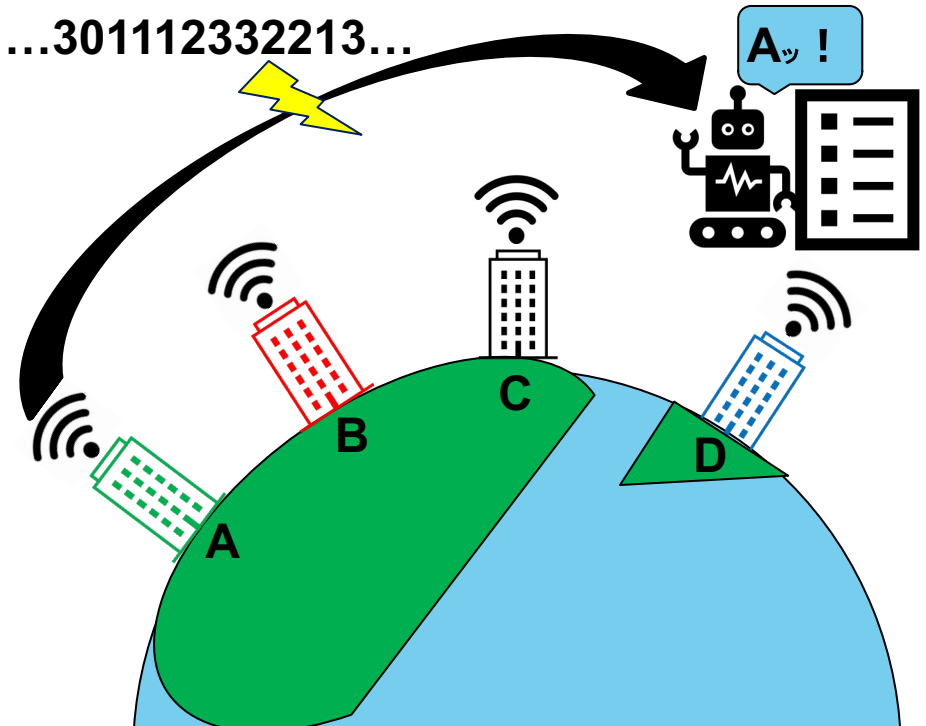
- 観測者は聞こえてくる音声データの一部をもとに基地局を当てる
  - 観測者は基地局毎の音声内容を知っている
  - 音声データは0,1,2,3で構成されている

