# DirectRM

## 0 Brief introduction

DirectRM allows simultaneous detection of the six most abundant modifications, namely ac4C, m1A, m5C, m7G, m6A and pseudourdine, from nanopore direct RNA sequencing data. It employs a two-stage detection framework:

- De novo detection: identification of potential modified kmers regardless of their modification types and positions.
- Modification type(s) and position(s) inference: the modification class(es) and its (their) position(s) within the modified k-mers are determined with an attention-based multi-instance multi-label framework.

It can confidently applied to diverse biological contexts and provide evaluable information to the complex epitranscriptome.

## 1 Features

- Support both SQK-RNA002 and newly introduced SQK-RNA004 sequencing data.
- Deep learning-based, can deal with data of high complexity (e.g., SQK-RNA004 data).
- Can provide read-level, transcriptome-level, or genome level modification profiles.
- Optimized data processing pipeline.
    - Users could analyze data in parallel batches, which significantly reducing memory and computational demands.
    - Less time and storage space required.

## 2 Before running DirectRM

### Platforms

All processes related to DirectRM were tested on

- Platform: Linux x86_64
- GPU: Nvidia GPUs
- CPUs

### Softwares

- Dorado: https://github.com/nanoporetech/dorado/tree/release-v0.6
- Remora: https://github.com/nanoporetech/remora
- Pytorch
- Python3

- Basic python packages: `math`, `os`, `re`, `sys`, `csv`, `json`, `argparse`, `tqdm`, `numpy`, `pandas`, `random`

## Input data

- **Raw data in Pod5 format**

  Please make sure your data are in Pod5 format. If your data is in Fast5 format, please use the python module pod5 (https://pod5-file-format.readthedocs.io/en/latest/) to convert it to Pod5.

  ```
  pod5 convert fast5 ./input/*.fast5 --output outputs/
  ```

- **Reference sequence**

- **Target regions**

  Please provide a **CSV** file of regions you want analyze. The table should contain following columns, and each line represents one region. The coordinates should be consistent to above reference file.

| seqnames | start | end | width | strand |
|---|---|---|---|---|
| chr1 | 207 | 307 | 101 | + |
| chr4 | 620 | 987 | 368 | + |

# 3 Getting Started

## 3.1 Base calling and alignment

**Dorado** is used to perform base calling and alignment simultaneously.

To optimize the memory usage, we split the total Pod5 directory into multiple parallel batches and perform base calling as several independent and parallel tasks. We prepared a script called `preprocessing.py`, which will:

1. Split the original Pod5 folder into multiple folders, and each contain 20 Pod5 files (denoted as **splitN**).
2. Perform base calling and alignment for each split
3. Compress, sort, and index the alignment file.
4. Delete the intermediate SAM files to save space.

```
python3 ./DirectRM/scripts/preprocessing.py \
-i <pod5_dir> \
--new_dir <new_pod5_dir> \
-o <bam_dir> \
--dorado <path to dorado software> \
--ref <reference file> \
--model <path to base calling model> \
--device <device>
```

### Parameter explaination

- `-i`: the original pod5 folder
- `--new_dir`: the script will randomly move 20 Pod4 files to a new folder (named as **splitN**) under <new_pod5_dir> folder.
- `-o`: output directory for the base calling and alignment results
- `--ref`: reference.fa
- `--dorado`: path to the dorado package. For example, `./software/dorado-0.6.2-linux-x64/bin/dorado`

```
dorado download --model all
```

- `--model`: path to the dorado model used for base calling.

