RNA methylation calling with nanopore sequencing

Sebastian Krautwurst
RNA Bioinformatics
FSU Jena

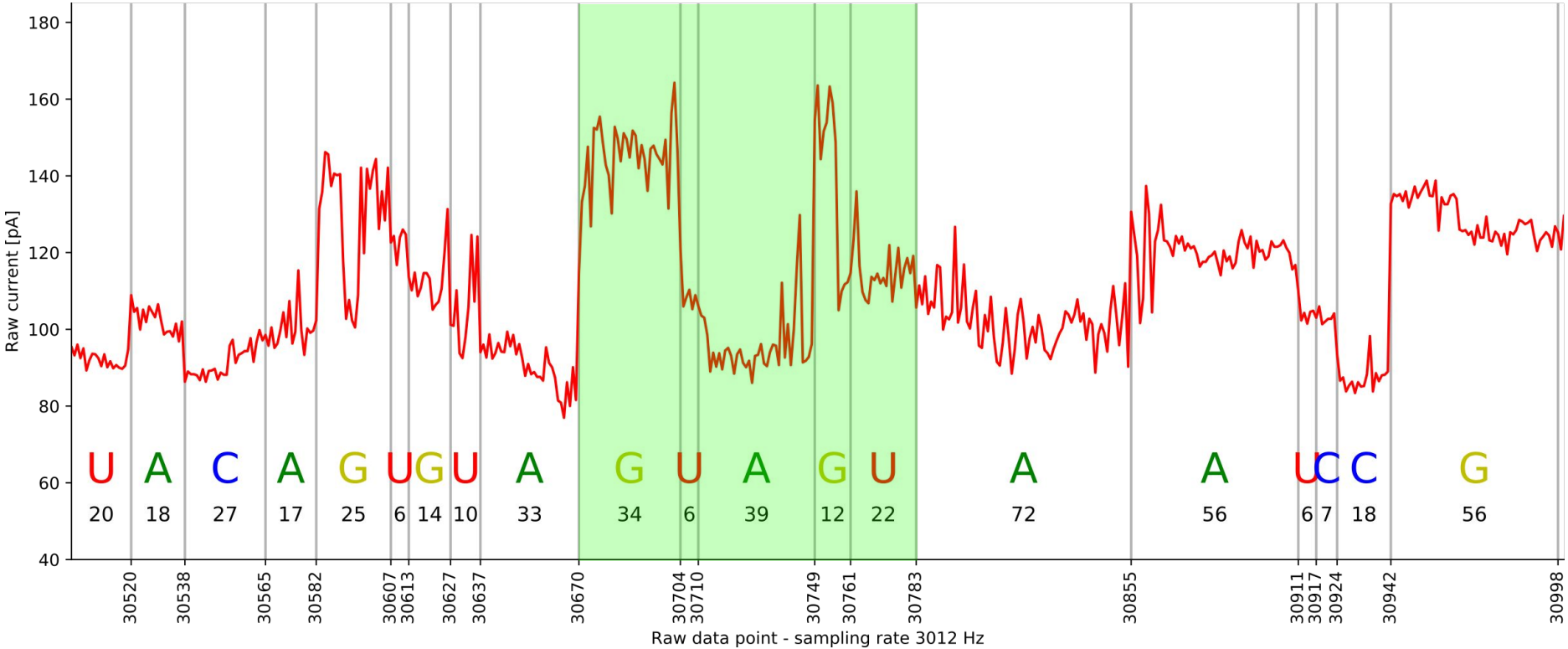
2020-02-11
35th TBI Winterseminar in Bled