:観測者

:基地局

:ノイズ

:基地局毎の音声内容

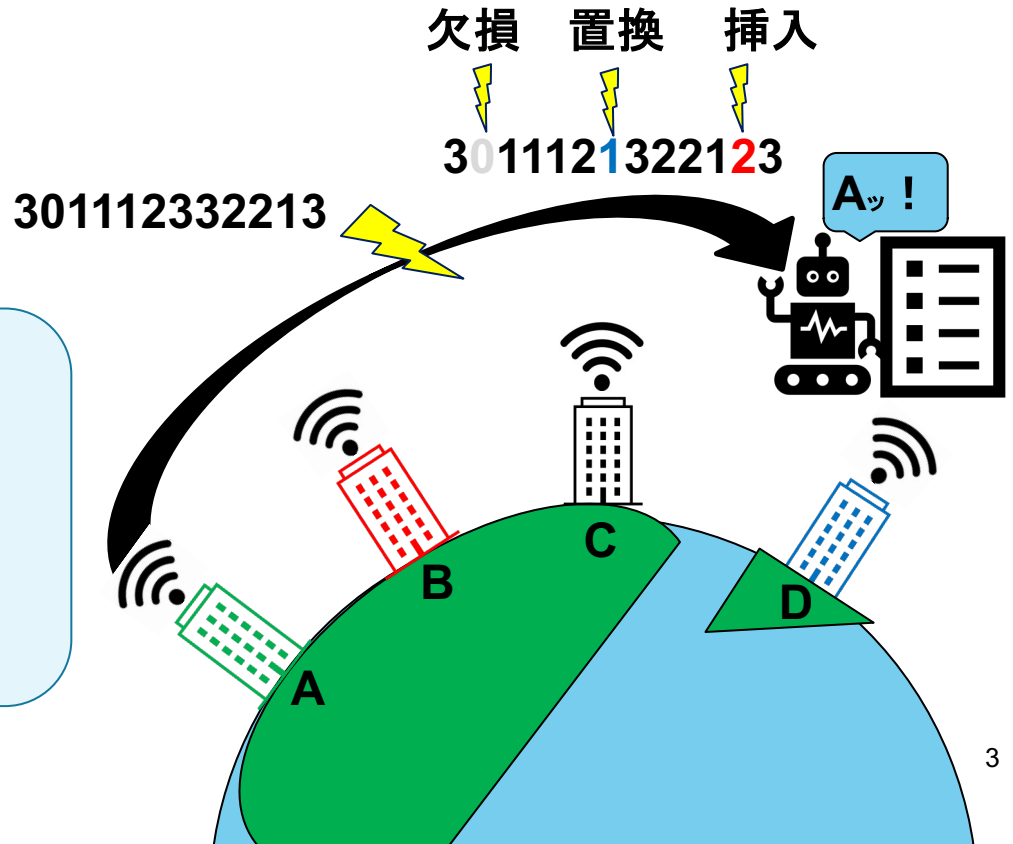


- ノイズにより音声データは損傷を受ける
  - 損傷は欠損、置換、挿入の3種類
    - 欠損:ある文字がなくなる
    - 置換:ある文字が他の文字に置き換わる
    - 挿入:ある文字が追加される

