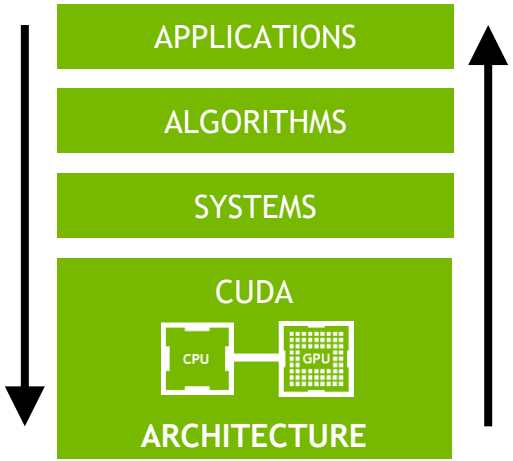
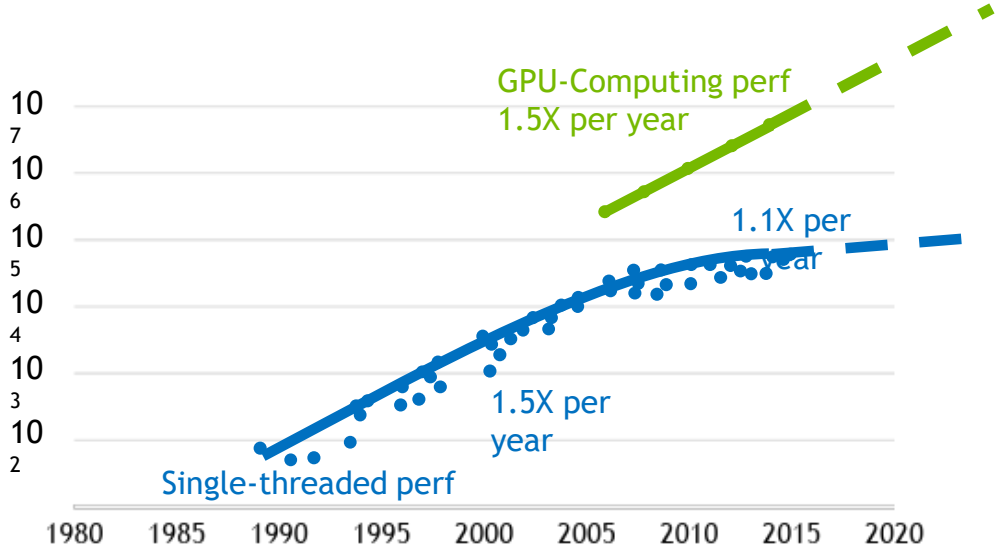
# Accelerating Sequencing with GPU Computing and Deep Learning

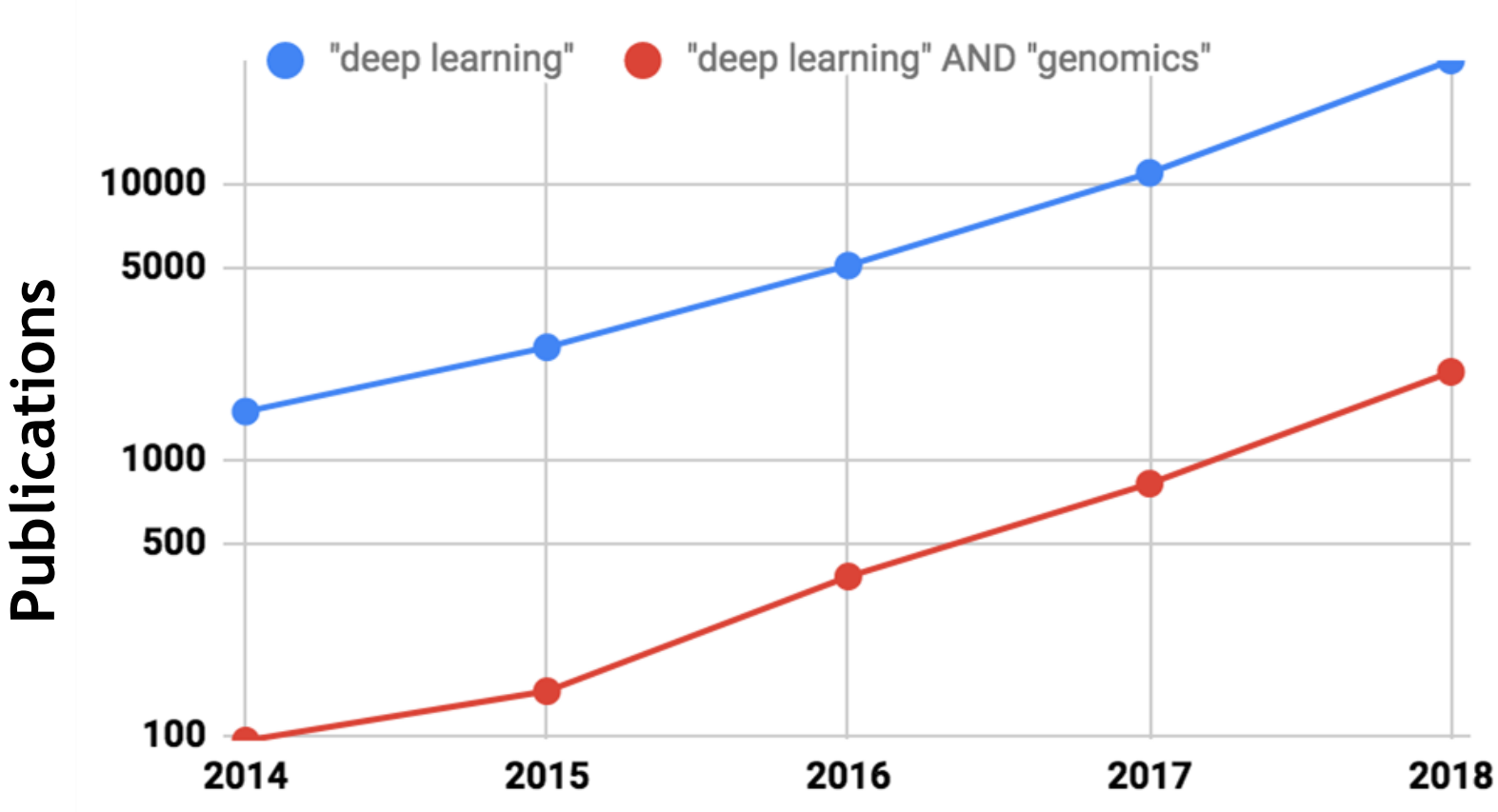
Johnny Israeli

# COMPUTE TRENDS



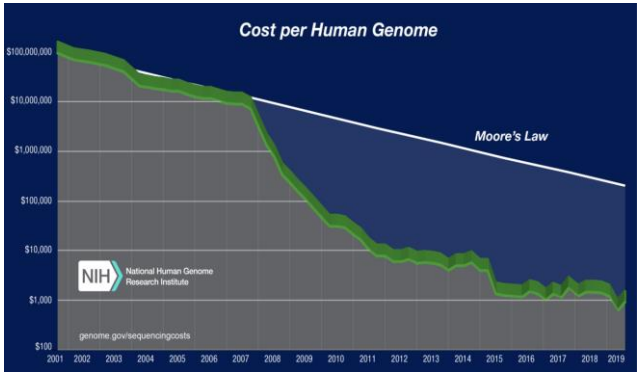
# COMPUTE TRENDS



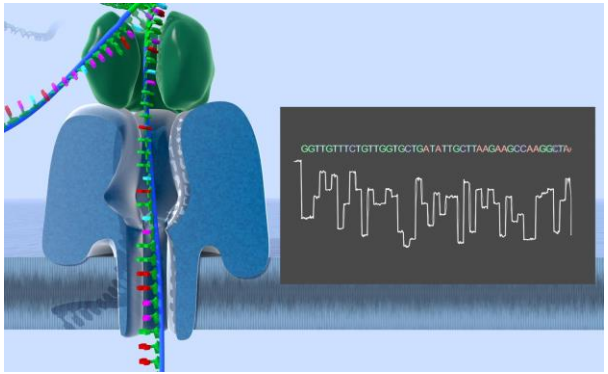

Sequencing Trends

# SEQUENCING TRENDS

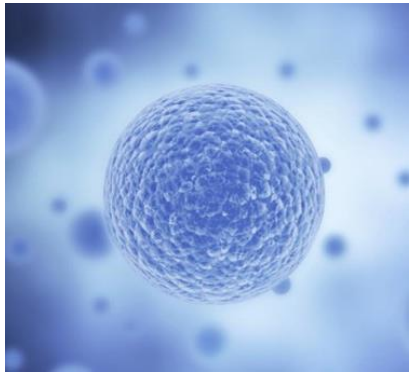
## Sequencing Data Growing in Volume and Complexity