Direct RNA sequencing with nanopores

- Protocol for MinION
- Directly sequences RNA strands

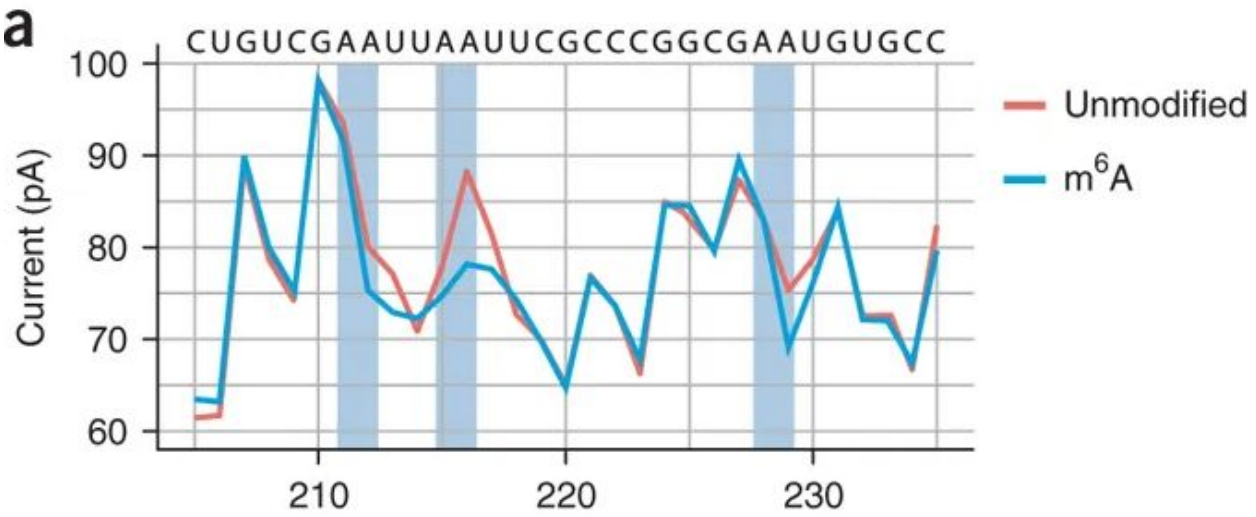
- No fragmentation
- No amplification
- Capture full transcripts
- Modifications are retained

Basecalling: squiggle to basecalls



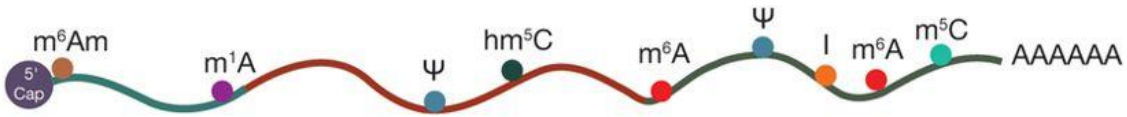
Modification calling

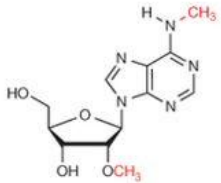
- Modifications change the signal
- DNA:



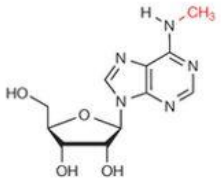
Garalde et al. *Nature Methods*. 2018

RNA modifications

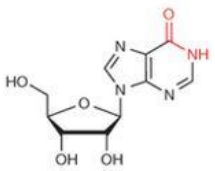




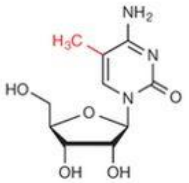
*N*⁶,2'-O-dimethyladenosine (m⁶Am)



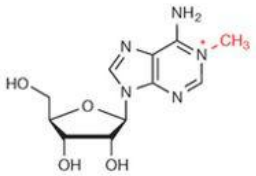
*N*⁶-methyladenosine (m⁶A)



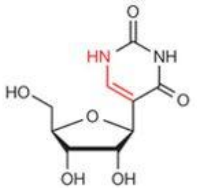
Inosine (I)



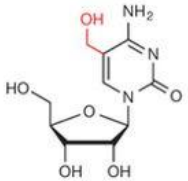
5-methylcytidine (m⁵C)



*N*¹-methyladenosine (m¹A)



Pseudouridine (Ψ)



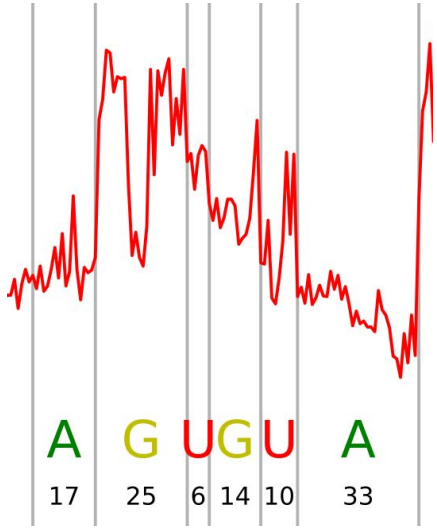
5-hydroxylmethylcytidine (hm⁵C)