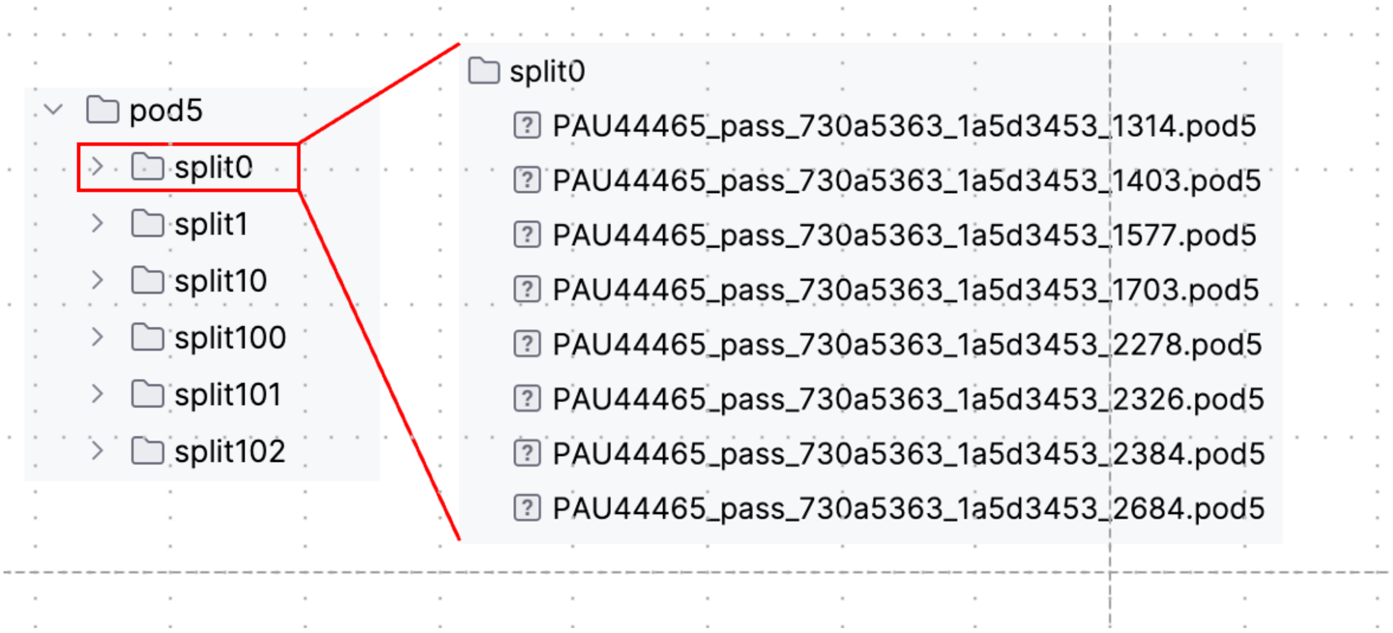
  Please download available Dorado models with:

  ```
  dorado download --model all
  ```

  - `rna002_70bps_hac@v3` for data sequenced with SQK-RNA002 kit,
  - `rna004_130bps_hac@v3.0.1` for data sequenced with SQK-RNA004 kit.
- `--device`: device id. For example, `cuda:0`, `cuda:all`

## Output

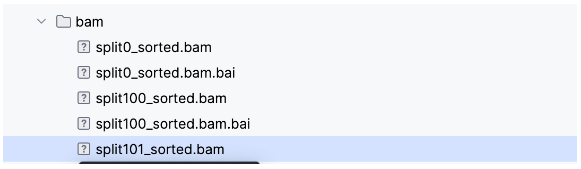
1. Organized new_pod5_dir



```
pod5
  split0
  split1
  split10
  split100
  split101
  split102

split0
  PAU44465_pass_730a5363_1a5d3453_1314.pod5
  PAU44465_pass_730a5363_1a5d3453_1403.pod5
  PAU44465_pass_730a5363_1a5d3453_1577.pod5
  PAU44465_pass_730a5363_1a5d3453_1703.pod5
  PAU44465_pass_730a5363_1a5d3453_2278.pod5
  PAU44465_pass_730a5363_1a5d3453_2326.pod5
  PAU44465_pass_730a5363_1a5d3453_2384.pod5
  PAU44465_pass_730a5363_1a5d3453_2684.pod5
```