Decreasing Cost

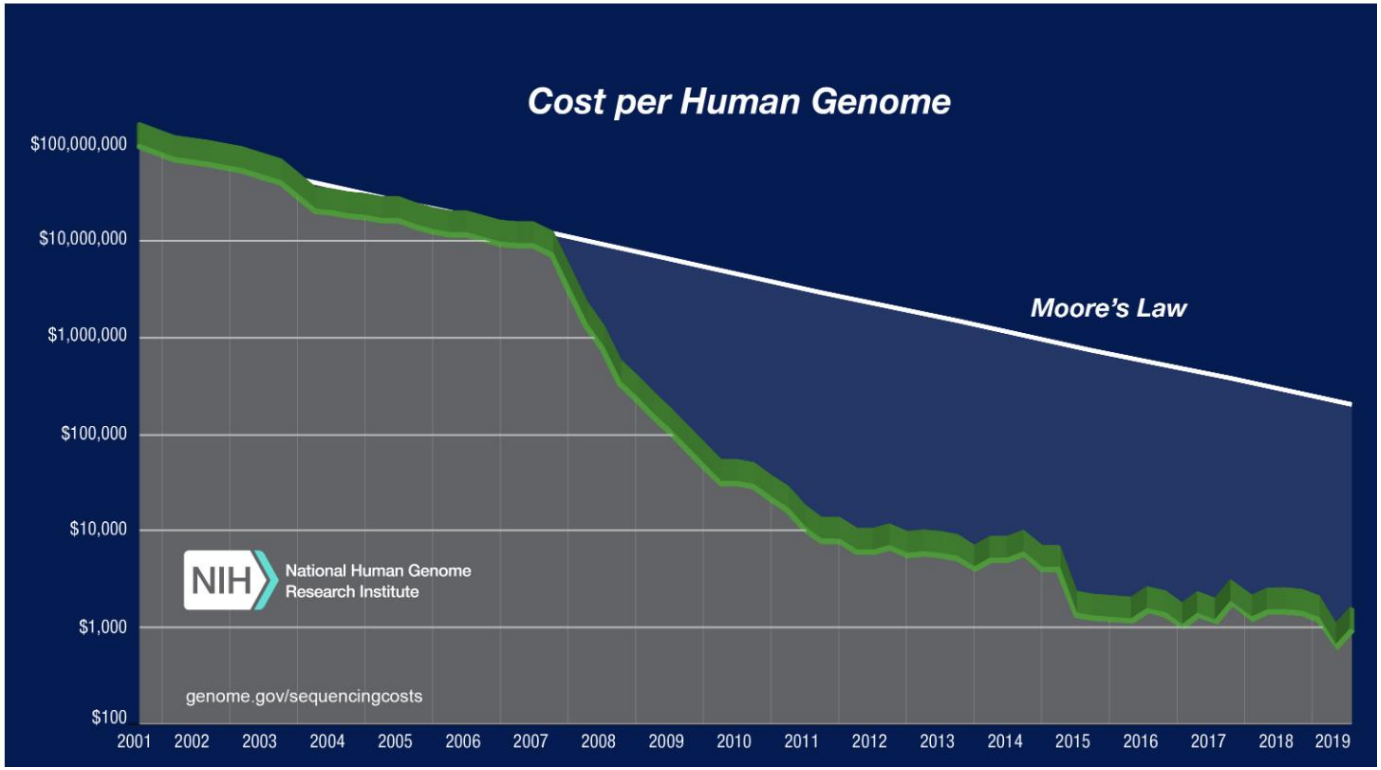



Increasing Read Length

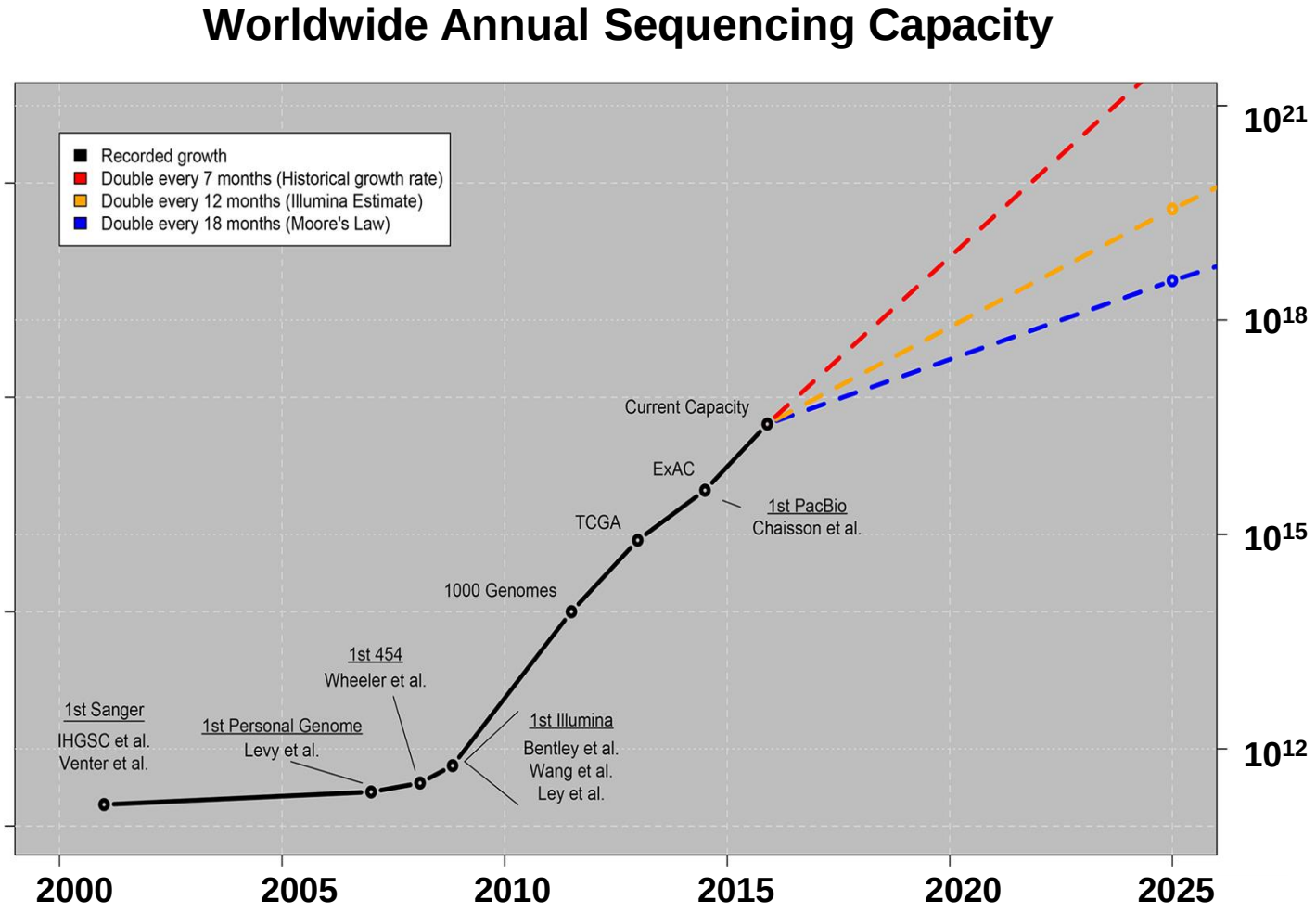


Rise of Single Cell Data

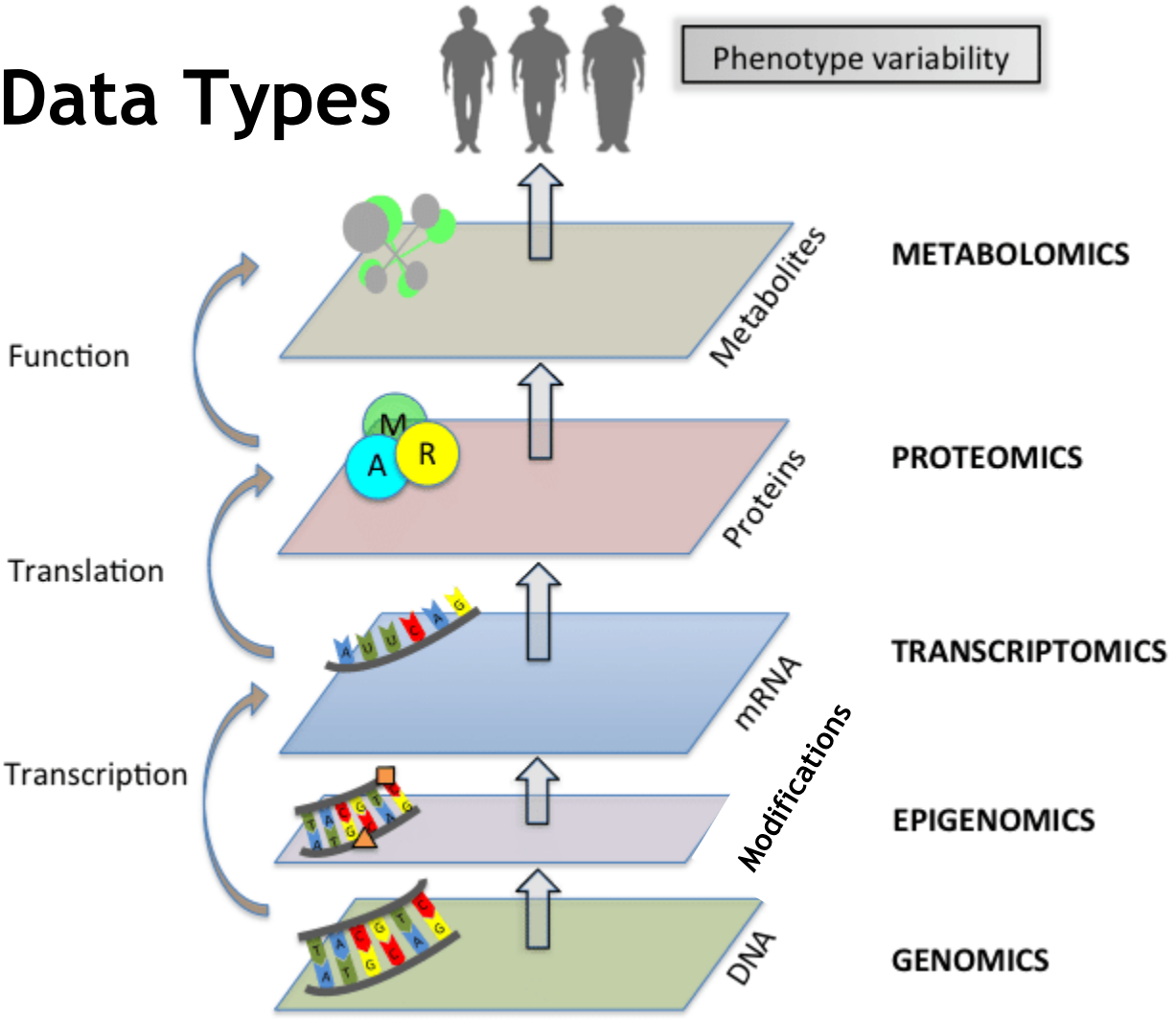
# SEQUENCING TRENDS



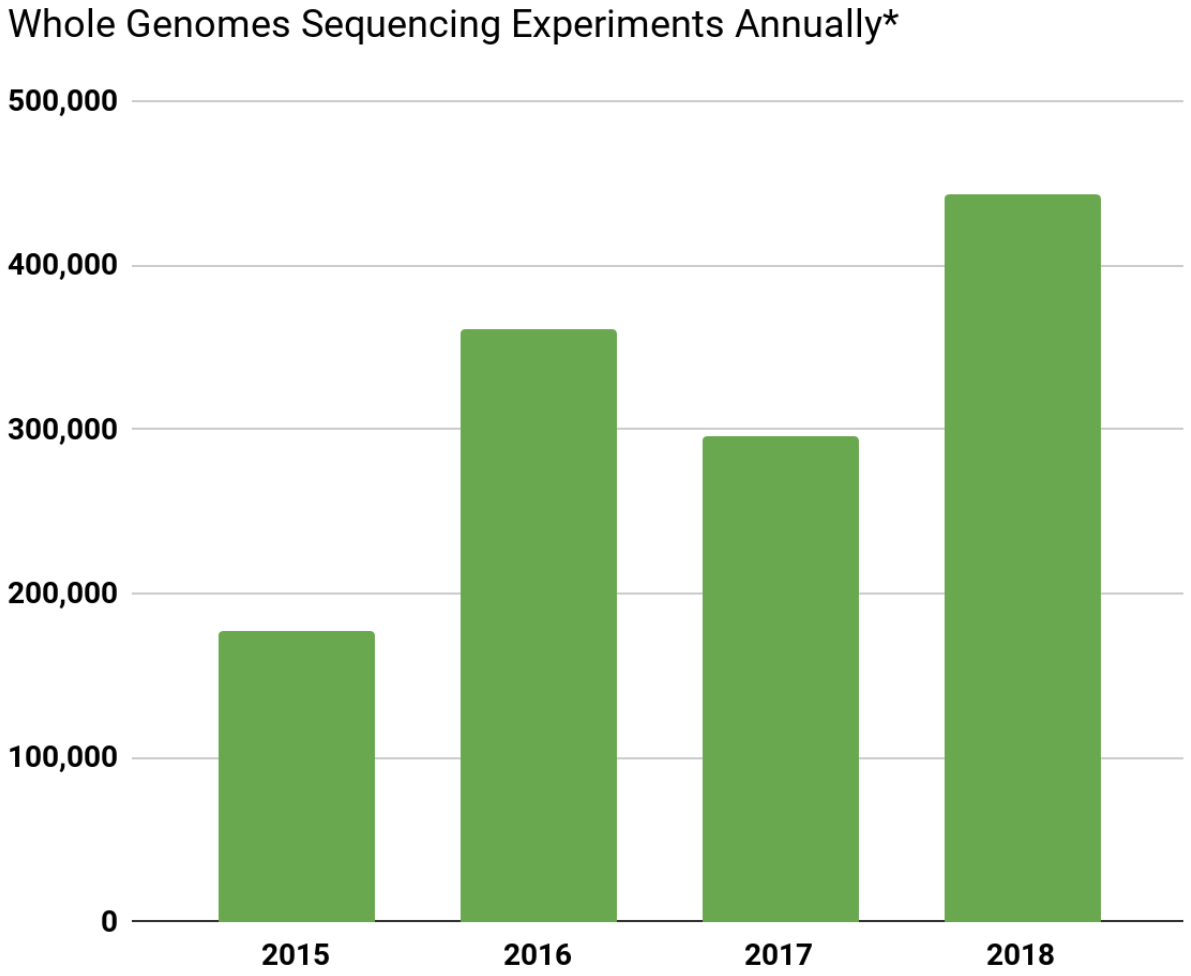

# SEQUENCING TRENDS



# Sequencing Data Types

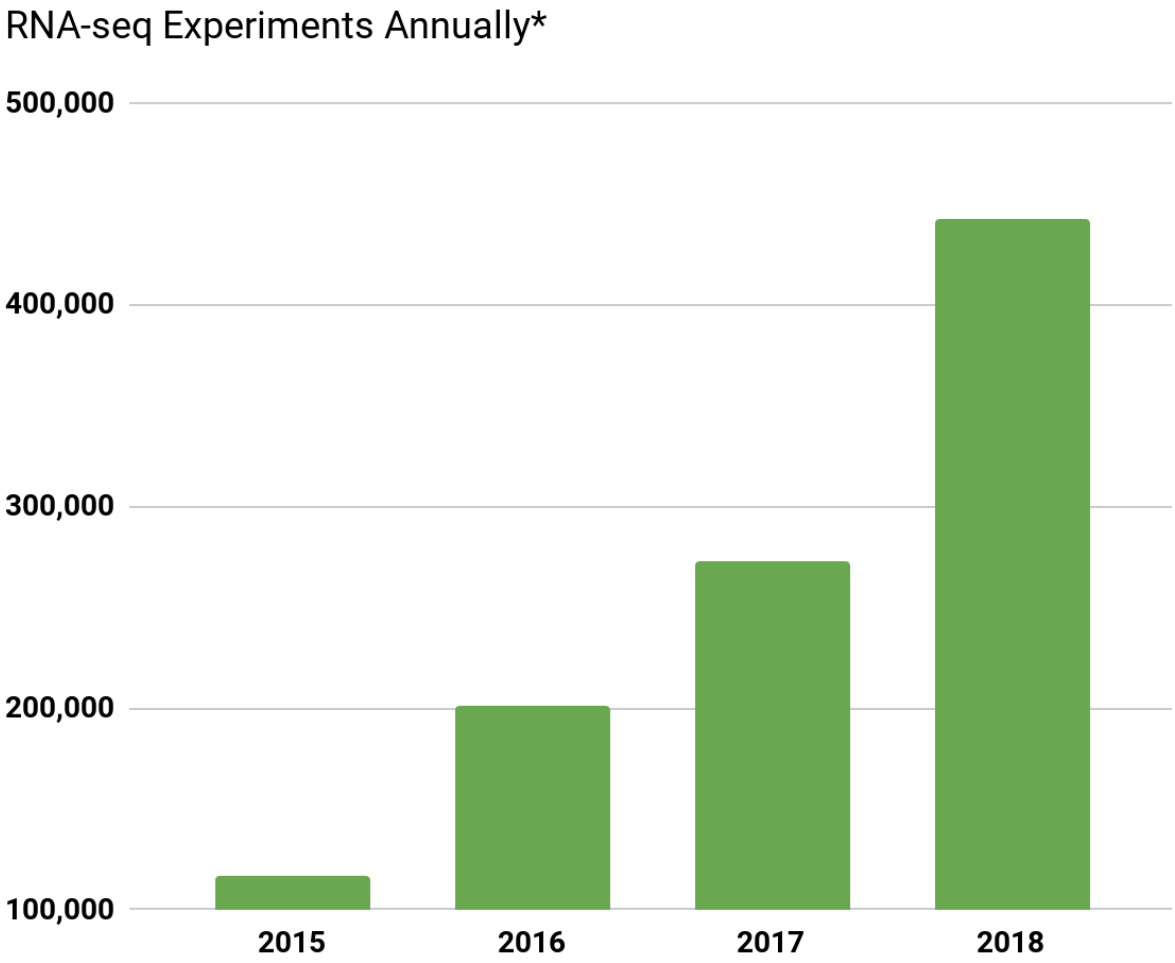


# SEQUENCING TRENDS: Genomics



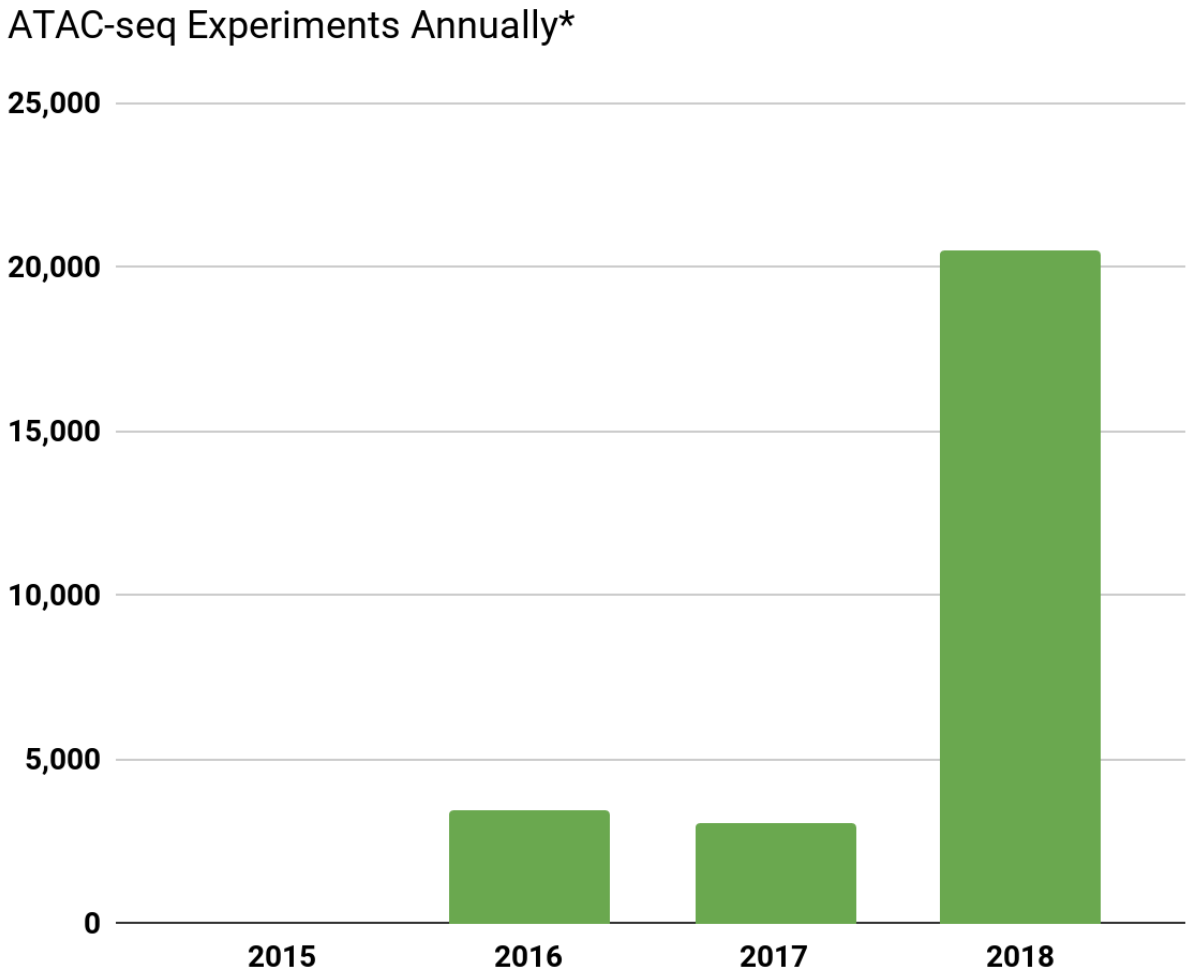

*ENA Database

# SEQUENCING TRENDS: Transcriptomics



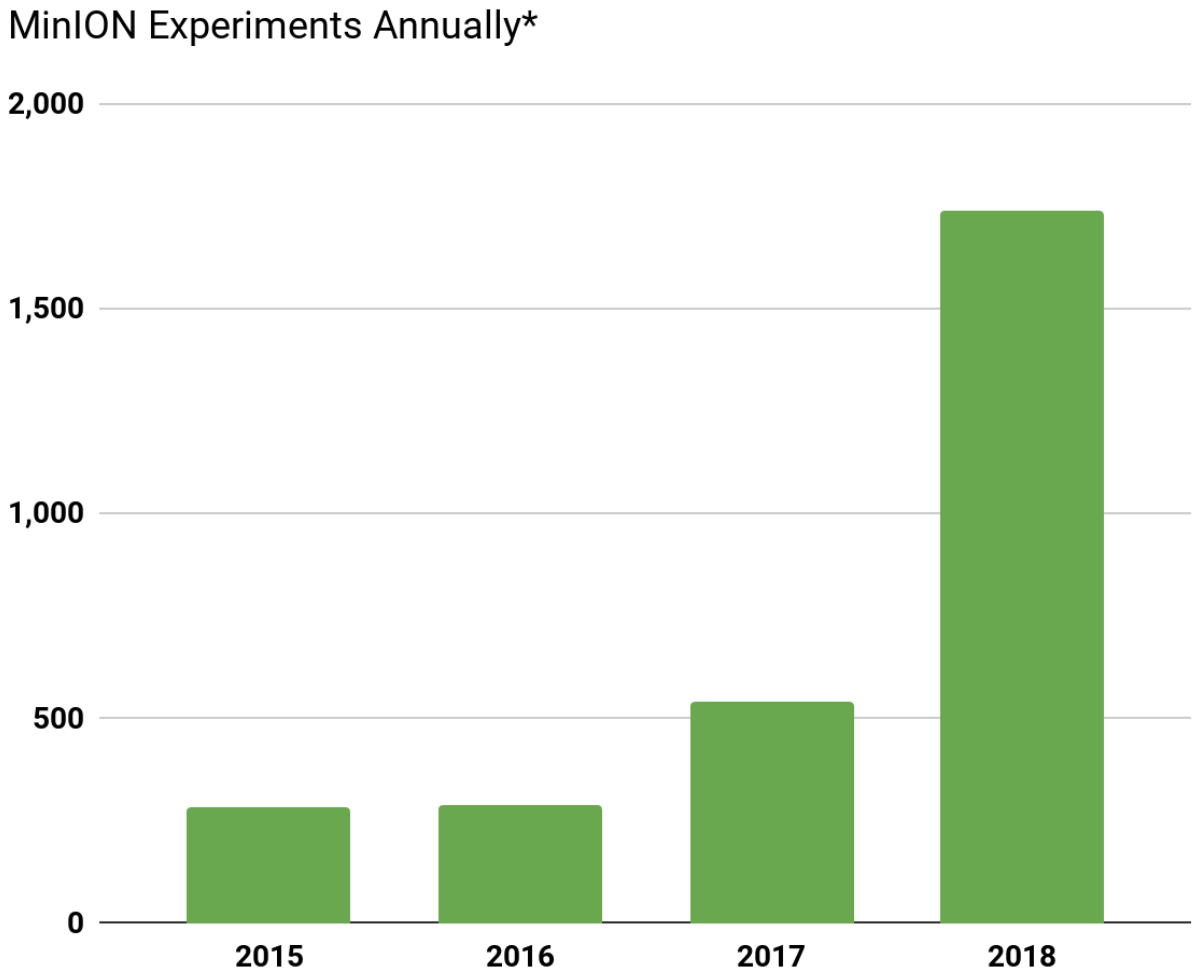

*ENA Database

# SEQUENCING TRENDS: Epigenomics




*ENA Database

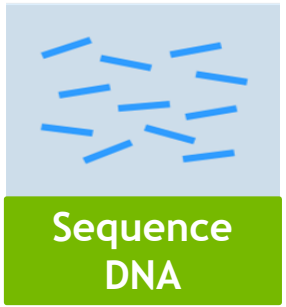
