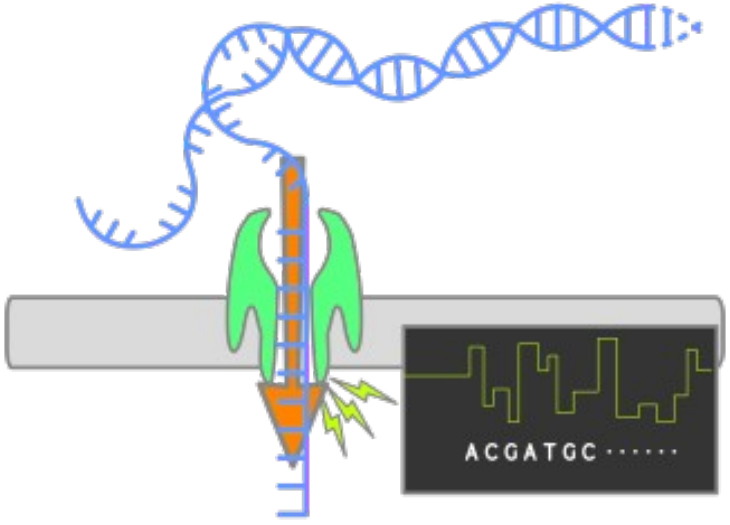
Monitoring and Controlling Nanopore Sequencing Runs

Master’s Thesis

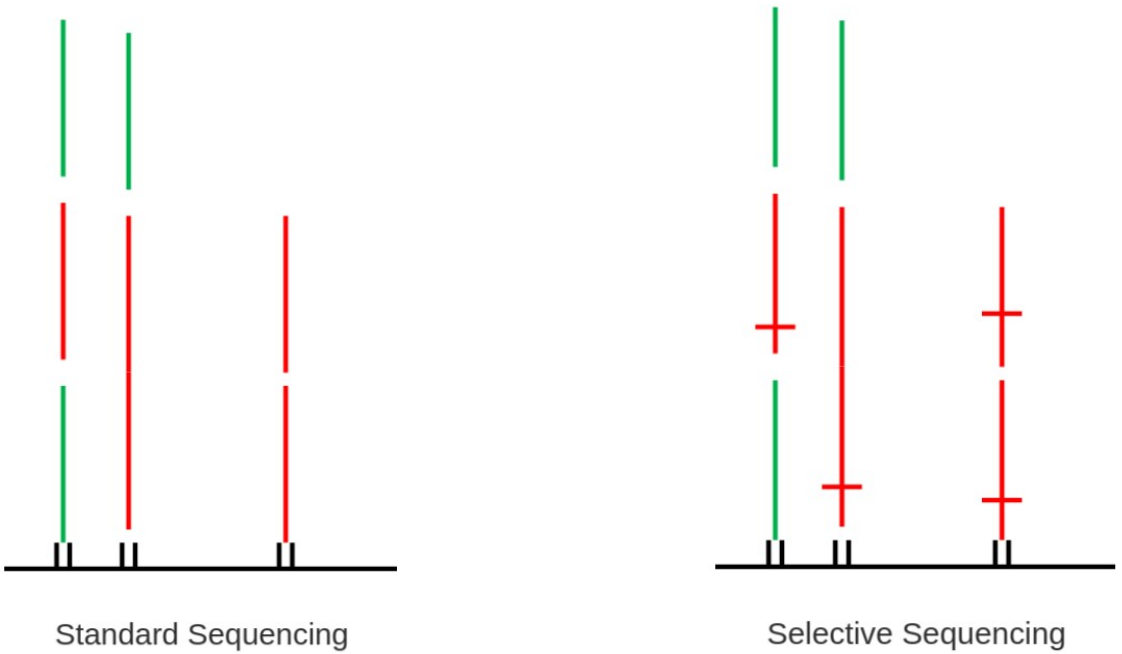
Author: Bc. Matej Fedor
Supervisor: doc. Mgr. Tomáš Vinař, PhD.

