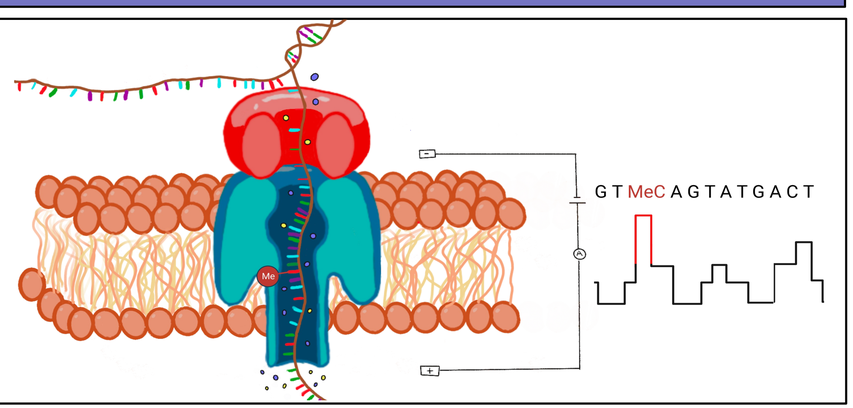
Squiggle Analysis for Metagenomic Viability Inference

Harika Urel, Amit Fenn, Tim Reska, Anoop Chandran, Michael Schloter, Stefan Kesselheim, Lara Urban

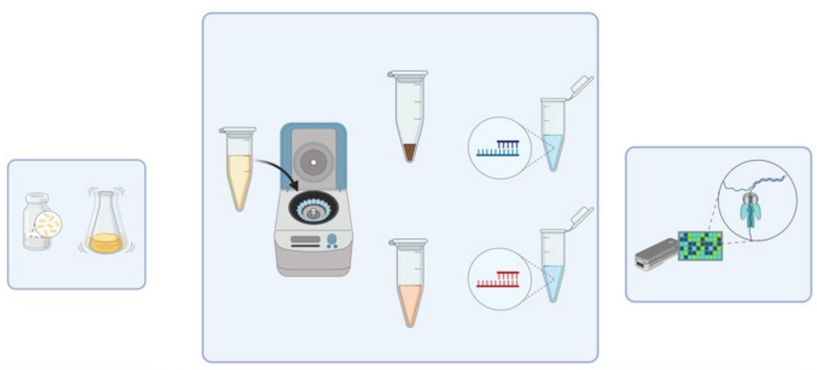
Motivation

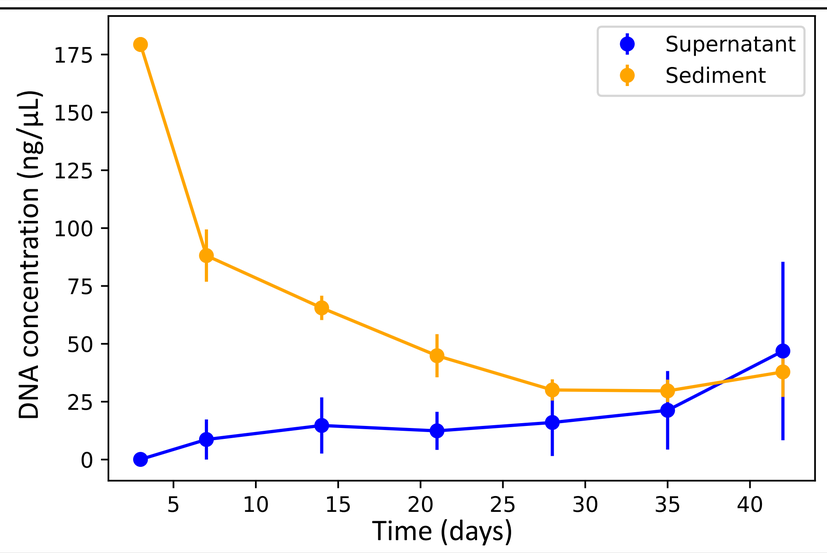
Metagenomic approaches enable unbiased whole microbial community characterizations but cannot differentiate between living and dead microbes, which is however crucial for virulent pathogen detection. Traditional methods for identifying living microbes are labor-intensive and time-consuming. **This project aims to develop a computer-based framework using nanopore sequencing to predict microorganism viability from raw metagenomic squiggle data.** Nanopore sequencing measures ionic current fluctuations in signal traces ("squiggle") in real-time as single-strand nucleotides pass through membrane-embedded nanopores. Squiggles can detect atomic changes that reveal functionally important genomic characteristics. We hypothesize that DNA from dead microorganisms gets exposed to environmental damage and lacks DNA repair mechanisms, thereby generating a squiggle signal that is distinct from DNA in living organisms. We extracted DNA from living and dead bacteria, obtained squiggle data via nanopore sequencing, and retrained an existing deep neural network, "SquiggleNet" (Bao et al., 2021), to explore differences in squiggle data from such AI predictions.