# SEQUENCING TRENDS: Nanopore Long Read Sequencing

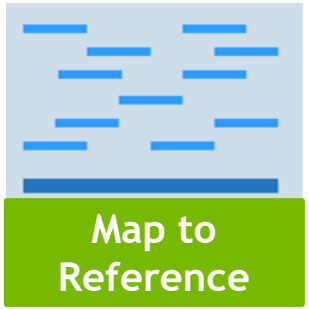



*ENA Database


Variant Calling

# Variant Calling



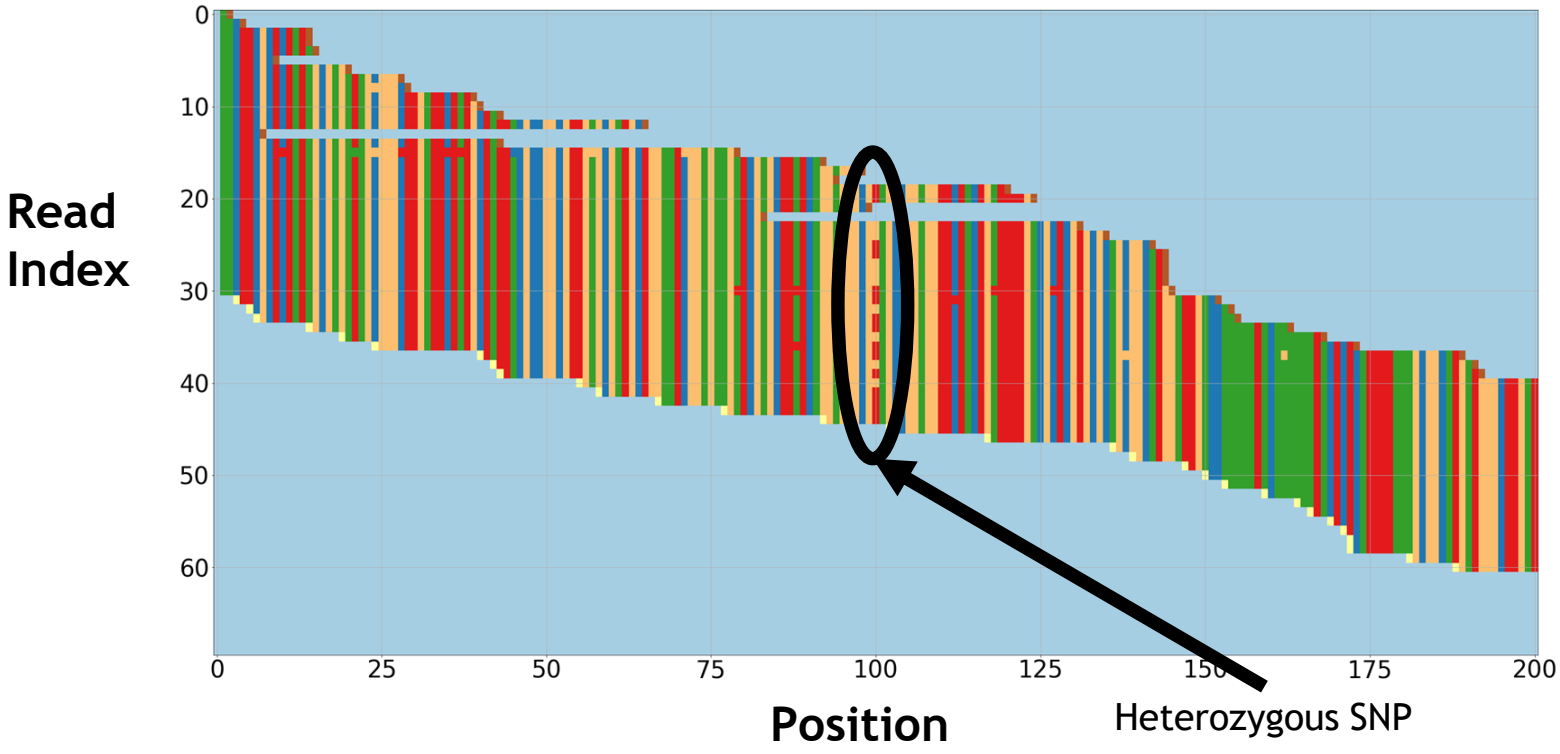

Reference TGGATTTGAAAACGGAGCAAATGACTG

Illumina Reads
TGGATTTGAAAACGGAGCAAATGACTG
TGGATTTGAAAACGGAGCAAATGACTG
TGGATTTGAAAACAGAGCAAATGACTG
TGGATTTGAAAACAGAGCAAATGACTG
TGGATTTGAAAACAGAGCAAATGACTG
TGGATTTGAAAACGGAGCAAATGACTG

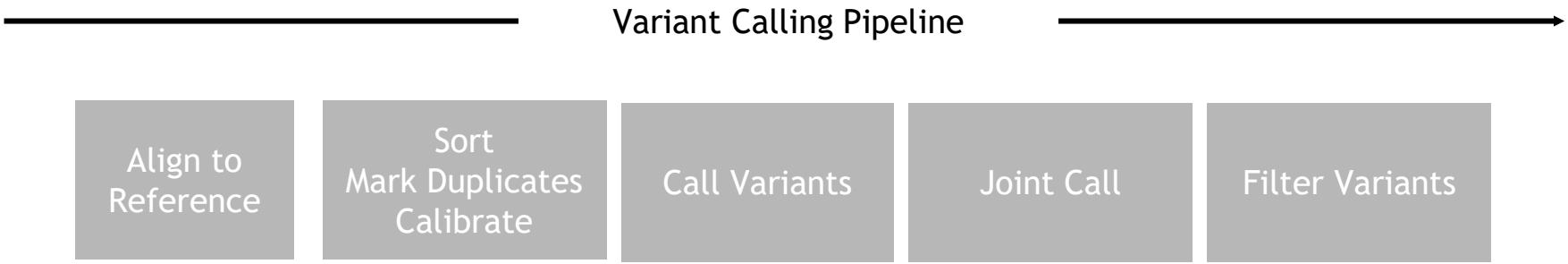
Likely heterozygous variant

- Identify sites with potential mismatch
- True variants or instrument errors?
- SNPs or insertions or deletions?
- Heterozygous or homozygous variants?