Nanopore Sequencing



Selective Sequencing

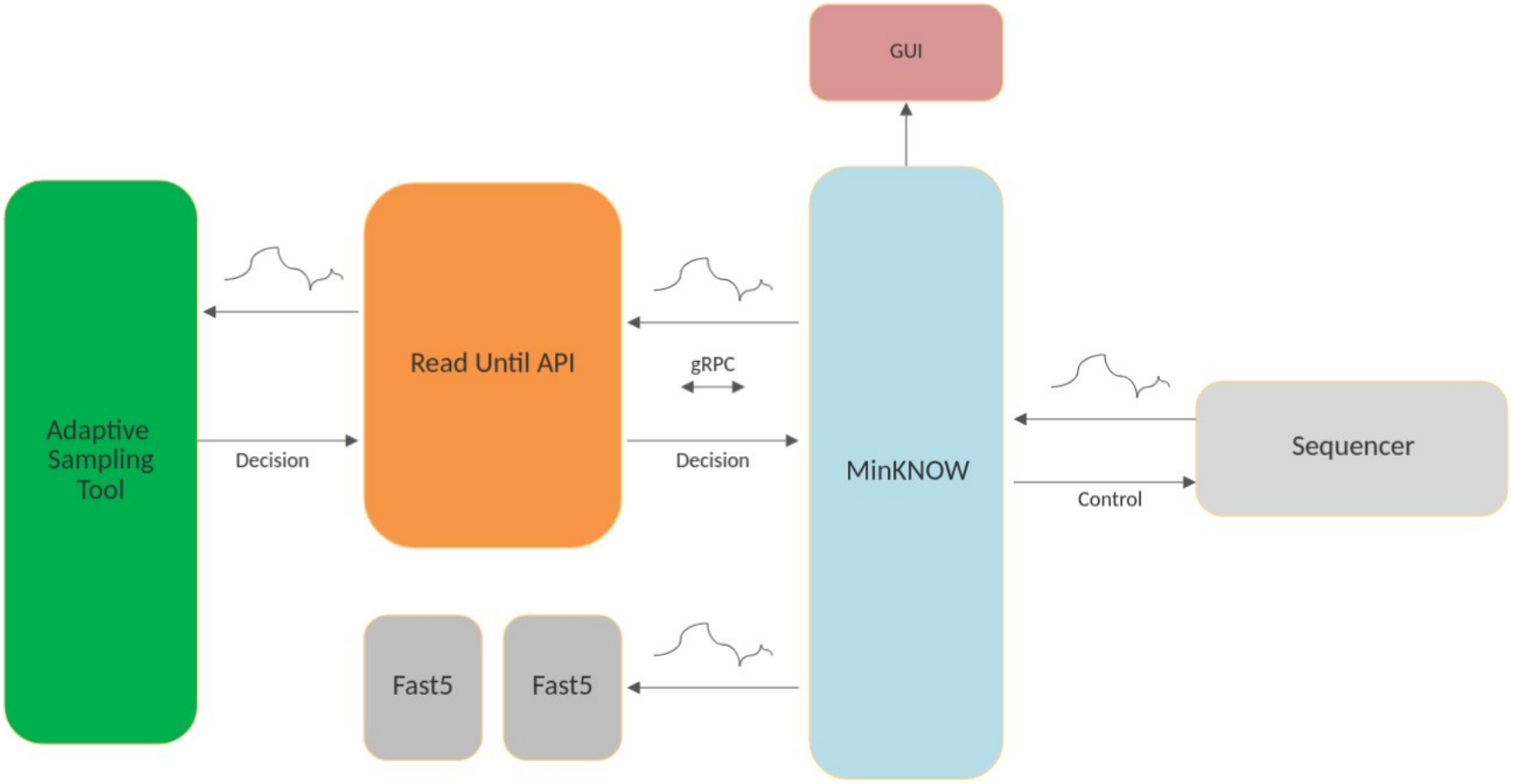
- User has access to the raw signal produced in fixed period of time in real time
- User has option to intervene during the sequencing run and decide a DNA sequence is rejected