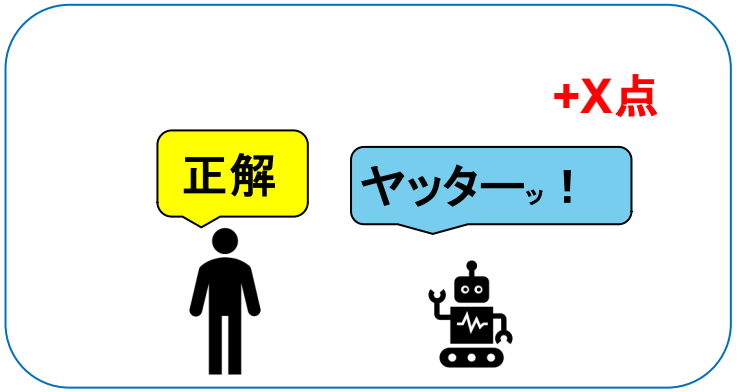
ノイズ無 : 30111233221 3

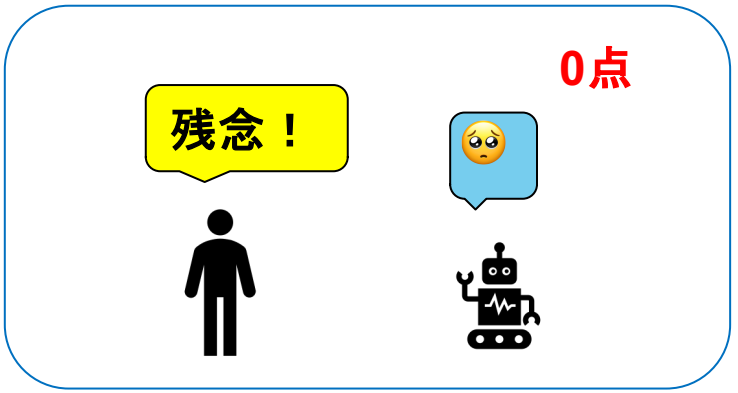
欠損　置換　挿入

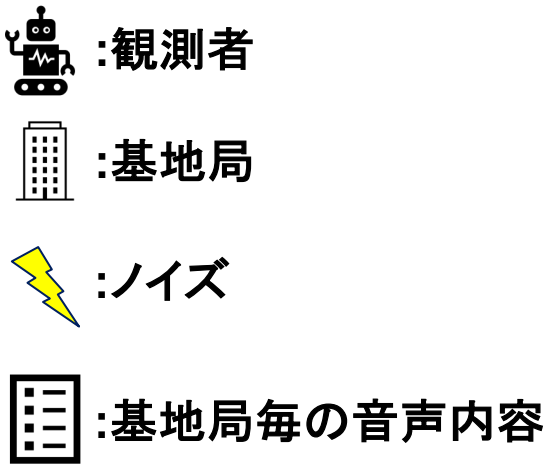
ノイズ入 : 3011121322123

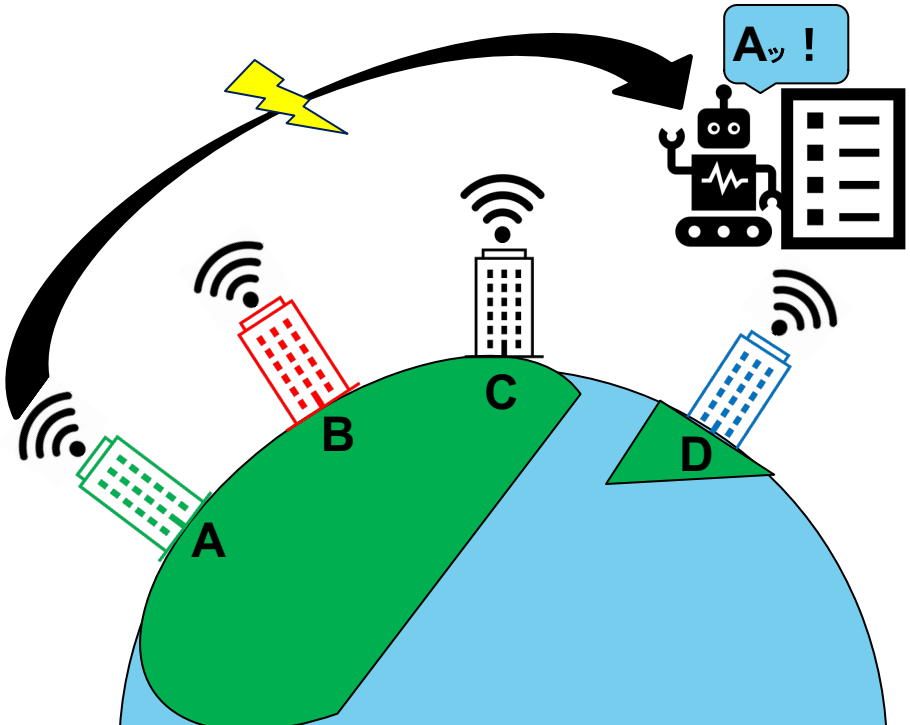