2. Bam files



```
bam
  split0_sorted.bam
  split0_sorted.bam.bai
  split100_sorted.bam
  split100_sorted.bam.bai
  split101_sorted.bam
```

## 3.2 Feature extraction

**Remora** is used to extract features from above generated BAM file and Pod5 file. Remora will take the Pod5 file and corresponding bam file as input to form a Remora object. Then, signal refinement (re-squiggle) will be performed to correct the signal data to the reference sequence. Three groups of features will be extracted from the Remora object:

1. kmer sequence: the 9-mer or 5-mer sequence will be extracted and encoded with One-Hot encoding
2. Signal features: for each nucleotide within the kmer, the minimum, maximum, 1st and 3rd quartiles, mean, median, standard deviation, and length of signal events will be extracted.
3. Base call error features: mismatch, insertion, deletion, and quality.

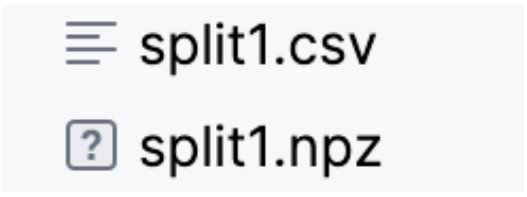
**Note**: To use remora, we recommend to create an independent conda environment with **python=3.9** for it.

```
python3 ./DirectRM/scripts/feature_extraction.py \
--pod5_dir <new_pod5_dir> \
--bam <bam_dir> \
--reg <interested_regions> \
--level <kmer_level_tables> \
-o <feature_dir> \
--splits 0 10 \
--kmer 9 --step 5
```

### Parameter explaination

- `--pod5_dir`: organized new_pod5_dir
- `--bam`: path to bam dir
- `--reg`: regions want to analyze
- `--level`: the k-mers levels table. `./DirectRM/5mer_levels_v1.txt` for data sequenced with SQK-RNA002 kit, `./DirectRM/9mer_levels_v1.txt` for data sequenced with SQK-RNA004 kit.
- `-o`: output dir for extracted features
- `--splits`: ids of **splits** for feature extraction. We have two ways for specifying the split
  - provide the start number and end number (two numbers): e.g., `0 10` refers to split0, split1, split2, ..., split10
  - provide the number of ids: e.g., `0 1 10 20 30` refers to split0, split1, split10, split20 and split30
- `--step`, `--kmer`: it will extract kmer features with a sliding window. `--kmer` specify the window size [5, 9], `--step` specify the step size of the sliding window.

### Output


split1.csv
split1.npz

1. CSV files: coordinates of kmers

| read_id | seqnames | start | end | width | strand |
|---|---|---|---|---|---|
| 4377a3e0-895e-4ee4-8eda-872a03e868aa | chr3 | 186788257 | 186788266 | 9 | + |
| 4377a3e0-895e-4ee4-8eda-872a03e868aa | chr3 | 186788262 | 186788271 | 9 | + |

2. npz files: features of kmers
    1. seq: one hot encoded sequence
    2. stat: mean, median, and etc of signal events
    3. level: expected kmer levels
    4. bse: base call errors

## 3.3 De novo modification detection

This step aims to detect modified kmers

```
python3 ./DirectRM/scripts/denovo_inference.py \
--feature_dir <feature_dir> \
--model_path <denovo_model>\
--splits 0 10 \
--device cuda:0
```

### Available models

We provided three binary de novo modification models, each model was trained with different label sources:

- **id1**: Wilcoxon test was used to compare the observed kmer signals and expected levels. K-mers with P-value < 0.01 (significantly differentiated) were labeled as positive, and vice versa.
- **id2**: K-mers with base call error frequence > 0 were labeled as positive, and vice versa.
- **id3 (recommended)**: Intersection of above two.