Standard Sequencing

Selective Sequencing

Real Adaptive Sampling Setup



Overview

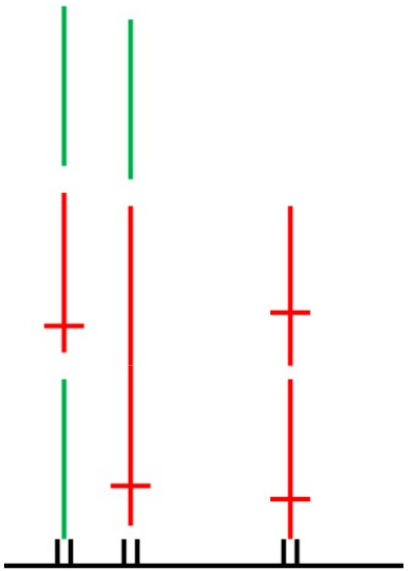
- We emulate selective sequencing runs to facilitate adaptive sampling method research
- We combine well-known adaptive sampling tool with the emulator to demonstrate its capabilities
- We develop our own adaptive sampling tool using the emulator

Adaptive Sampling

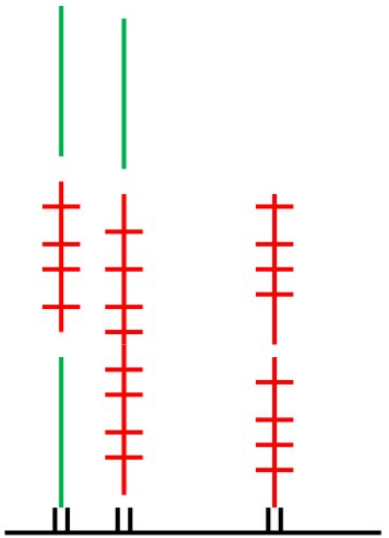
- Adaptive sampling tool development is expensive
  - Need for sequencing run using a physical sequencer to observe the adaptive sampling performance
  - Expertise in both the fields of biology and informatics is required
  - Emulation options are limited