Data

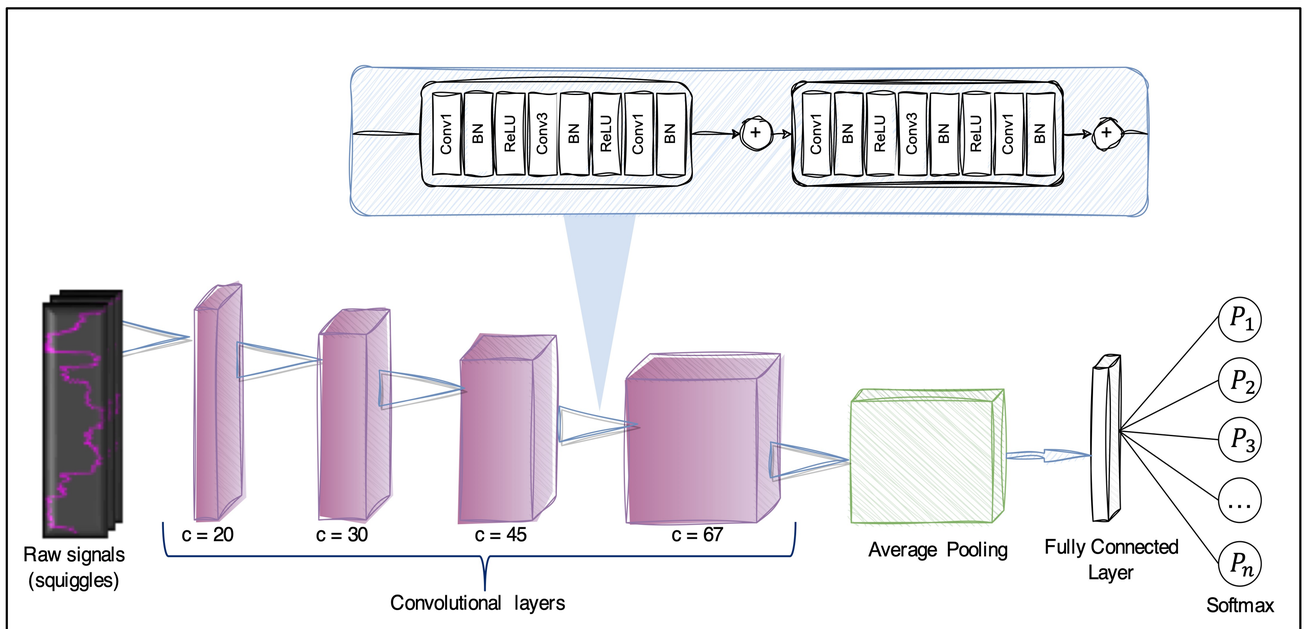
We cultivated *Escherichia coli* for 42 days, with nutritional deprivation from day 3 leading bacteria to die. We then extracted DNA and used nanopore rapid barcoding sequencing (RBK004-24) to obtain squiggle data from living and dead DNA. We used centrifugation to separate DNA from dead and living cells via **phase separation**, where intact cells with **"living DNA"** will be collected in the sediment while **"dead DNA**" will be present in the supernatant**. As a result, the DNA concentration in the supernatant increases over time as the DNA released through the disrupted membrane of dead cells accumulates.** For model training and evaluation, we used the squiggle data of last-day dead DNA and first-day alive DNA.





1D-CNN Squiggle Viability Model

Model Training

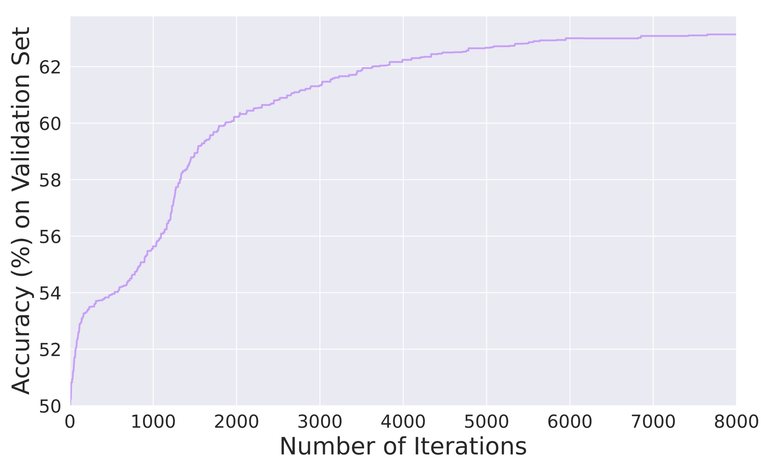


The viability model is a **1D-ResNet-based binary classifier** designed for **squiggles**.