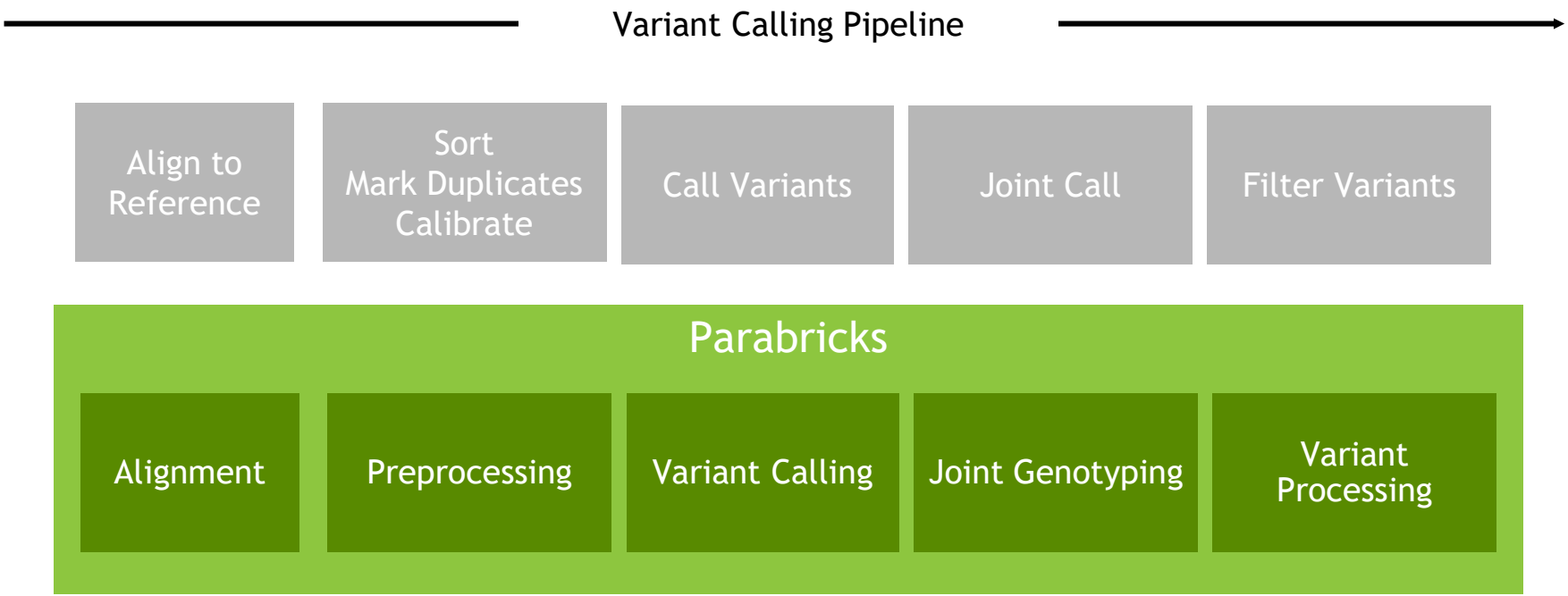
# Example Pileup Input Data



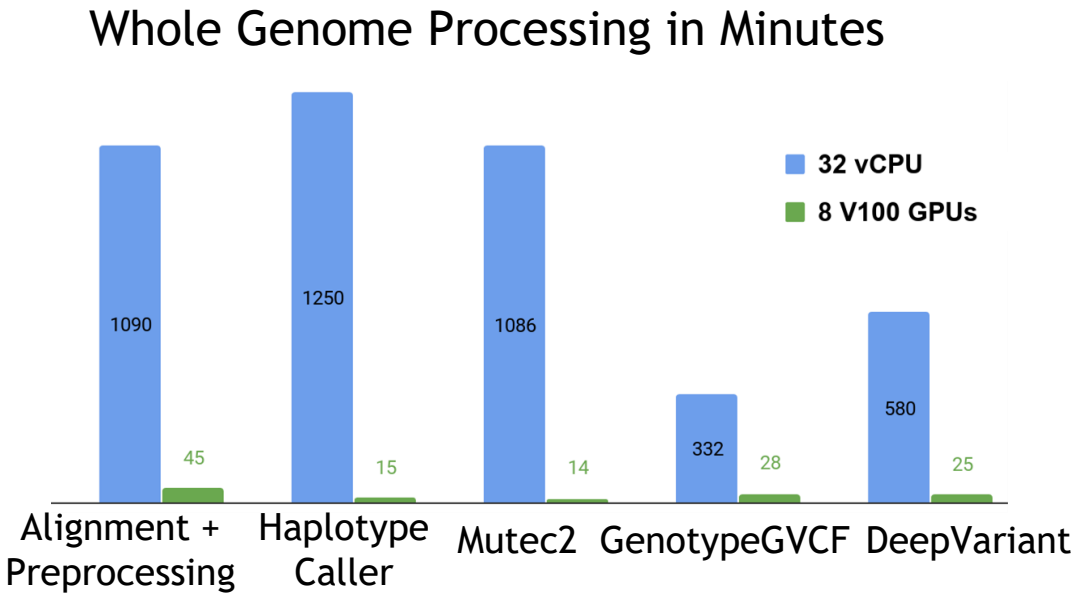
# GATK Variant Calling Pipeline

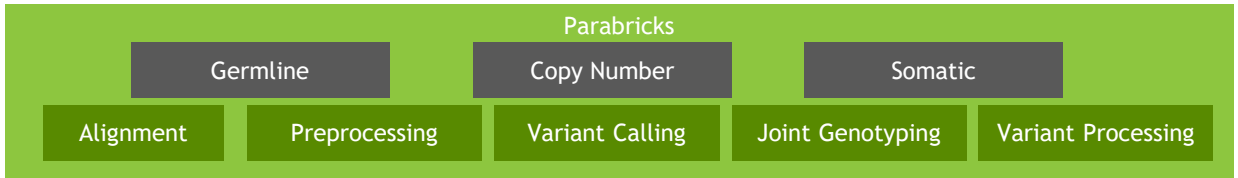


# Accelerated GATK Variant Calling Pipeline

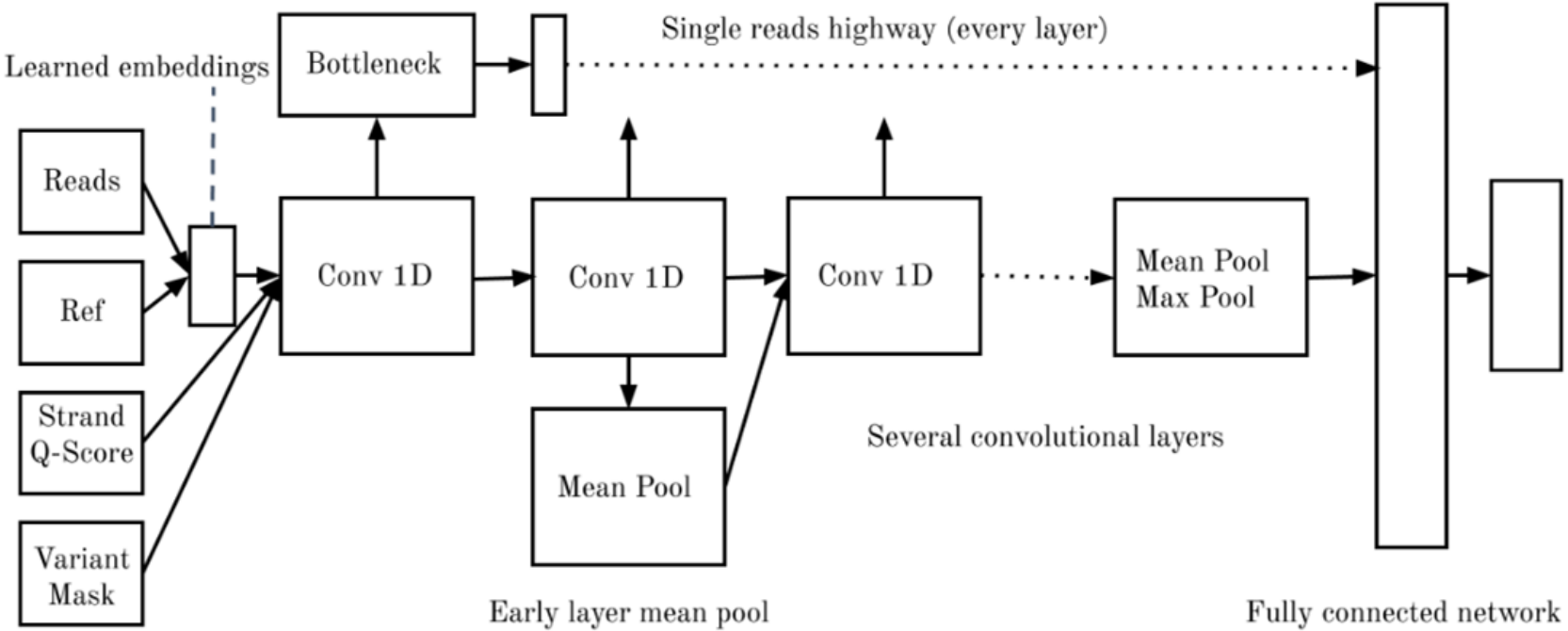


# Accelerated Variant Calling Pipelines



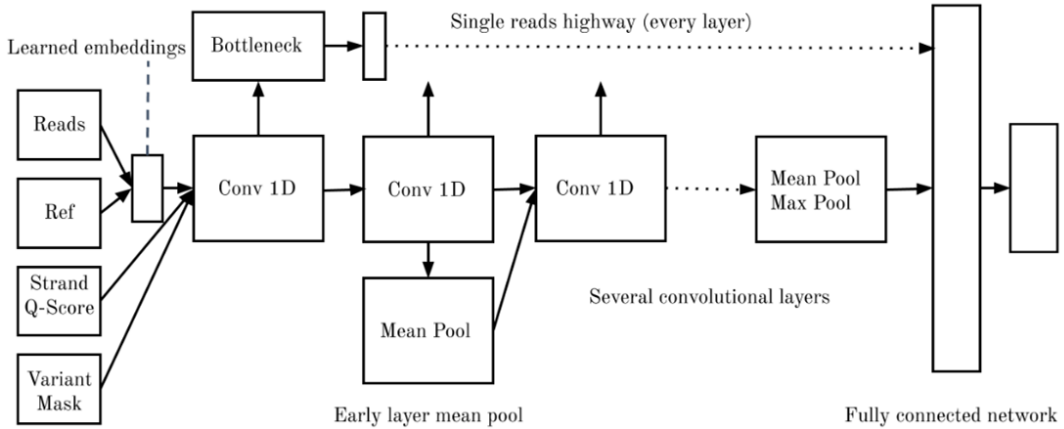



# Deep Averaging Network (DAN)

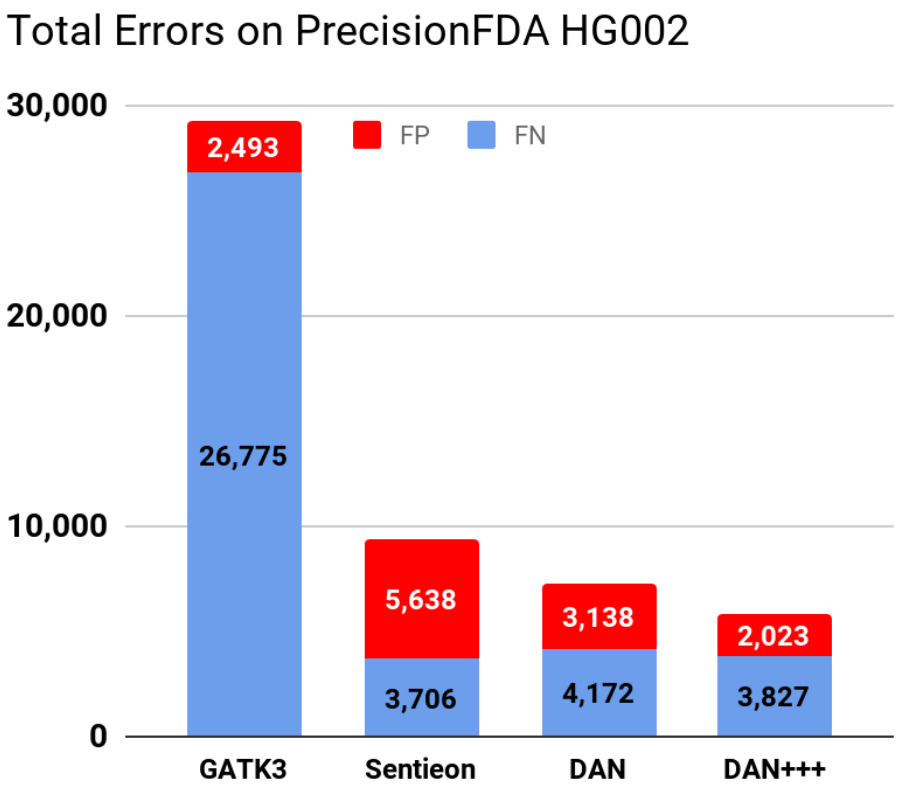


# DAN Development

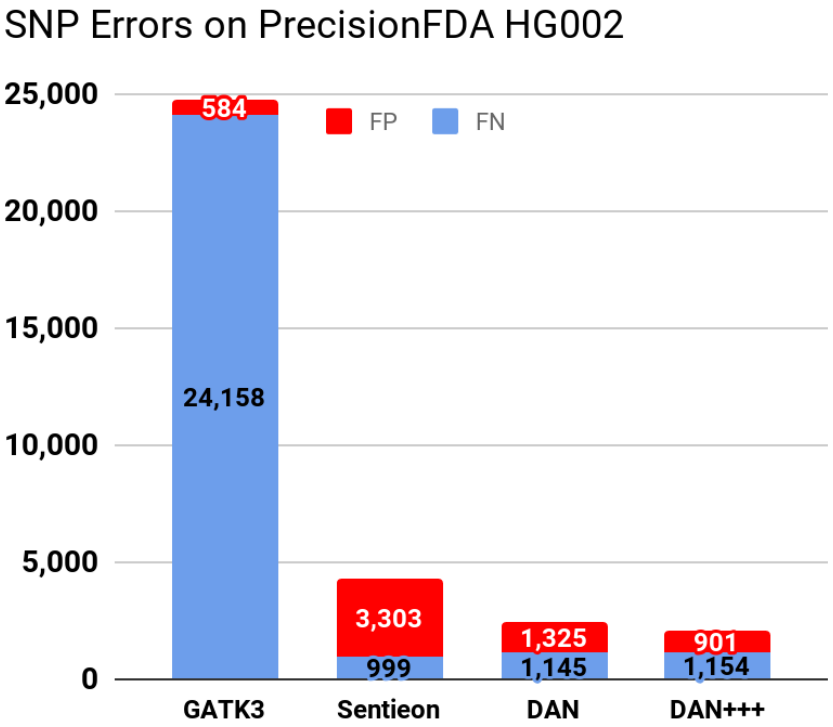
- PyTorch-based 1D model
- Learned embeddings of bases
- Encoding variant proposals
- Downsample easy variant candidates during training