Li et al. *Nature Methods*. 2017

Guppy basecaller + **Taiyaki** training tool

Guppy:

- For DNA: 5mC, m6A calling is available
- Only trained for certain contexts: CpG, RRACH motif
- Basecall output + Modification output



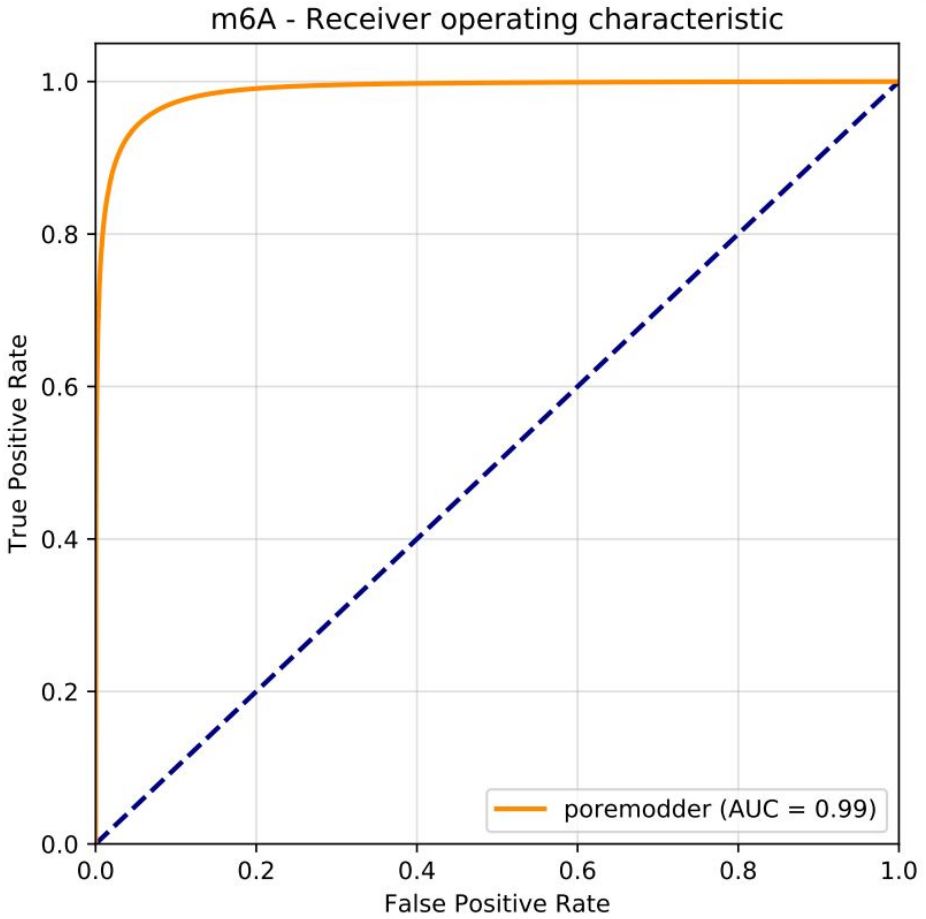
Taiyaki:

- Tool used for training Guppy
- Can train for modified bases
- Has “re-squiggle” functionality to prepare data for training