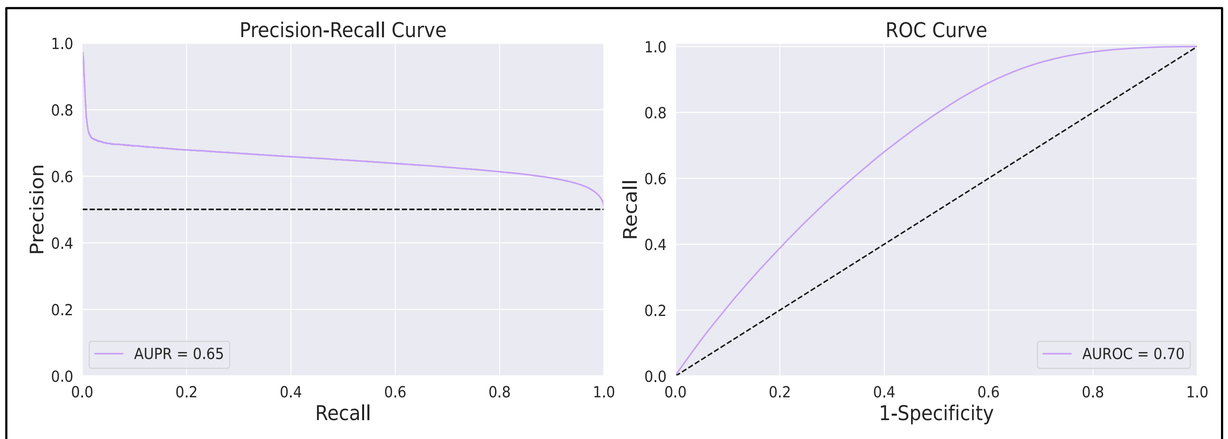
- The architecture comprises four 1D-ResNet layers, each featuring two BottleNeck blocks that include convolutional layers with kernel sizes of 1 and 3, ReLU activation functions, and batch normalization.
- The number of channels in each layer increases by a factor of 1.5.
- After the final convolutional layer, average pooling is applied, followed by a fully connected layer.
- We compute the predicted probabilities by applying the softmax activation function to the raw probabilities.

Data	Training Set (1:1)	Validation Set (1:1)	Test Set (1:1)
Dead (42th day) & Alive (1st day)	4,427,300	1,475,766	1,475,766

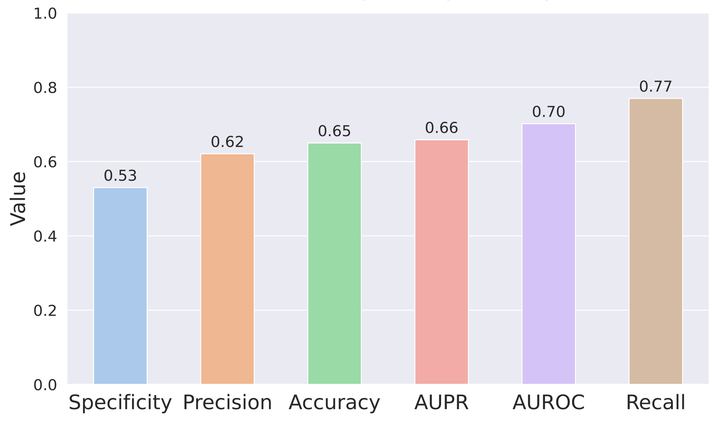
- We trained our model on multiple GPUs for over **50 epochs** using the Adam Optimizer on a viability dataset with the learning rate of **1e-3** and the batch size of **30,000**.
- During training, we evaluated the model's accuracy on a validation dataset, and **the highest accuracy achieved was 63%.**



Model Evaluation



We evaluated the model performance on the test set. An Area Under the Precision-Recall curve (AUPR) of 0.66 represents moderate efficacy in managing imbalanced datasets, and an Area Under the Receiving Operator Characteristic curve (AUROC) of 0.70 suggests that the model exhibits a reasonable balance between sensitivity and specificity.



A sensitivity of 0.77 indicates fairly good performance in identifying viable DNA (possible pathogens). These results indicate that there is room for improvement in the model's performance particularly in specificity (0.53), to achieve better overall performance.

Outlook

We propose the following steps for further improvement:

1. Increase channel sizes of the first layer (similar to Guppy's first layer) to capture more complex features.
2. Integrate dropout layers after existing layers for regularization to prevent overfitting and improve generalization.
3. Perform hyperparameter tuning to optimize the model's performance.
4. Investigate the use of explainable AI techniques to understand which features are being recognized by the model, potentially leading to further insights and refinements.
5. Test the model on data from different time points to see if the model picked up temporal information.

References

Bao, Y., Wadden, J., Erb-Downward, J.R. *et al.* SquiggleNet: real-time, direct classification of nanopore signals. *Genome Biol* **22**, 298 (2021). https://doi.org/10.1186/s13059-021-02511-y