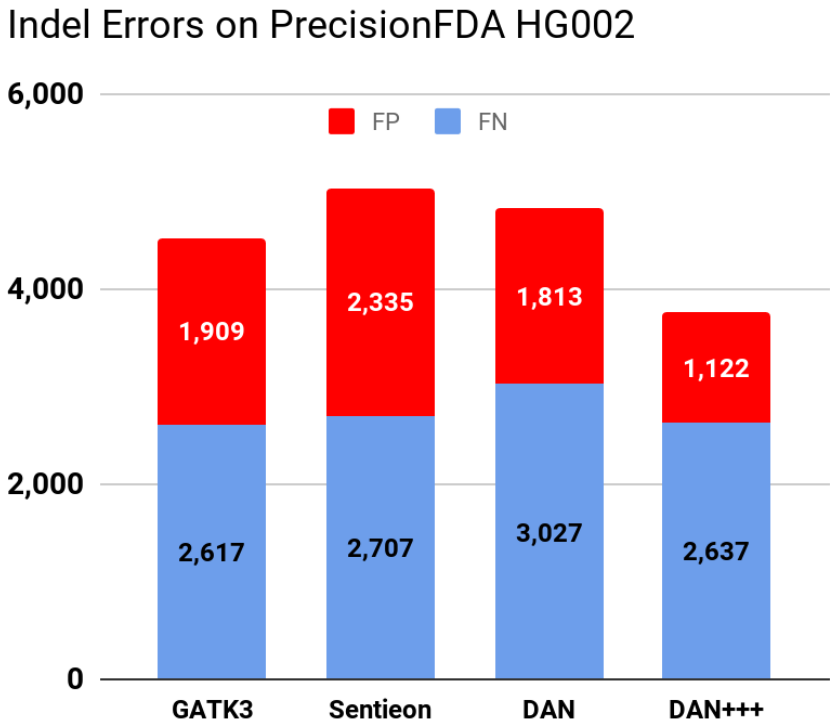


# Variant Calling Errors



# Variant Calling Error Breakdown



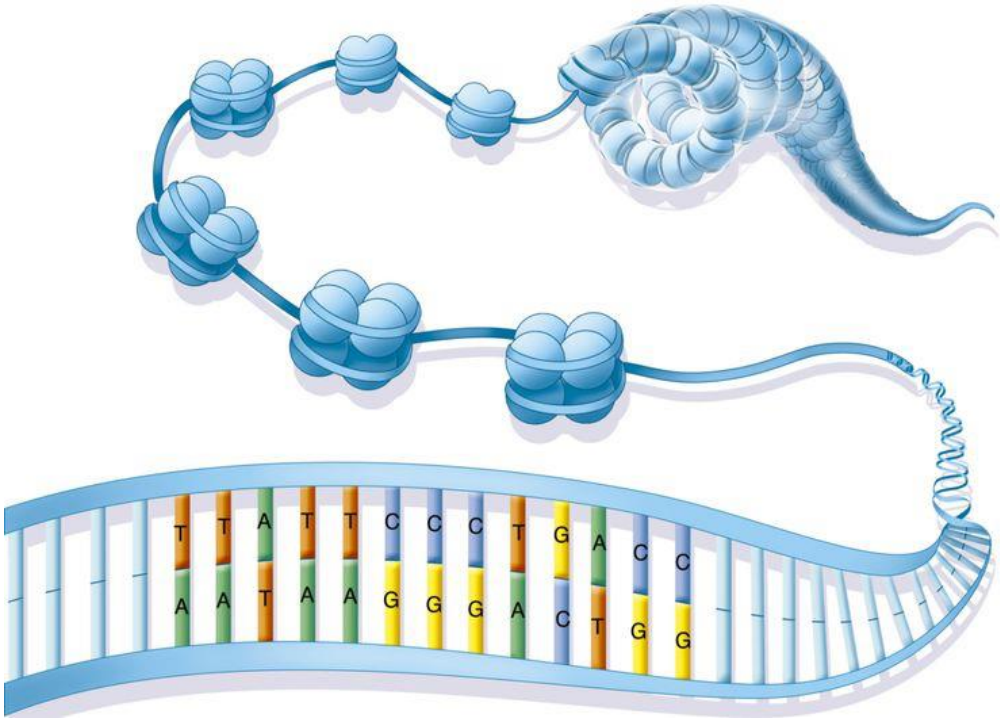

Atac Sequencing

# DNA: Open And Closed

Closed DNA inactive

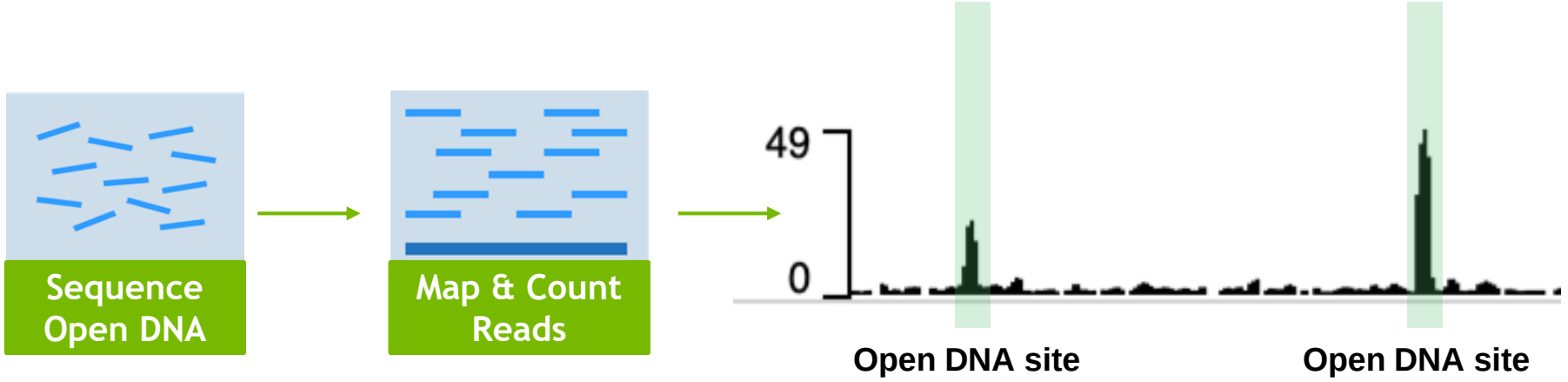
Open DNA active

Open DNA changes affect development & disease



# Atac Sequencing

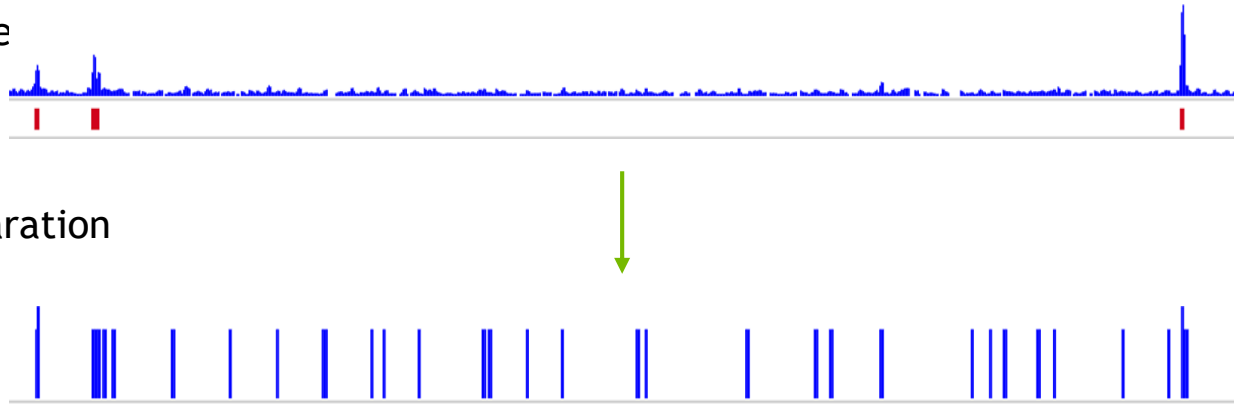
## Mapping Open DNA Sites



# Atac-seq Limits

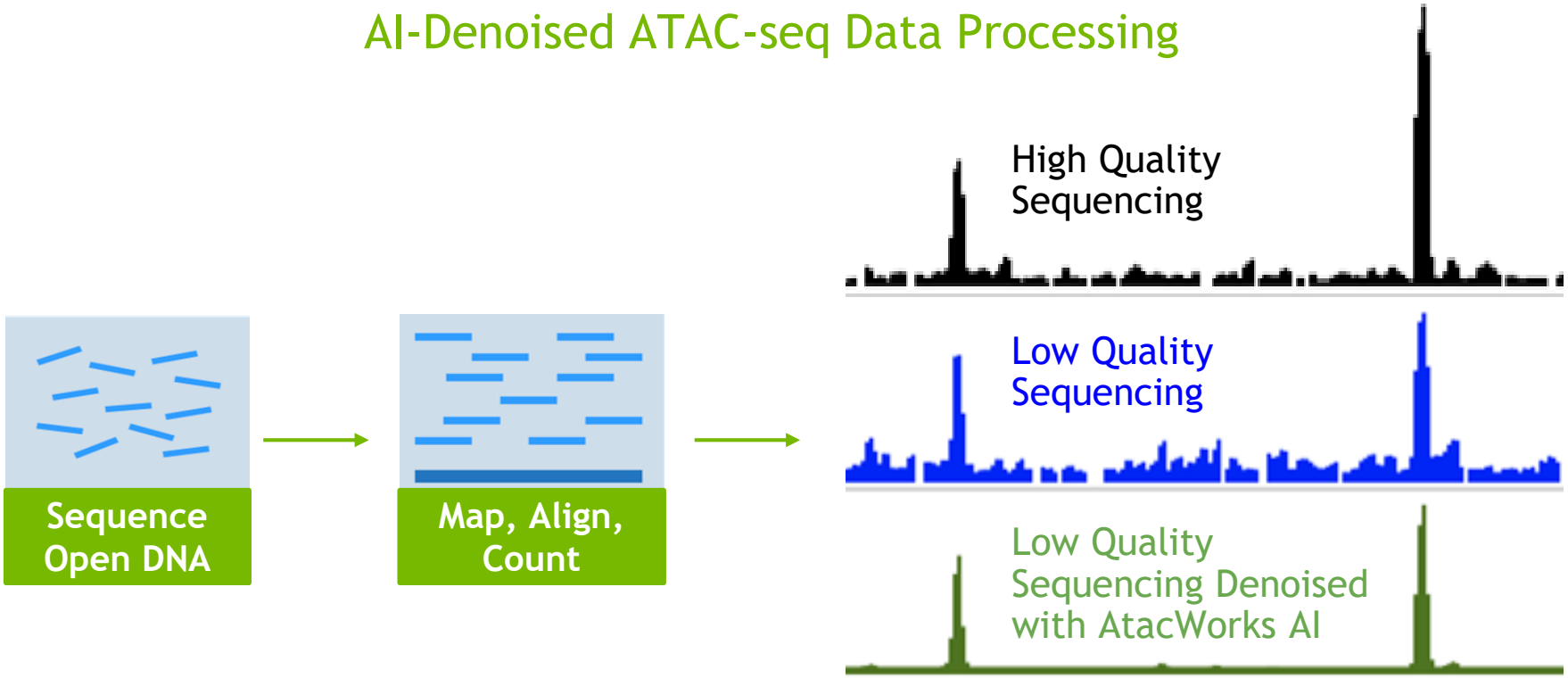
Atac-seq signal degrades in due

- Less sequencing
- Low quality sample preparation
- Small cell populations



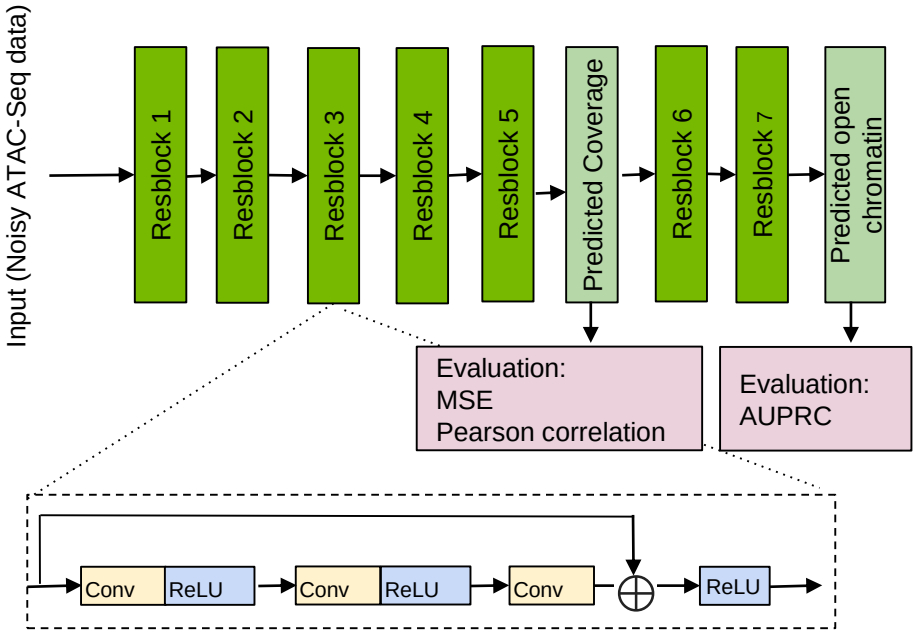