- 答えた基地局と聴いた基地局があってれば得点を得る

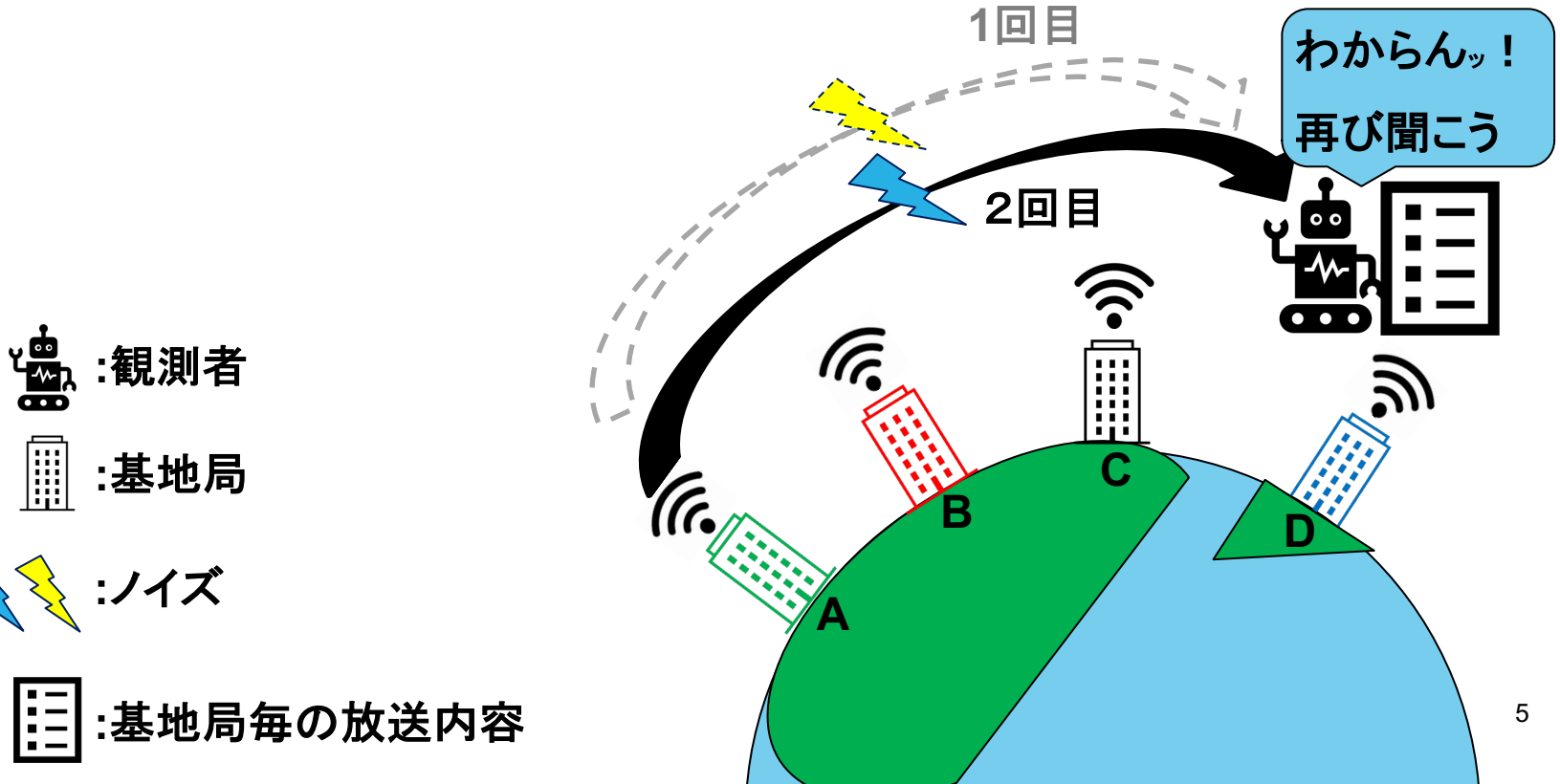






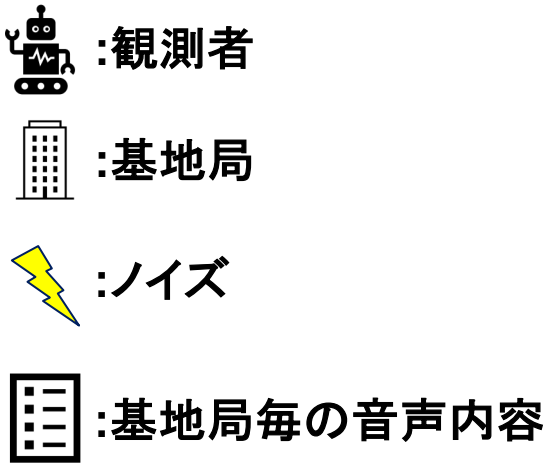


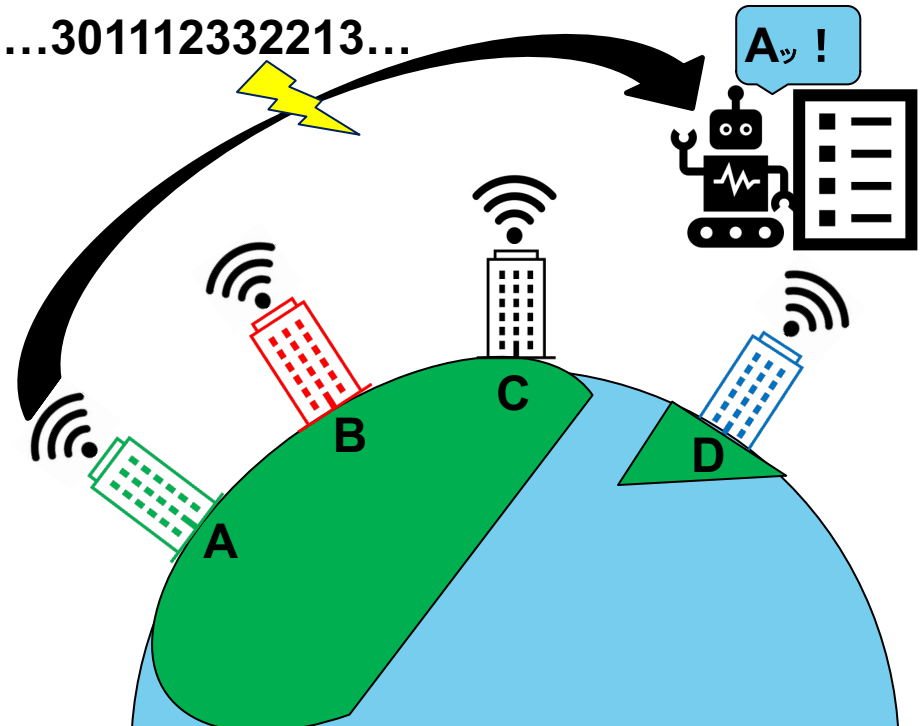
- 基地局が特定できなければ、同じ箇所を聞き直すことも可能
  - ただし、聞き直した分だけペナルティが発生する
  - ノイズは同じとは限らない