Sequencing Emulators

- MinKNOW’s playback feature



Selective Sequencing

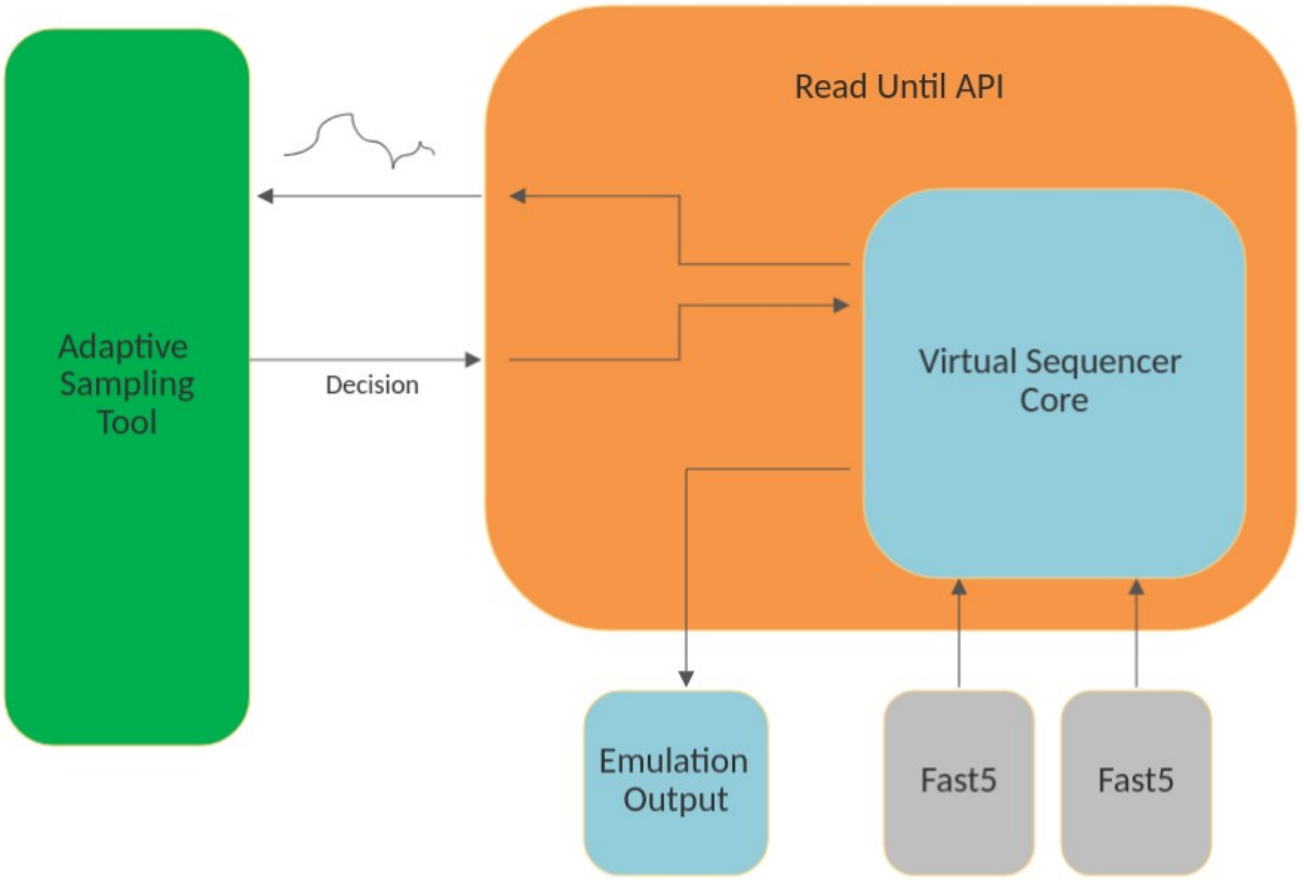


MinKNOW's playback feature

Virtual Sequencer

- We introduce emulator capable of emulating selective sequencing
- On read rejection
  - Emulates the ejection of DNA sequence and loading another one
  - Uses future read sequenced by the nanopore channel as a base for emulation
  - Preserves the DNA sequence distribution in the sample
  - Modifies the number of sequenced on-target/off-target bases
  - Allows to observe impacts on adaptive sampling performance through increased target genome coverage, e.g. inspect coverage details