# AtacWorks SDK

## AI-Denoised ATAC-seq Data Processing



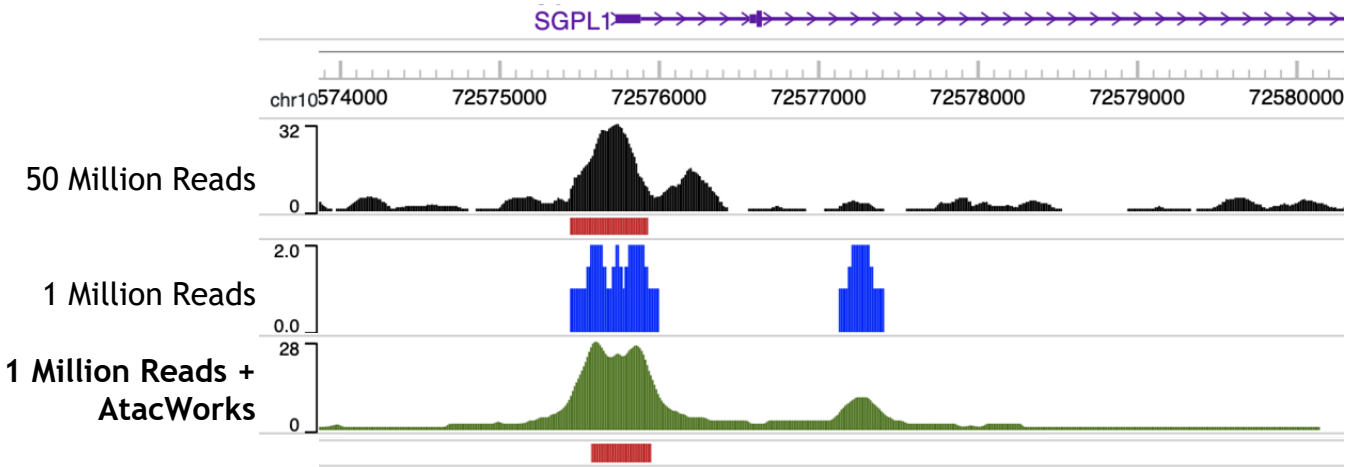
# AtacWorks Model

## Denoising + Open Chromatin Identification



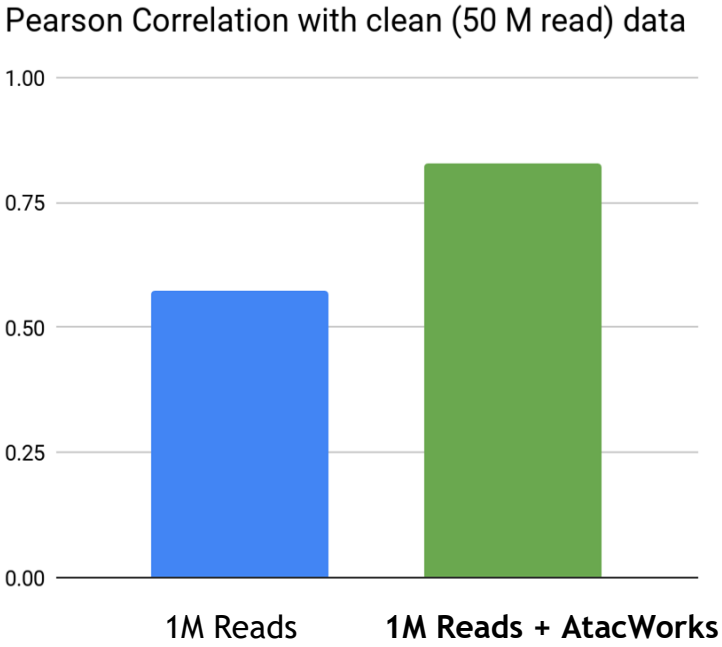
# Denoising Low Sequencing Data

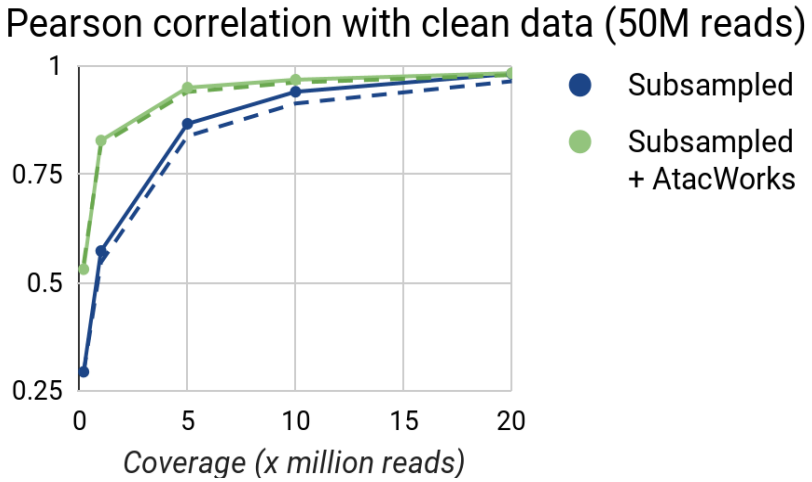
## AtacWorks identifies open chromatin from low-coverage data



# Genome-wide Sequencing Reduction

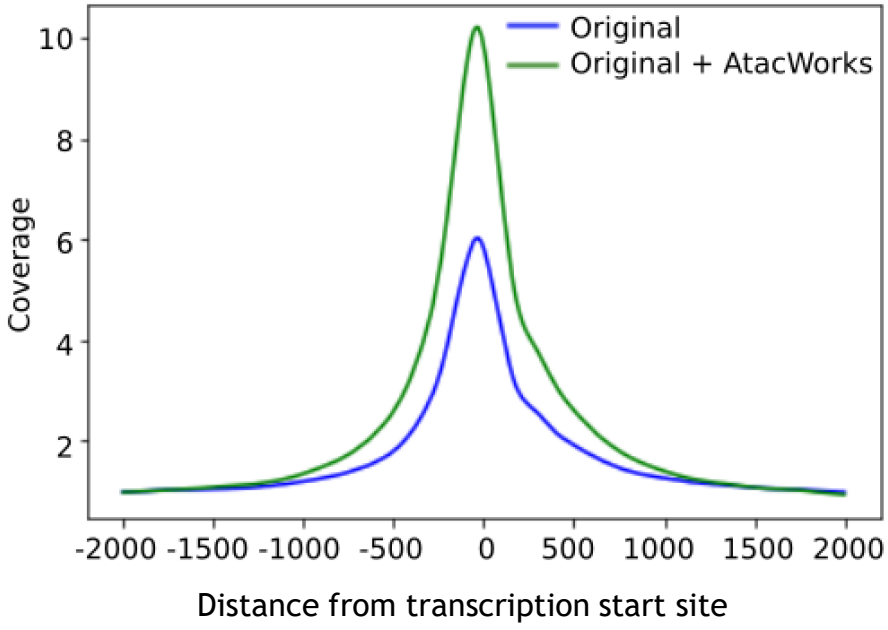
## AtacWorks Reduces Sequencing Requirements 3x

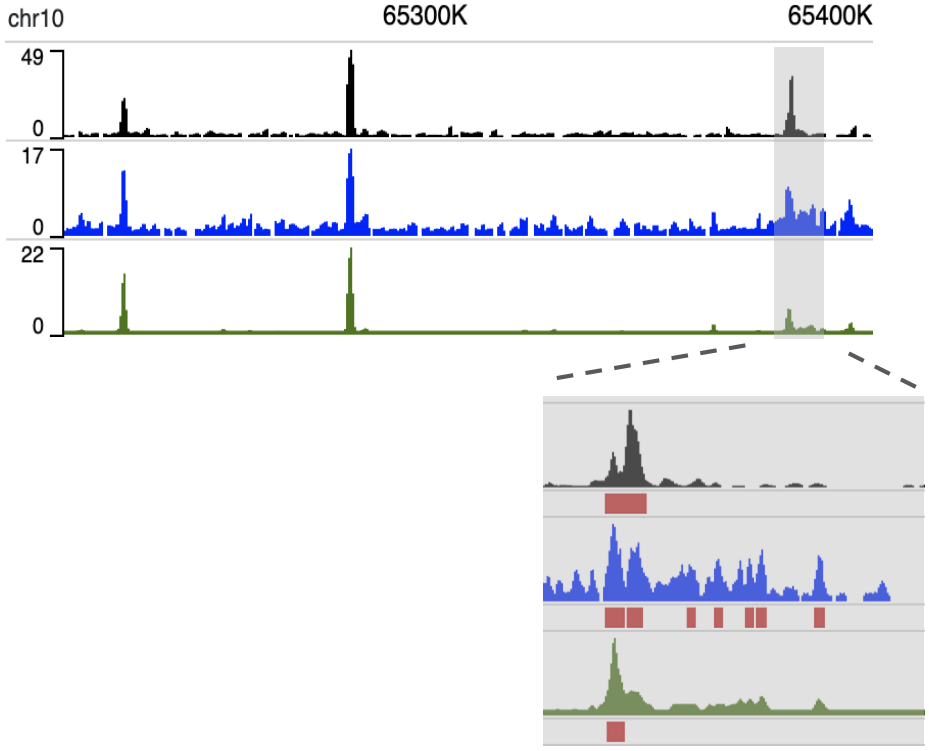


# Denoising Low Quality Sample

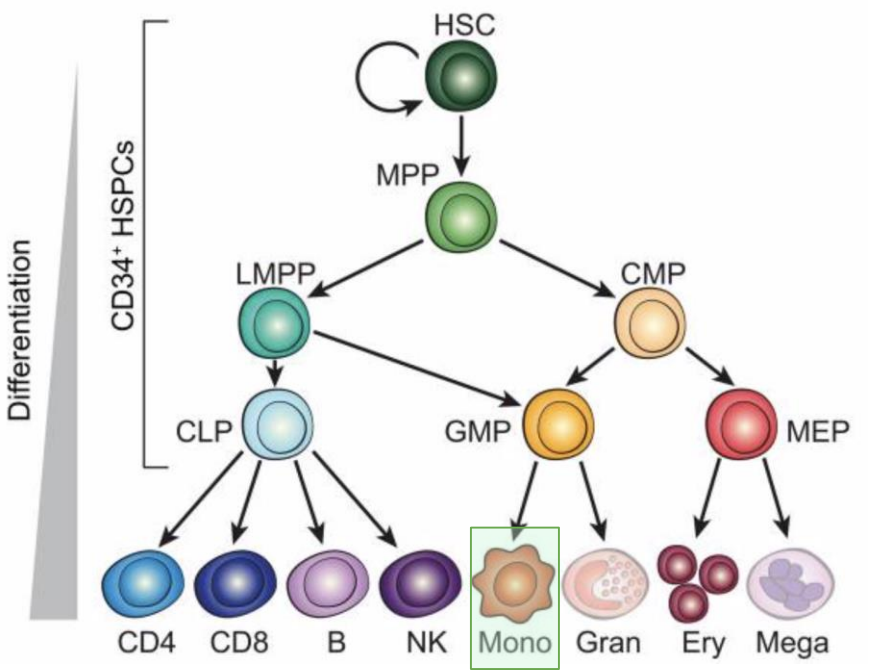
## AtacWorks improves signal-to-noise ratio in low quality samples