### Parameter explaination

- `--model_path`: path to de novo detection model
    - `./DirectRM/model/RNA004/id3_binary/model.pt` for data sequenced with SQK-RNA004 kit
    - `./DirectRM/model/RNA002/id3_binary/model.pt` for data sequenced with SQK-RNA002 kit
- `--device`: device used for de novo detection

### Output

npy file speficy the probability of being modified

split0_mod.npy

## 3.4 Modification type and position inference

This step aims to identify the modification type(s) and its(their) position within the modified kmers.

```
python3 ./DirectRM/scripts/inference.py \
--feature_dir <feature_dir> \
--output_dir <preds_dir> \
--device cuda:0 --splits 0 10 \
--config <path to configuration file> \
--ml True --model_id 1
```

### Available models

We provided four model artchitecture:

- **model1**: {attention} + {fcnn}
- **model2**: {attention + LSTM} + {fcnn}
- **model3**: {attention + positional encoding} + {fcnn}
- **model4**: {attention + positional encoding + LSTM} + {fcnn}

### Parameter explaination

- `--config`: please provide json file `.json` as following, which specify the path to integrated or the six independent models.

```
{
  "model1":{
    "integrated": "./DirectRM/model/RNA004/ml1/model.pt",
    "ac4c": "./DirectRM/model/RNA004/ac4c_m1/model.pt",
    "m1a": "./DirectRM/model/RNA004/m1a_m1/model.pt",
    "m5c": "./DirectRM/model/RNA004/m5c_m1/model.pt",
    "m6a": "./DirectRM/model/RNA004/m6a_m1/model.pt",
    "m7g": "./DirectRM/model/RNA004/m7g_m1/model.pt",
    "psi": "./DirectRM/model/RNA004/psi_m1/model.pt"
  },
  "model2": {
    "integrated": "./DirectRM/model/RNA004/ml2/model.pt",
    "ac4c": "./DirectRM/model/RNA004/ac4c_m2/model.pt",
    "m1a": "./DirectRM/model/RNA004/m1a_m2/model.pt",
    "m5c": "./DirectRM/model/RNA004/m5c_m2/model.pt",
    "m6a": "./DirectRM/model/RNA004/m6a_m2/model.pt",
    "m7g": "./DirectRM/model/RNA004/m7g_m2/model.pt",
    "psi": "./DirectRM/model/RNA004/psi_m2/model.pt"
  }
}
```

- `--ml`: whether to use integrated detection model (`True`) or independent detection model (`False`)
- `--model_id`: id of model structure

### Output

Read level prediction results for each class, grouped by modification type and chromsome/transcripts

| read_id | seqnames | pos(istion) | strand | ac4c(_probability) |
|---|---|---|---|---|
| 00172a9f-1aab-4044-a20d-87cc753ba3d0 | chr1 | 39004867 | + | 0.3917313 |
| 00172a9f-1aab-4044-a20d-87cc753ba3d0 | chr1 | 39004882 | + | 0.16477184 |

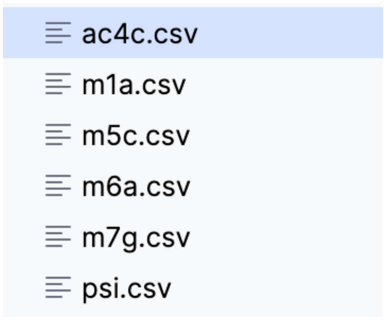
## 3.5 Read-level results to Site-level

This step aggregates the read-level results to provide site-level results

```
python3 ./DirectRM/scripts/read2site.py \
--indir <preds_dir> \
--outdir <preds_dir>
```

### Output

Site-level results for each modification class



ac4c.csv
m1a.csv
m5c.csv
m6a.csv
m7g.csv
psi.csv

| seqnames | pos(istion) | strand | max_prob | noisyor_prob | count | coverage |
|---|---|---|---|---|---|---|
| chr22 | 15787218 | + | 0.31133258 | 0.31133257 | 0 | 32 |
| chr22 | 15787232 | + | 0.16512871 | 0.16512871 | 0 | 35 |

## 4 Getting help

We appreciate your feedback and questions. You can report an error or suggestions related to DirectRM as an issue on github.