Emulated Adaptive Sampling Setup



Implementation Challenges

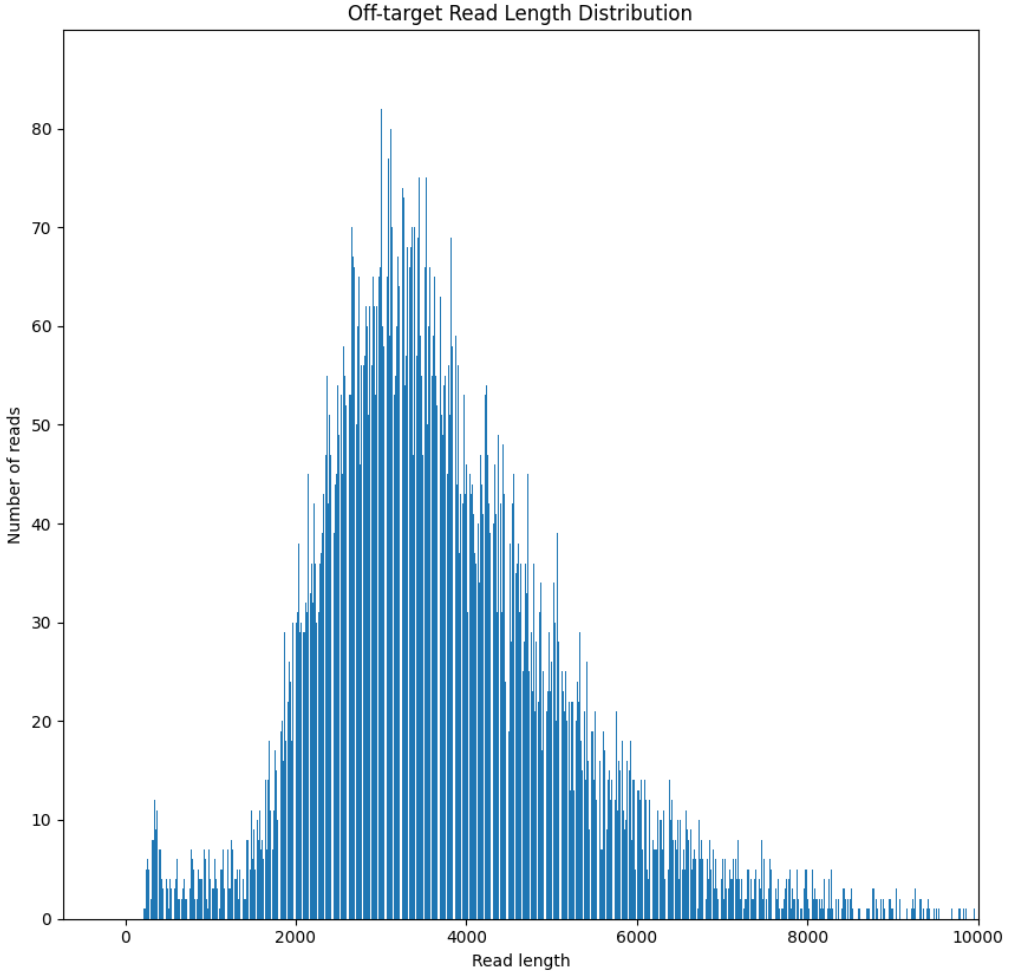
- Timing issues
  - Virtual Sequencer is a multi-threaded python application
  - Heavy load of rejection decisions can cause non-negligible latency
  - Preferring low unblock latency over scheduled start of sequencing
  - Emulation effectively slows down on slower platforms
- Lack of documentation
  - Data structures received from physical sequencer
  - Minimum obtainable chunk length
  - Meaning of obtained signal annotation