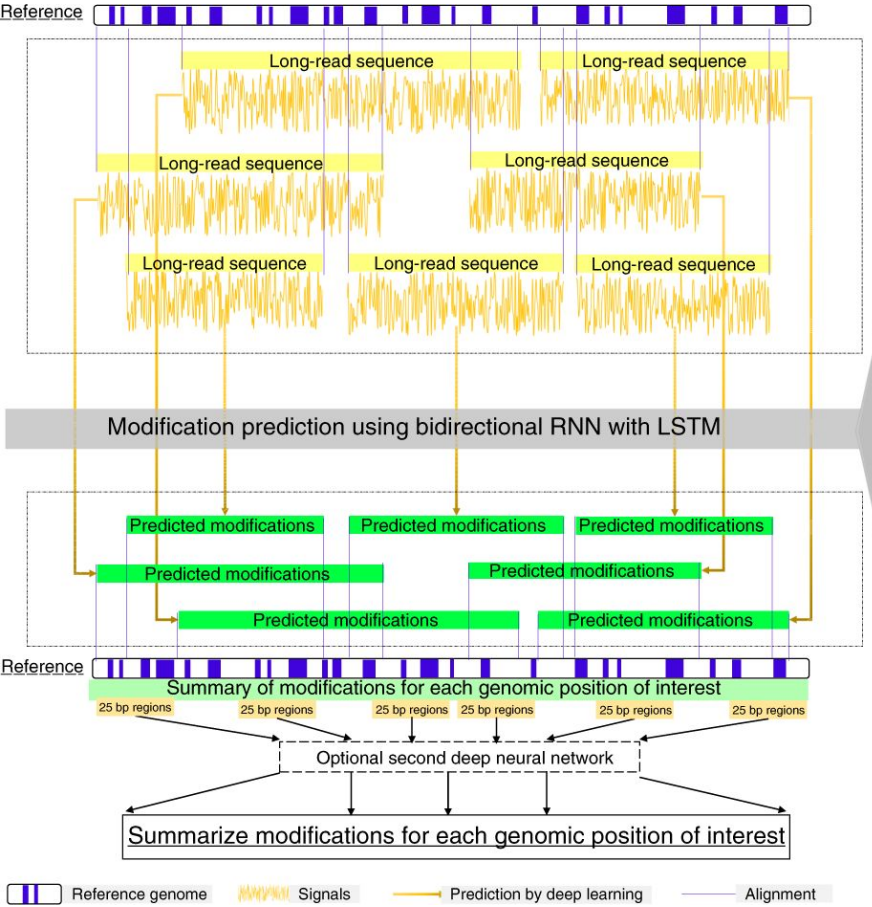
Training data for RNA?

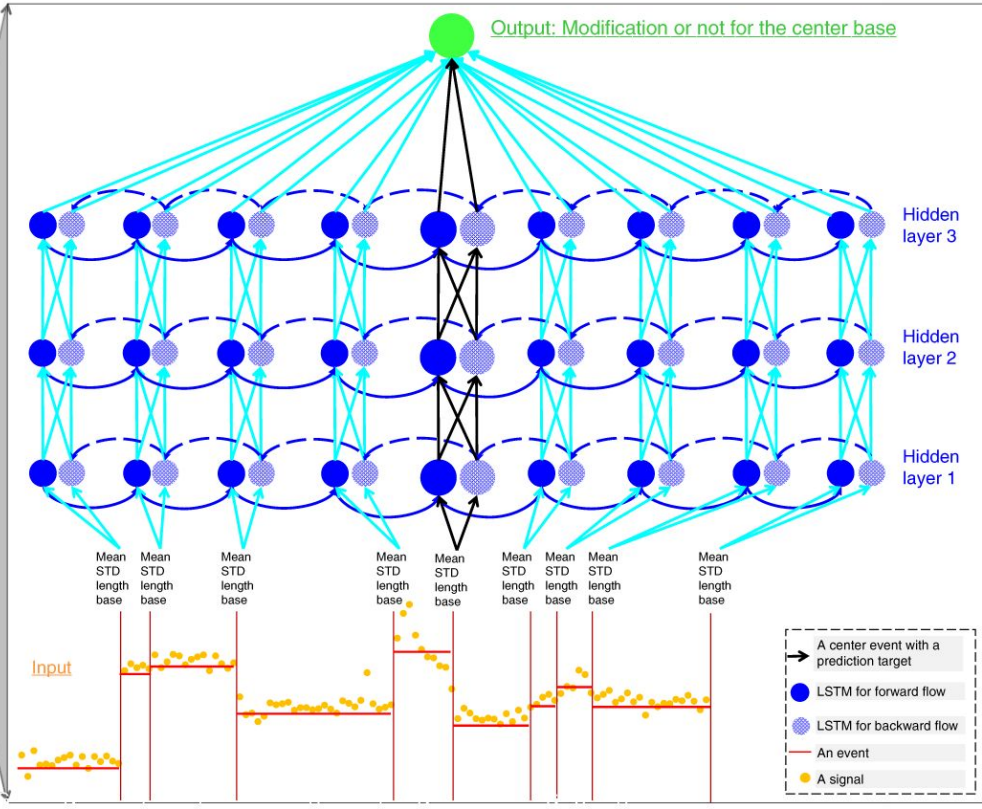
- No good ground truth datasets
- In vitro transcripts:
  - With m6A - fully methylated
  - With A - fully canonical
- Model trained with Taiyaki