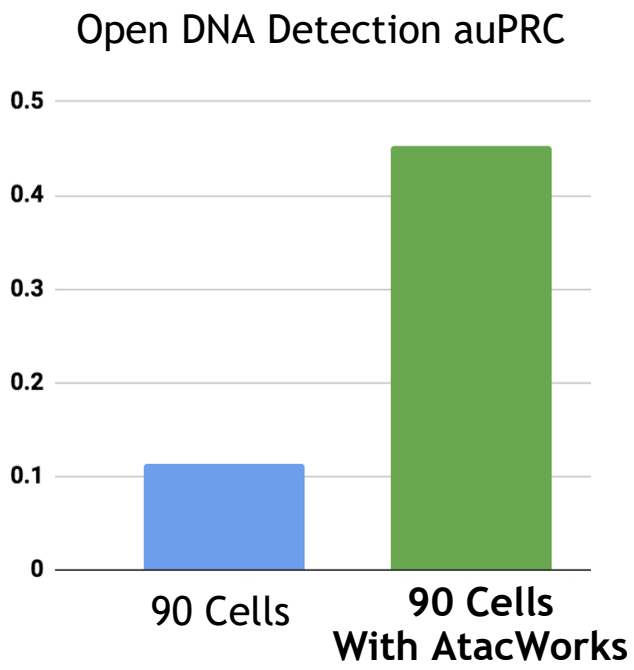


# Denoising Single Cell Atac-seq Data

## AtacWorks Improves Open DNA Detection From Few Cells

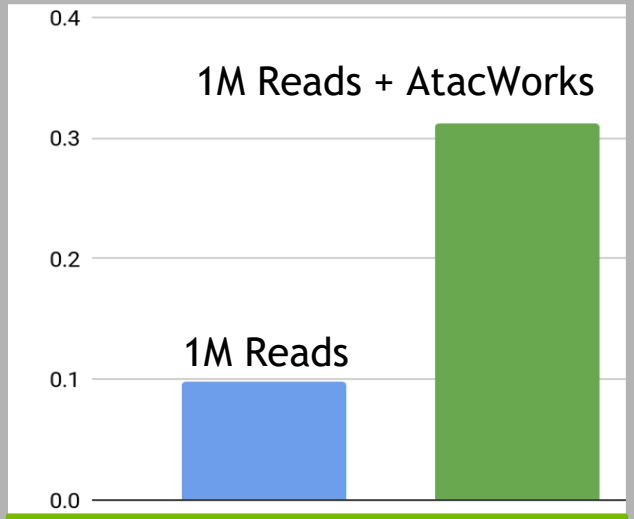


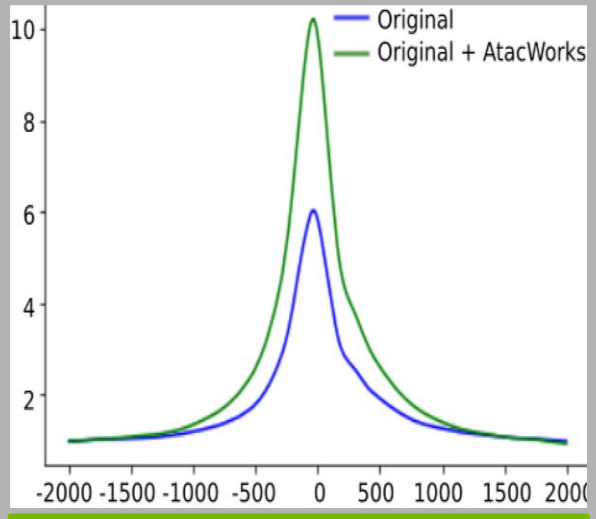
# AtacWorks SDK

SDK on Clara Genomics: https://github.com/clara-genomics/AtacWorks

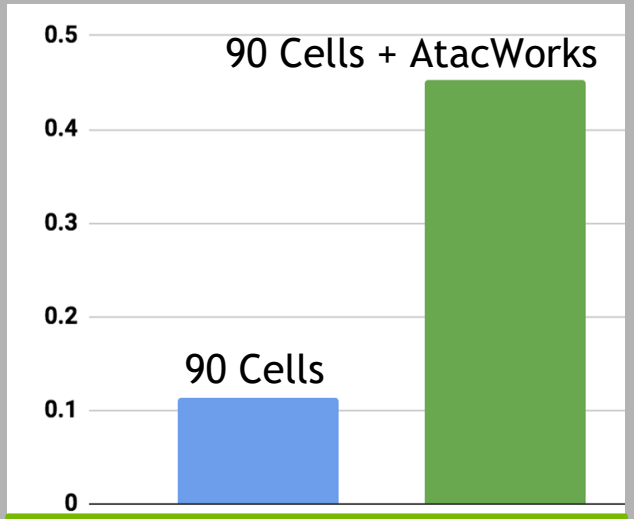
AtacWorks Preprint: https://www.biorxiv.org/content/10.1101/829481v1



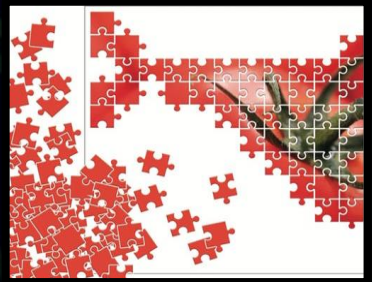

Reduce Sequencing Cost

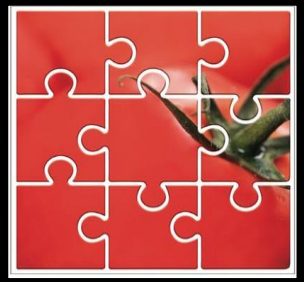



Improve Sample Quality




Increase Single Cell Resolution

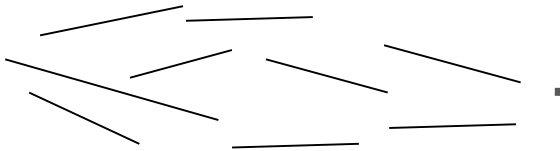




# Genome Assembly

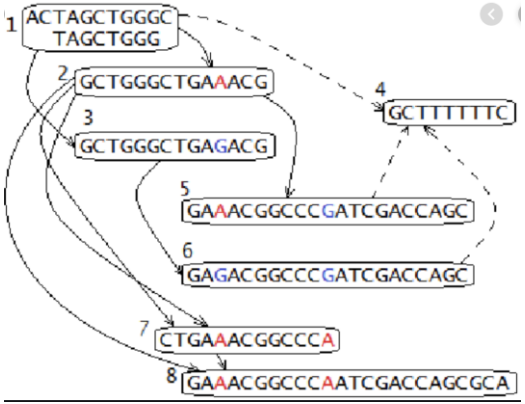
# Long Read De Novo Assembly

Step 1: Mapping to detect overlaps between reads

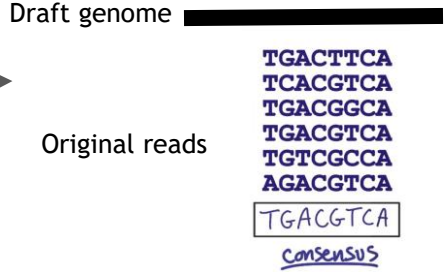


ACTCGGTCATTCGTGCTTTATC
GCGTTATCGTCTACTTCGT

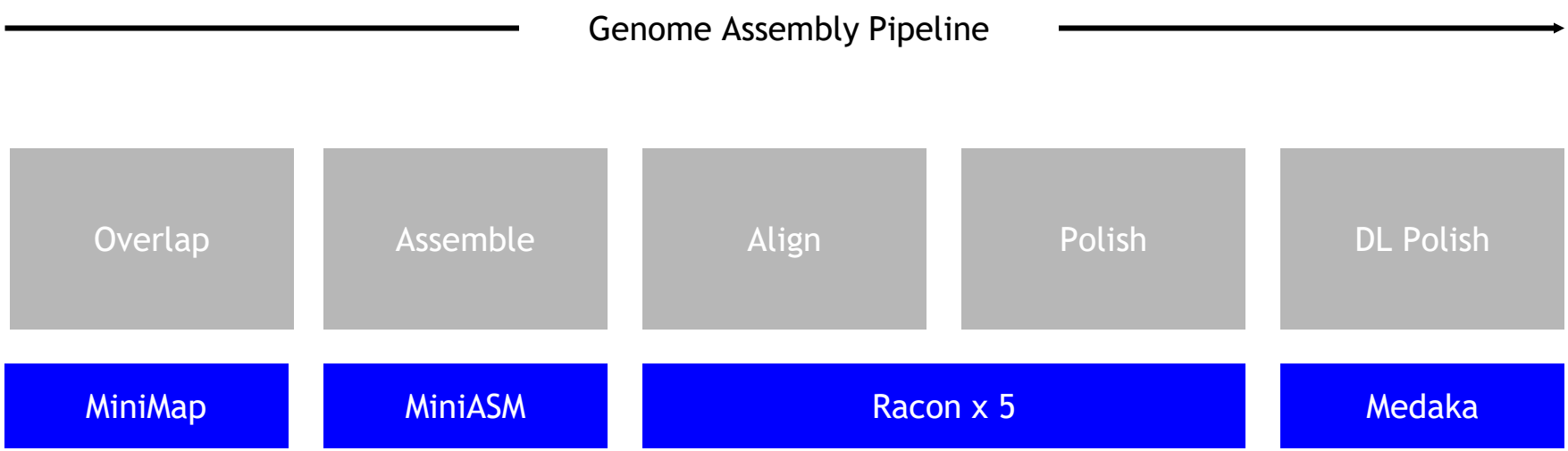
Step 2: Overlap graph traversal to generate draft genomes