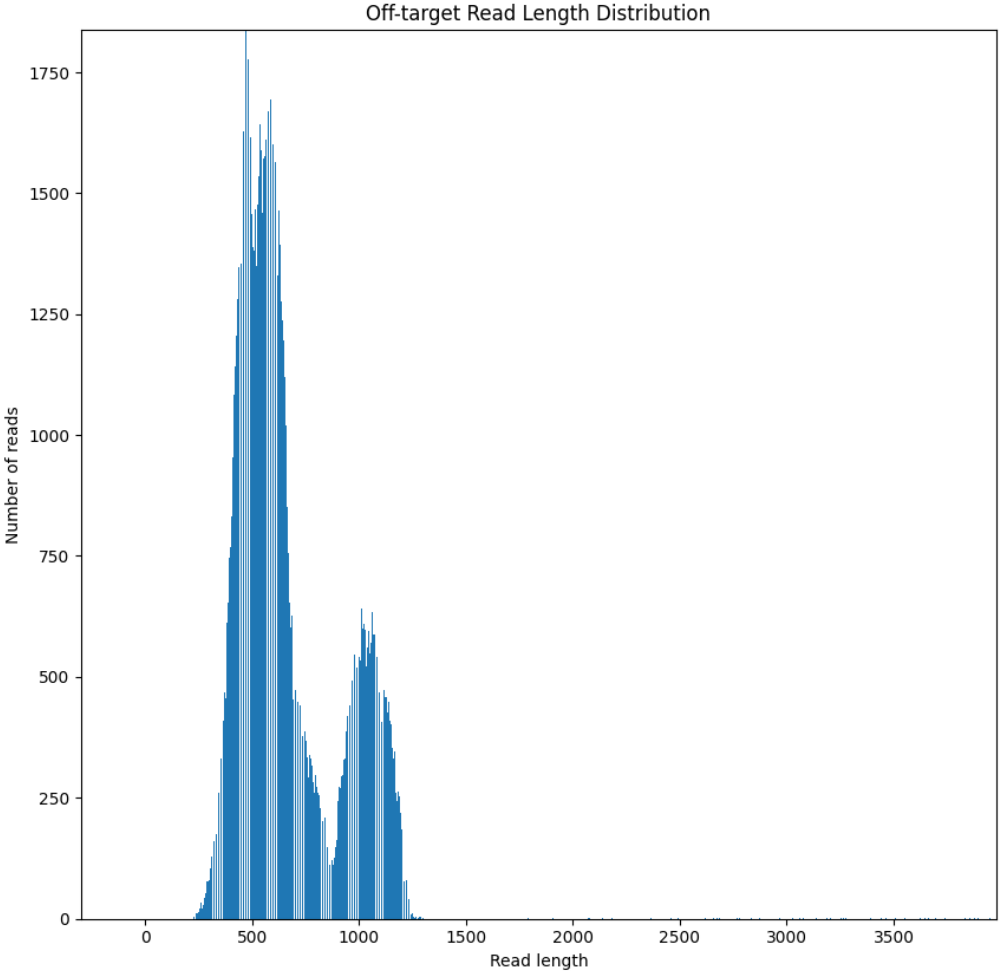
Readfish Integration

- We connect well-known adaptive sampling tool to the virtual sequencer
  - We demonstrate minor changes required to Readfish
  - We utilize the virtual sequencer to fine-tune Readfish configuration
  - We replicate published experiments conducted with Readfish

Payne et al. Nanopore adaptive sequencing for mixed samples, whole exome capture and targeted panels.

	Original Run	Adaptive Sampling Run
On-target Read Count	4310	14358
Off-target Read Count	189113	635277
On-target Avg. Length	3562.47b	3459.40b
Off-target Avg. Length	3725.40b	663.10b
On-target Bases	15.35M	49.67M
Off-target Bases	704.52M	421.26M
Absolute Enrichment	1.00x	3.24x


Off-target Read Length Distribution
Number of reads
Read length
0
10
20
30
40
50
60
70
80
0
2000
4000
6000
8000
10000


Off-target Read Length Distribution
Number of reads
Read length
0
250
500
750
1000
1250
1500
1750
0
500
1000
1500
2000
2500
3000
3500

Future Work

- Achieved 3.24x absolute enrichment is not completely realistic
  - Payne et al. report 1.6x absolute enrichment in a real experiment
- We do not model all aspects of selective sequencing run yet
  - The ejection speed of nanopore channel remains unknown
  - Failure rate of nanopore channels increases with the intensity of adaptive sampling

Machine Learning-Based Adaptive Sampling (1)

- We propose our own adaptive sampling method
  - We sacrifice the ability to adaptively sample arbitrary genome
  - We specifically sample SARS-CoV-2 from clinical sample
  - We skip basecalling step to save time and operate directly with the raw signal

Machine Learning-Based Adaptive Sampling (2)

- We designed a CNN model to classify read chunks
- Attempted by other authors but never tested in realistic sequencing run
- Only testing accuracy reported

```
cnn = Sequential()
cnn.add(Conv1D(filters=64, kernel_size=60, strides=7, activation='relu',
    padding='same', input_shape=input_shape))
cnn.add(Dropout(rate=0.1, seed=SEED))
cnn.add(Conv1D(filters=128, kernel_size=60, strides=7, activation='relu',
    padding='same'))
cnn.add(MaxPooling1D(pool_size=2))
cnn.add(Dropout(rate=0.1, seed=SEED))
cnn.add(Conv1D(filters=128, kernel_size=60, strides=7, activation='relu',
    padding='same'))
cnn.add(Flatten())
cnn.add(Dense(2, activation='softmax'))

learning_rate_schedule =
    keras.optimizers.schedules.ExponentialDecay(initial_learning_rate=0.0005,
    decay_steps=10_000, decay_rate=0.96, staircase=True)
```

