Руководство по пользовательскому филогеномному пайплайну

Это руководство описывает, как использовать пайплайн на Python для построения филогенетического дерева по набору пользовательских белок-кодирующих генов. Пайплайн предназначен для запуска в WSL (Ubuntu) на Windows и использует стандартные инструменты биоинформатики, такие как Prodigal, HMMER, MAFFT и FastTree.

---

1. Установка и настройка

Необходимое ПО (установите через терминал WSL):

```
sudo apt update
sudo apt install python3 python3-pip python3-venv mafft fasttree prodigal hmmer
```

Создание виртуального окружения Python:

```
cd ~/your_project_folder
python3 -m venv venv
source venv/bin/activate
pip install biopython
```

---

2. Структура каталогов

Разместите файлы в следующей структуре:

```
project/
├── data/
│   ├── genomes/              # Входные геномы (.fna)
│   └── hmms/                 # Пользовательские HMM-профили генов (.hmm)
├── results/
│   ├── aligned_genes/        # Выравнивания MAFFT
│   ├── concatenated/         # Супермассив
│   ├── extracted_seqs/       # Лучшие совпадения по каждому гену
│   ├── hits/                 # Результаты hmmsearch
│   ├── parsed_hits/          # Обработанные таблицы совпадений
```

```
│   ├── predicted_genes/      # Предсказанные Prodigal белки (.faa)
│   └── tree/                 # Финальное филогенетическое дерево
├── scripts/                  # Все Python-скрипты
│   ├── align_genes_mafft.py         # Шаг 5: Выравнивание последовательностей с
помощью MAFFT
│   ├── concatenate_alignments.py    # Шаг 6: Объединение выравниваний в
супермассив
│   ├── extract_best_hit_seqs.py     # Шаг 4: Извлечение лучших
последовательностей с помощью Biopython
│   ├── parse_hmmsearch_hits.py      # Шаг 3: Разбор вывода hmmsearch для
получения лучших совпадений
│   ├── run_fasttree.py              # Шаг 7: Построение дерева с помощью
FastTree
│   ├── run_hmmsearch.py             # Шаг 2: Поиск генов с помощью hmmsearch
│   └── run_prodigal.py              # Шаг 1: Предсказание генов с помощью
Prodigal
```

---

3. Использование пайплайна

Запускайте каждый скрипт в приведённом ниже порядке. Все скрипты находятся в каталоге `scripts/`.

Шаг 1: Предсказание белков (Prodigal)

```
python3 scripts/run_prodigal.py
```

Что делает:

- Использует **Prodigal** для предсказания белок-кодирующих генов в геномах.

Вход: Файлы геномов в `data/genomes/`

Выход: Файлы белков (.faa) в `results/predicted_genes/`

Шаг 2: Поиск пользовательских генов (HMMER)

```
python3 scripts/run_hmmsearch.py
```

Что делает:

- Использует **HMMER (hmmsearch)** для поиска белков по пользовательским HMM-профилям, выявляя интересующие гены.

Вход:

- Файлы белков из шага 1
- HMM-профили из `data/hmms/`

Выход: Таблицы результатов HMMER в `results/hits/`

Шаг 3: Разбор результатов HMMER (Python)

```
python3 scripts/parse_hmmsearch_hits.py
```

Что делает:

- Разбирает вывод HMMER, определяя лучшее совпадение для каждого гена и генома.

Вход: Таблицы совпадений HMMER из шага 2

Выход: Обработанные таблицы в `results/parsed_hits/`

Шаг 4: Извлечение лучших последовательностей (Python + Biopython)

```
python3 scripts/extract_best_hit_seqs.py
```

Что делает:

- Использует Biopython для извлечения последовательностей белков по лучшим совпадениям из предыдущего шага.

Вход:

- Обработанные таблицы из шага 3
- Файлы белков из шага 1

Выход: Один FASTA-файл на ген с лучшими совпадениями, сохраняется в `results/extracted_seqs/`

Шаг 5: Выравнивание каждого гена (MAFFT)

```
python3 scripts/align_genes_mafft.py
```

Что делает:

- Использует **MAFFT** для выравнивания последовательностей каждого гена между всеми геномами.

Вход: Последовательности из шага 4

Выход: Выравнивания в `results/aligned_genes/`

Шаг 6: Объединение выравниваний (Python)

```
python3 scripts/concatenate_alignments.py
```

Что делает:

- Объединяет выравнивания всех генов в единый супермассив, пригодный для построения дерева.

Вход: Выравнивания из шага 5

Выход: FASTA-файл супермассива `results/concatenated/supermatrix.faa`

Шаг 7: Построение дерева (FastTree)

```
python3 scripts/run_fasttree.py
```

Что делает:

- Использует **FastTree** для построения филогенетического дерева на основе объединённого выравнивания.

Вход: `supermatrix.faa` из шага 6

Выход: Дерево в формате Newick `results/tree/tree.nwk`