**SynthLongRead Framework**

**Applications**

SynthLongRead addresses a critical need in the long-read single-cell RNA sequencing field by providing synthetic data with known ground truth. Its applications include:

* **Benchmarking computational tools**: Objectively compare the performance of tools like FLAMES, Bambu, FLAIR, and others for tasks including isoform detection, quantification, and differential expression analysis.
* **Method development**: Support the development of new algorithms by providing datasets with known characteristics to validate against.
* **Protocol optimization**: Assess how factors like sequencing depth, read length, or error rates affect analytical performance, helping researchers design more effective experimental protocols.
* **Rare isoform detection**: Evaluate how well tools can identify low-abundance splice variants or isoforms by deliberately including them in the synthetic data.
* **Training machine learning models**: Generate large-scale training data for deep learning approaches to isoform detection and quantification.
* **Cross-platform comparison**: Create equivalent datasets that mimic different technologies (ONT vs PacBio) to assess platform-specific strengths and limitations.
* **Educational purposes**: Provide students and trainees with realistic datasets that have known answers, facilitating learning about long-read scRNA-seq analysis.

The key advantage of SynthLongRead is that it generates data where we know exactly what transcripts exist in each cell, allowing for precise evaluation of analytical performance without the uncertainty that exists when using real data as a benchmark.

**Simplified Explanation**

SynthLongRead is a tool that creates artificial (synthetic) long-read single-cell RNA sequencing data. This synthetic data mimics the characteristics of real sequencing data, including its errors and biases, but with the advantage that we know exactly what genes and transcript variants (isoforms) are present in each cell.

**How It Works**

The framework has four main steps:

1. **Learning from real data**: We analyse real sequencing data to understand what types of errors occur
2. **Modelling these errors**: We train machine learning models to reproduce these error patterns
3. **Creating cell expression patterns**: We generate realistic patterns of which genes and isoforms are expressed in each cell
4. **Generating synthetic reads**: We produce artificial sequencing reads with the learned error patterns

**Step 1: Learning from Real Data**

First, we align real sequencing reads to a reference sequence using minimap2. By comparing the reads to the reference, we identify where errors occur:

* **Substitution errors**: When one nucleotide (A, C, G, or T) is incorrectly read as another
* **Insertion errors**: When extra nucleotides appear in the read
* **Deletion errors**: When nucleotides are missing from the read

We also measure:

* How error rates change across the length of the read
* How often errors occur in repeated nucleotides (homopolymers, like AAAA)
* The quality scores associated with different types of errors
* The typical lengths of reads

**Step 2: Modelling Errors**

We use two neural networks to model how errors depend on the surrounding sequence:

1. **Error Type Predictor**: This network looks at a small window of sequence (5 nucleotides) and predicts whether the centre base will be read correctly, substituted, or deleted.
2. **Quality Score Predictor**: This network predicts the quality score (a measure of confidence) that would be assigned to each base, based on the sequence context and error type.

These models help us reproduce not just random errors, but errors that occur in specific sequence contexts - an important feature of real sequencing data.

**Step 3: Creating Expression Patterns**

Next, we generate which genes and isoforms appear in each cell:

1. For each cell, we randomly select which genes are "on" (expressed)
2. For expressed genes, we assign relative abundances to each isoform
3. We convert these relative abundances into counts using statistical distributions that mimic biological variability

This creates a "ground truth" matrix of transcript counts for each cell, which we can later compare with analysis results.

**Step 4: Generating Synthetic Reads**

Finally, we generate the actual sequencing reads:

1. For each transcript in each cell:
   * We determine how many reads to generate based on its expression level
   * We choose a read length from our learned distribution
   * We add cell barcodes and protocol-specific adapter sequences
   * We introduce errors according to our learned models
2. For each position in the read:
   * We look at the surrounding sequence context
   * We predict whether an error will occur based on our models
   * If an error occurs, we introduce the appropriate substitution, insertion, or deletion
   * We assign a quality score based on our quality model

The result is a FASTQ file that looks like real sequencing data but with a known ground truth.

**Benchmarking and Evaluation**

To evaluate how well our synthetic data mimics real data, we compare:

* Read length distributions
* Base composition
* Error rates
* Quality score distributions

To test analysis tools, we:

1. Run the tool (e.g., FLAMES) on our synthetic data
2. Compare the results to our ground truth
3. Calculate accuracy metrics like:
   * How well the tool detects which transcripts are present (F1-score)
   * How accurately it quantifies transcript abundance (correlation)
   * How well it captures the relative proportions of different isoforms from the same gene

This allows researchers to objectively compare different analysis methods and understand their strengths and limitations.

**Data-Driven Approach**

The SynthLongRead pipeline is largely data-driven and agnostic in its approach. It adapts to whatever reference dataset you provide rather than relying on hardcoded error profiles or distributions. Here's how it works:

**Data-Driven Components**

1. **Error profiles**: The pipeline learns all error patterns (substitutions, insertions, deletions) directly from your provided real long-read dataset
2. **Quality scores**: Quality score distributions are extracted from your specific dataset, not from generalized assumptions
3. **Read length distributions**: The length profiles are modelled based on your input data's read length distribution
4. **Sequence context effects**: The relationship between surrounding sequence contexts and error probabilities is learned from your data
5. **Expression patterns**: While statistical models are used to generate expression matrices, the overall characteristics will reflect your reference GTF annotation

**Minor Platform-Specific Elements**

There are a few areas where the pipeline has default settings that can be customized:

1. **Platform presets**: Different alignment parameters for ONT vs PacBio data (using minimap2's platform-specific presets)
2. **Protocol elements**: Default adapter sequences and barcode structures are included but can be customized for your specific protocol
3. **Cell barcode handling**: Default settings for barcode length and error rates that should be adjusted for your protocol

**Flexibility**

The framework is designed to work with:

* Different organisms (human, mouse, or any other organism with a reference transcriptome)
* Different sequencing technologies (ONT or PacBio)
* Different single-cell protocols (full-length or fragmented)
* Different cell numbers and sequencing depths

You simply provide:

1. A reference transcriptome FASTA file
2. A reference GTF annotation
3. A real long-read scRNA-seq or snRNA-seq dataset
4. (Optional) A reference genome for benchmarking

The pipeline then extracts all the necessary characteristics from your data and builds synthetic reads that match those properties. This agnostic, data-driven approach means SynthLongRead can generate realistic synthetic data for diverse experimental conditions without needing to be specifically reconfigured for each scenario.