Model Training

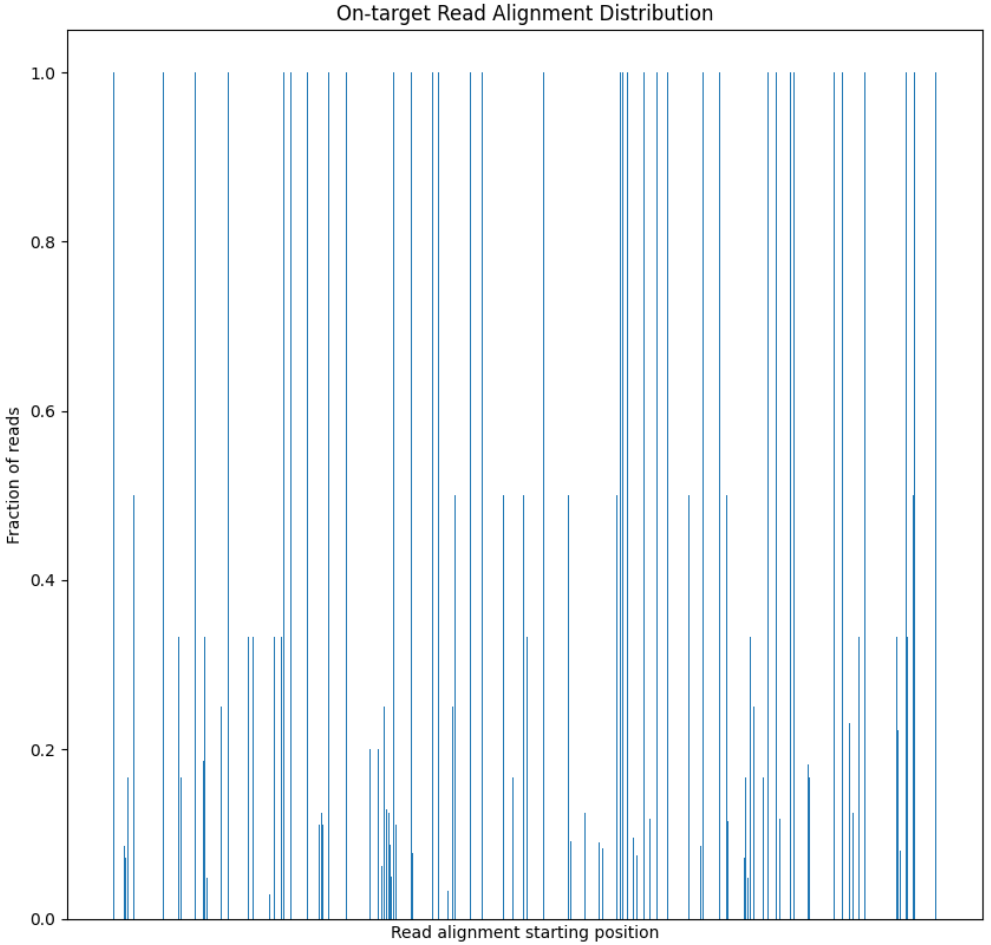
- Training dataset properties
  - Extracted from sequencing data of PCR amplified SARS-CoV-2 sample
  - Approximately balanced with emphasis on covering the entire target genome
  - ~800k SARS-CoV-2 examples, 1.6M examples overall, extracted in ~20 minutes
  - Model trained for ~2 hours
  - Testing accuracy of ~96%

Selectify

- We utilize the virtual sequencer to test Selectify in realistic scenario
  - We observe model deciding fast but being unable to classify regions not explicitly covered by training dataset

	Readfish	Selectify - 90%	Selectify - 75%
Sensitivity	98.78%	91.00%	88.87%
Specificity	37.90%	8.13%	10.23%

	Readfish	Selectify - 90%
Time per data chunk	4.45ms	2.68ms
Acceleration	1.00x	1.66x



- Reads mapping to some SARS-CoV-2 regions were systematically rejected
- Method is unable to generalize knowledge learnt from dataset
- Current version not suitable for samples with unknown composition

Conclusion

- We developed selective sequencing emulator that facilitates our adaptive sampling experiments
  - Future work can focus on improving its credibility
- We demonstrate use of the emulator when developing adaptive sampling tool
  - We achieve superior classification speed
  - Classification specificity needs to be improved by further research

Thank you for your attention!

Time for your questions