Step 3: Error correction to polish genomes

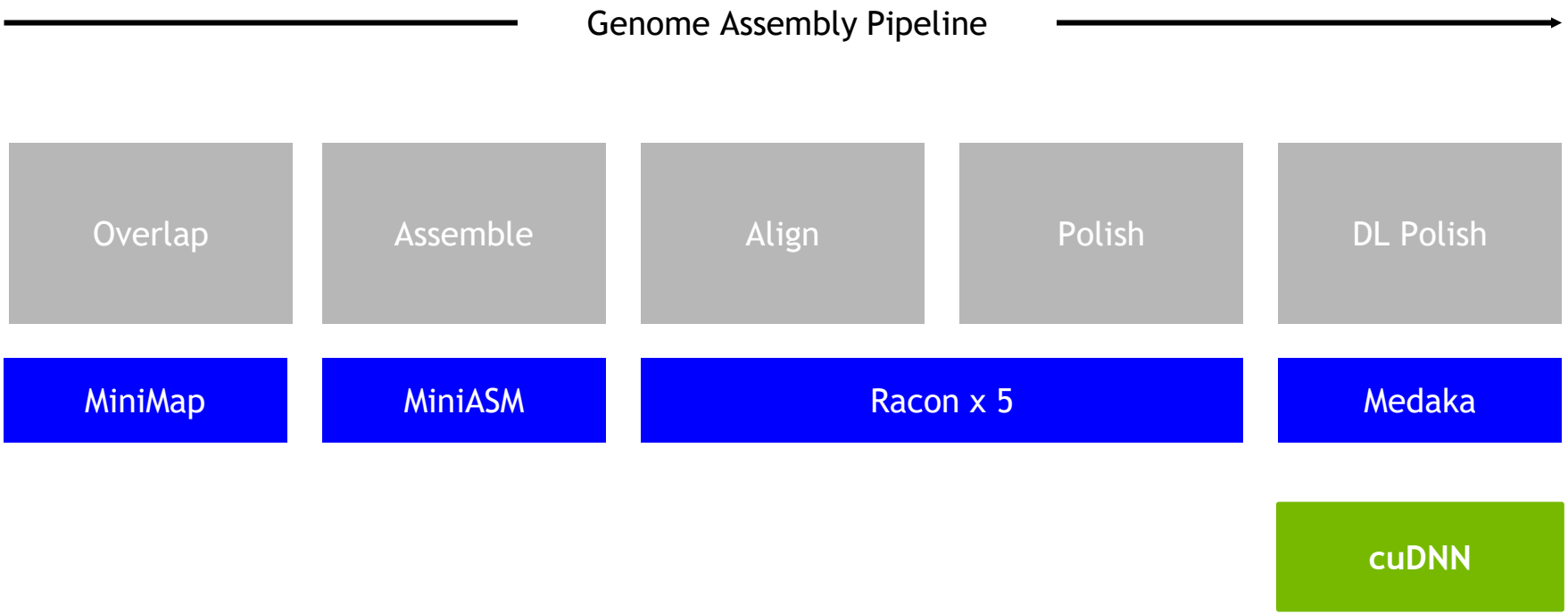


# Genome Assembly Workflow



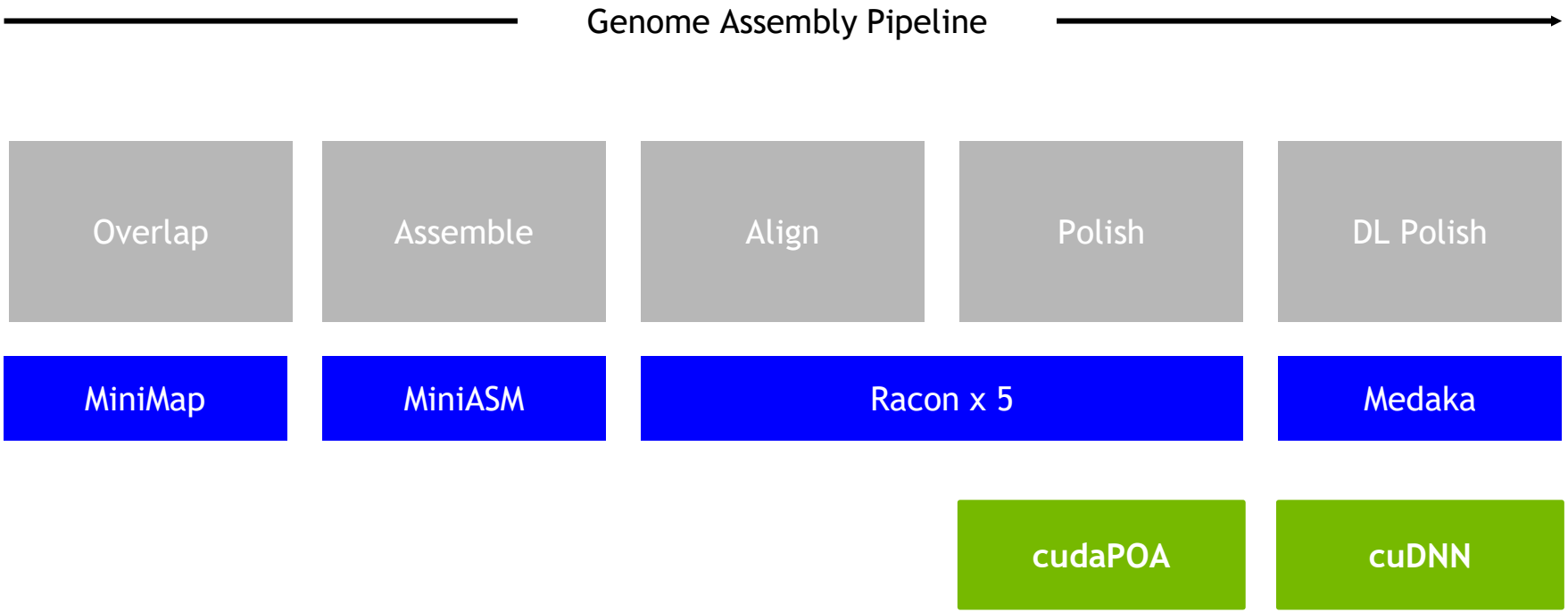
# Accelerated Genome Assembly Workflow

## Before ClaraGenomicsAnalysis



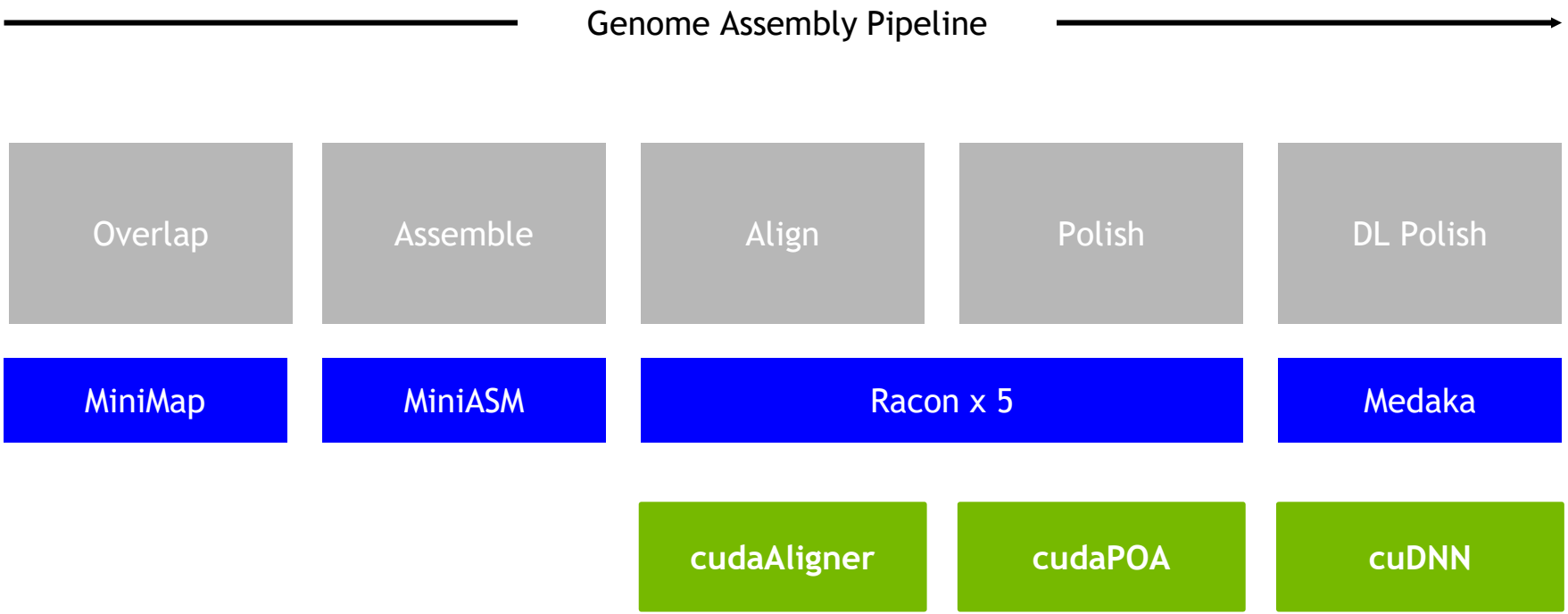
# Accelerated Genome Assembly Workflow

## ClaraGenomicsAnalysis 0.1



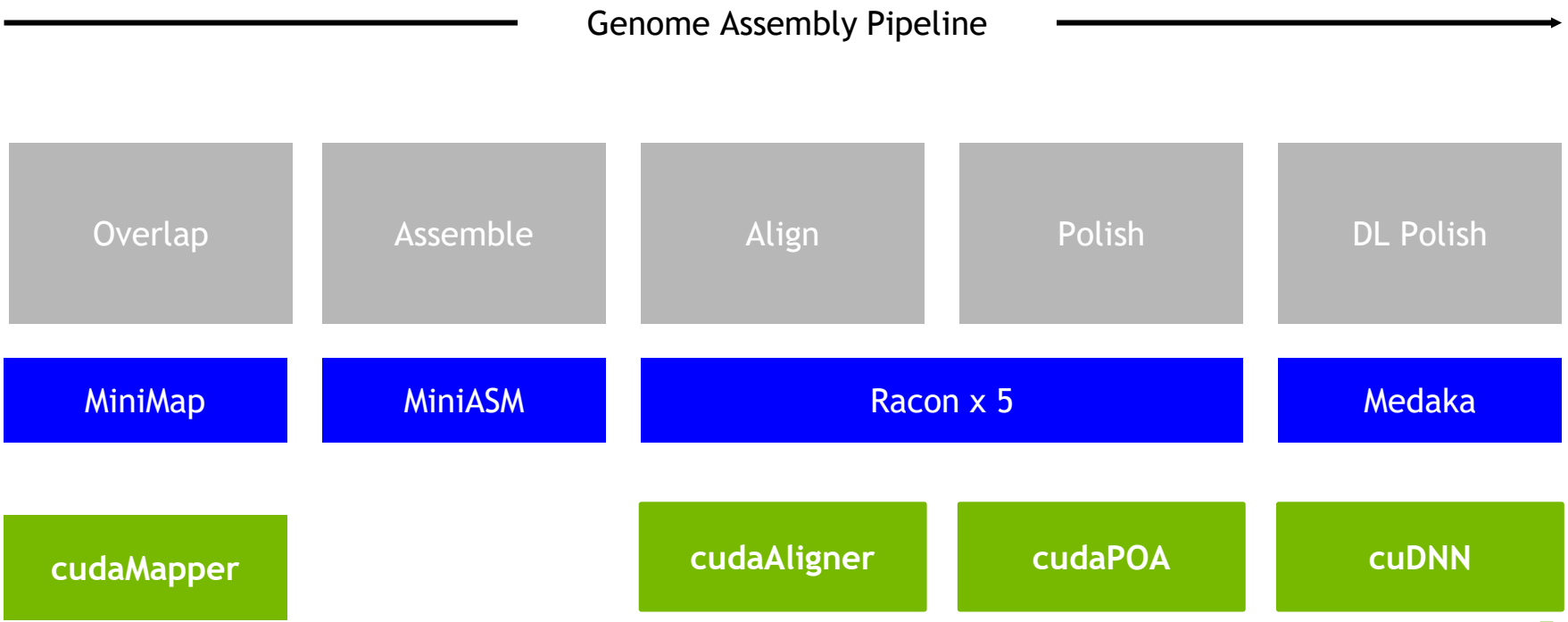
# Accelerated Genome Assembly Workflow

## ClaraGenomicsAnalysis 0.2



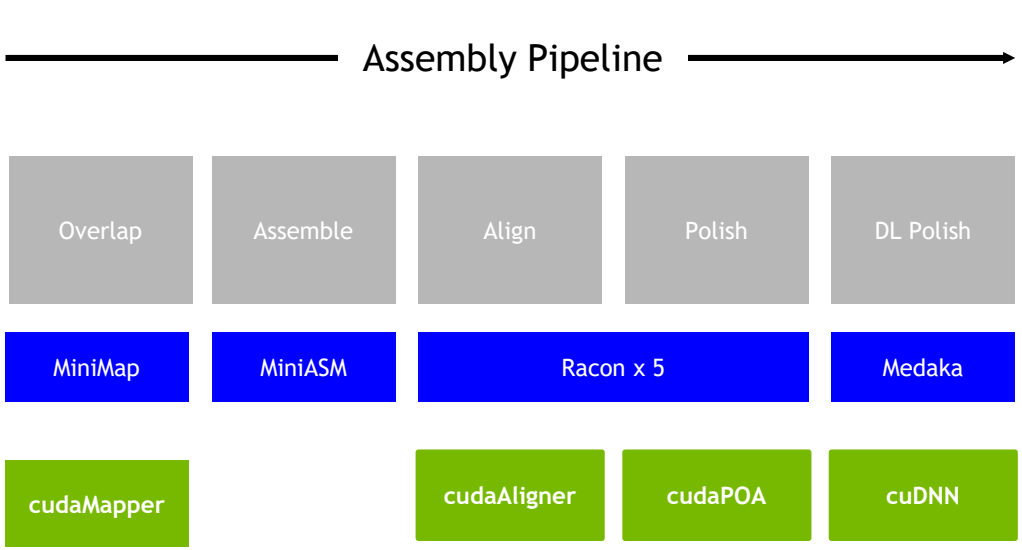
# Accelerated Genome Assembly Workflow

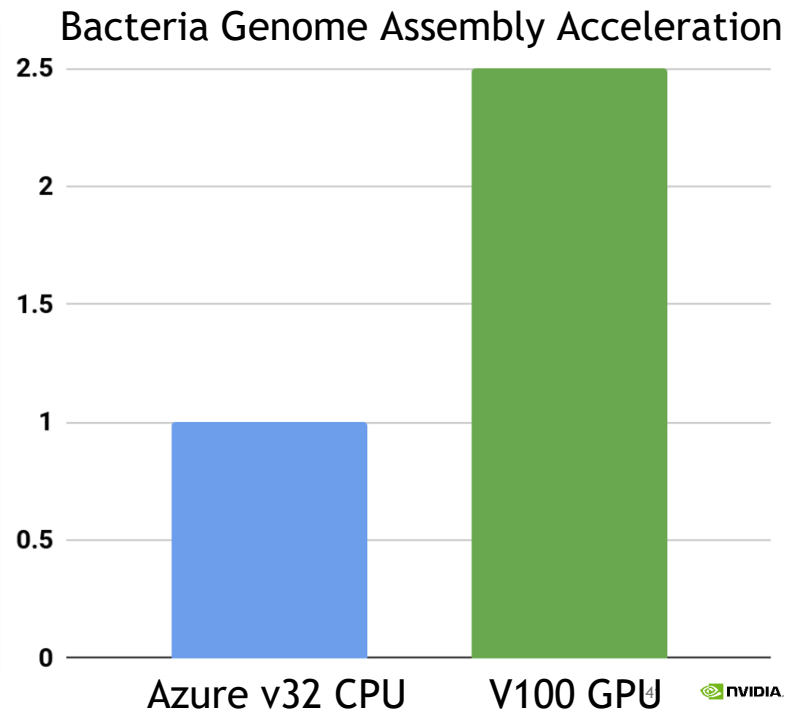
## ClaraGenomicsAnalysis 0.3



# ClaraGenomicsAnalysis SDK

## Enabling Accelerated Genome Assembly



# CLARA GENOMICS SW

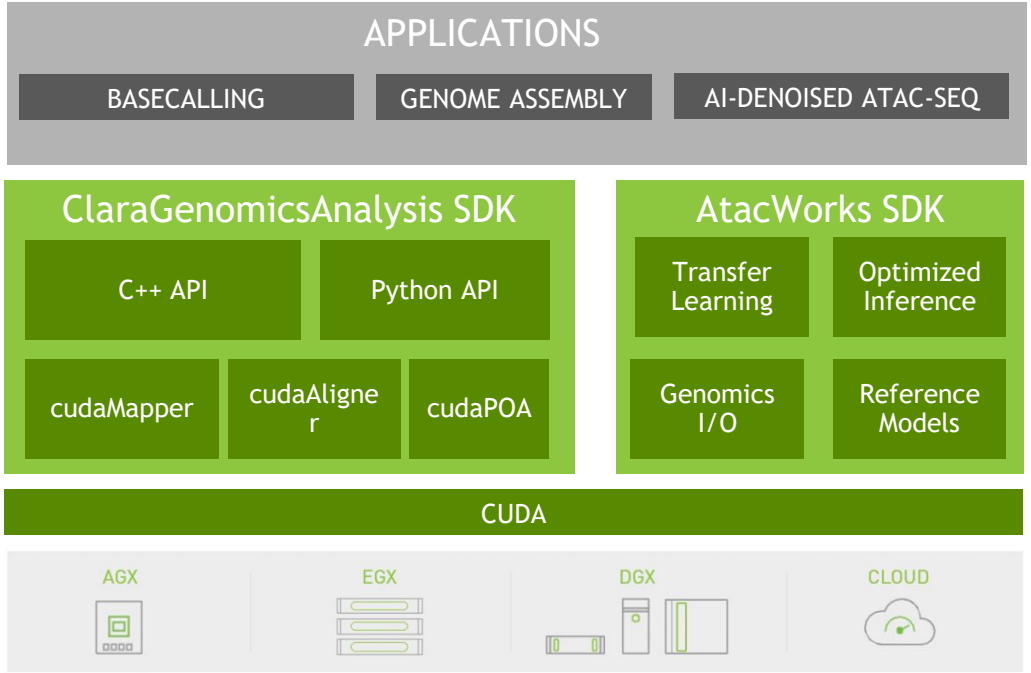
## Open Source CUDA-Accelerated Sequencing Analysis Tools

Reference Applications

Integration with 3rd Party Applications and Workflows

C++ and Python APIs

CUDA Accelerated HPC and Deep Learning Modules



# Useful Links

- Parabricks: https://www.parabricks.com
- ClaraGenomicsAnalysis
  - SDK on GitHub: https://github.com/clara-genomics/ClaraGenomicsAnalysis
  - C++ API Examples: cudapoa, cudaaligner
  - Python API Examples: cudapoa, cudaaligner
- AtacWorks
  - SDK on GitHub: https://github.com/clara-genomics/AtacWorks
  - AtacWorks Preprint: https://www.biorxiv.org/content/10.1101/829481v1
- **3rd party integrations:**
  - Racon: https://github.com/lbcb-sci/racon
  - Raven: https://github.com/lbcb-sci/raven
  - Bonito: https://github.com/nanoporetech/bonito
- Additional GPU Accelerated Genomics Applications:
  - Kipoi Model Zoo: https://ngc.nvidia.com/catalog/containers/hpc:kipoi
  - SigProfiler: https://github.com/AlexandrovLab/SigProfilerExtractor

# Accelerating Sequencing with GPU Computing and Deep Learning

Johnny Israeli