課題

- 音声の一部を聞き、どの基地局からかを当てる問題を解きます
  - 基地局は100個
  - 問題は全100問
- 高得点を得られるプログラムを作成してください





各データについて

- 各基地局の文章データ
  - どの基地局も文章データの長さは5000
  - $S_i = s_{i,1}s_{i,2} \cdots s_{i,5000}$ $(s_{i,j} \in \{0,1,2,3\})$
  - 基地局は100局

- 各クエリの文章データ
  - i番目のクエリでは長さLの文字列Q_iが与えられる
  - L は 20 以上 100 以下
  - $Q_i = q_{i,1}q_{i,2} \cdots q_{i,L}$ $\left(q_{i,j} \in \{0,1,2,3\}\right)$
  - クエリは100個

各データについて

- クエリのエラー率
  - 挿入エラー：p_{ins} % $(1 \leq p_{ins} \leq 10, \text{整数})$
  - 置換エラー：p_{sub} % $(1 \leq p_{sub} \leq 10, \text{整数})$
  - 欠損エラー：p_{del} % $(1 \leq p_{del} \leq 10, \text{整数})$

配布データについて

- エラー率・基地局の文章データ・クエリデータ
  - ファイル名：idata
  - フォーマット

    p_{ins} (半角スペース) p_{sub} (半角スペース) p_{del} (改行)
    S_1 (改行)
    S_2 (改行)
    …
    S_{100} (改行)
    Q_1 (改行)
    Q_2 (改行)
    …
    Q_{100} (改行)

配布データについて

- クエリに対する正解データ
  - ファイル名：answer
    - i 行目に i 番目のクエリの答えと、どの区間を切り取ったかが与えられる
      - $p_{ins,i}, p_{sub,i}, p_{del,i}$：$i$ 番目の挿入、置換、欠損確率
      - ans_i：i 番目のクエリの答え
      - $base_i$：S_{ans_i} から切り取った部分の文字列(ノイズ付加前)

idata, answer のデータの組を10セット配布します.（「チーム作業」⇒「資料」⇒ all.zip）

配布データについて

- クエリに対する正解データ
  - フォーマット

    p_{ins1}(半角スペース)p_{sub1}(半角スペース)p_{del1}(改行)
    ans_1(半角スペース)$base_1$(改行)
    ans_2(半角スペース)$base_2$(改行)

    …
    ans_{100}(半角スペース)$base_{100}$改行)

idata, answer のデータの組を10セット配布します.（「チーム作業」⇒「資料」⇒ all.zip）

提出に関して

- プログラムは一つのC言語ファイルとしてください．ただし、提出するファイルは “ask.h” をincludeしておくこと．（採点の際には配布した”ask.h”と同一のものを用いる．）

- 出力フォーマット
  - i $(1 \leq i \leq 100)$ 行目には i 番目のクエリの答えを、出力ファイルに出力すること．
    - 例えば、5番目のクエリの答えが 13番目の音声だった場合、5行目に “13” と出力し、改行すること．

- “sample.zip” にサンプルコードを用意したので、入力の受け取り方や出力方法を確認すること．

提出に関して

- クエリの聞き直しについて
  - ヘッダファイル “ask.h”にある ask()関数を用いること.
  - char* ask(int query_id, char* answer_file_name)
    - query_id
      - i $(1 \leq i \leq 100)$ 番目のクエリについて聞き直したい場合は query_id = i とすること.
    - answer_file_name
      - クエリに対する正解データ のファイル名（のパス）を指定すること.

提出に関して

- 性能評価に使用する実行コマンド

gcc adsb_{グループID}.c –o run_{グループID} –lm –O2
timeout 10 run_{グループID} (inputファイル名) (outputファイル名) (answerファイル名)

  - コマンドライン引数の第一引数に idataファイルへのパス、第二引数に出力ファイルへのパス、第三引数に answerファイル（クエリに対する正解データ）へのパスを受け取れるように実装してください.
  - 詳細は配布されるsample.zip内のREADMEを参照してください.

性能評価

- 評価データ
  - サンプルデータを生成したのと同じモデルを用いて作成．（つまり，サンプルデータと同質のデータを使って評価．）
- 実行速度
  - 計測環境（予定）
    - OS: Ubuntu 20.04 LTS
    - gcc: 9.4.0 (Ubuntu 9.4.0-1ubuntu1~20.04.1)
    - CPU: AMD Ryzen Threadripper 3970X
  - 実行時間は **10秒**で打ち切り

性能評価

- スコア
  - 各クエリについて
    - 正解した場合
      - $\mathbf{max(10, 100 - (}$そのクエリで質問した数$\mathbf{)\times 5)}$点を加算
    - 不正解の場合
      - 得点は貰えないが、そのクエリで質問した回数に関わらず**マイナスの値が加算されることはない**
  - 100個のクエリの合計得点をそのテストケースのスコアとする

提出時の動作確認

- 提出物は，以下の環境で動作することを確かめてから提出すること．
- OS　：ubuntu-20.04.1-desktop-amd64
  開発：build-essential

- Linux環境をお持ちでない方は，**インストールして利用することを推奨**しますが，インストールせずに利用することもできます．（次を参照）

1. https://releases.ubuntu.com/20.04/ より ubuntu-20.04.1-desktop-amd64.iso を入手
2. VirtualBoxをインストール
3. 仮想マシンの作成

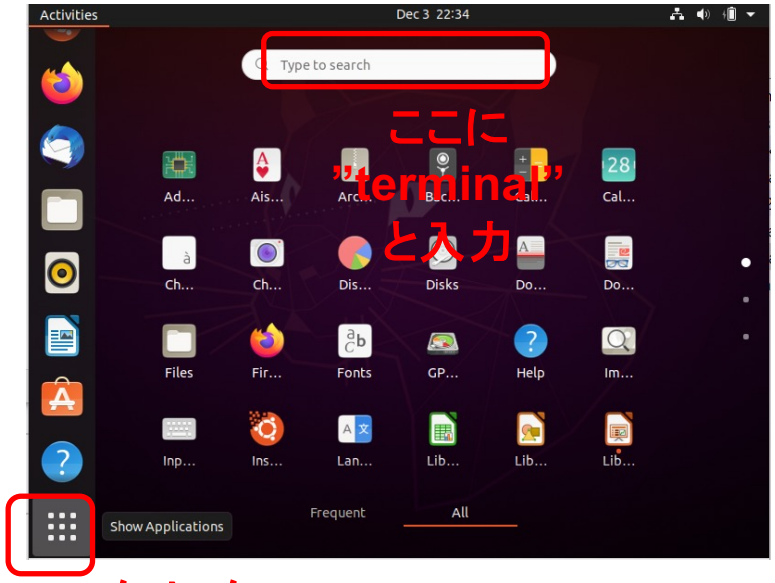
4. isoイメージを設定

「光学ドライブ」をクリックして
「ディスクファイルを選択」をクリックした後で
ubuntu-20.04.1-desktop-amd64.isoを設定

5. 仮想マシンの電源を入れる

クリック

6. Live CDを起動



クリック

クリック

7. terminalを立ち上げて以下を実行

sudo apt-get install build-essential

どんな方法で解くか？

- 観測者の間違いを考慮しながら，正解データとどうやって比較をするか？
- 文書データやクエリデータが大きいので，高コストな比較法では計算の打ち切りに合う可能性も？
- ノイズの確率は何か関係ある？

- ぜひ活発な議論を
  - Gather (https://app.gather.town/app/Npzm8sOATzCqgiYp/WasedaCS)
- ソースの共有
  - github (https://github.com/)
  - Dropbox
  - Google Drive