- Bad overfitting on training data
- Cannot basecall anything else



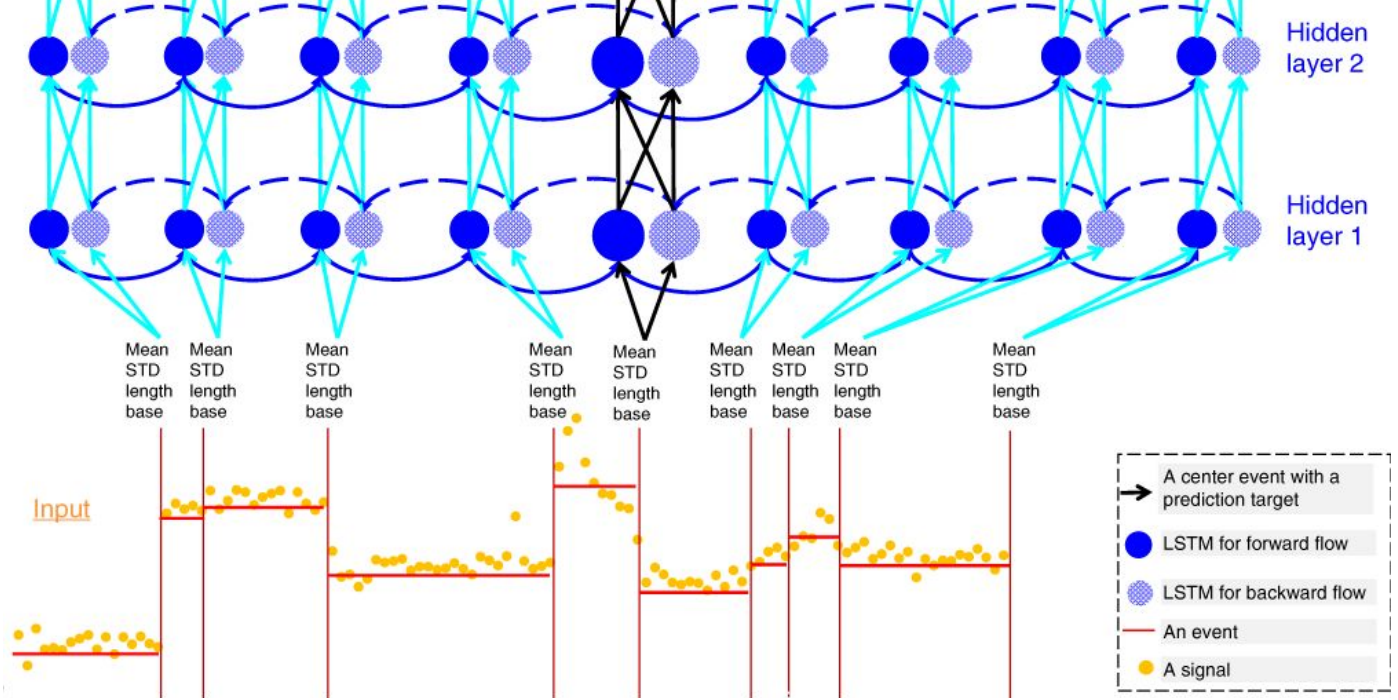
DeepMod - DNA only (Liu et al.)





Reference genome | Signals | Prediction by deep learning | Alignment | Long reads | Predicted modification for each base | Optional second neural network

Liu et al. *Nature Communications*. 2019



E. coli DNA data, per position:

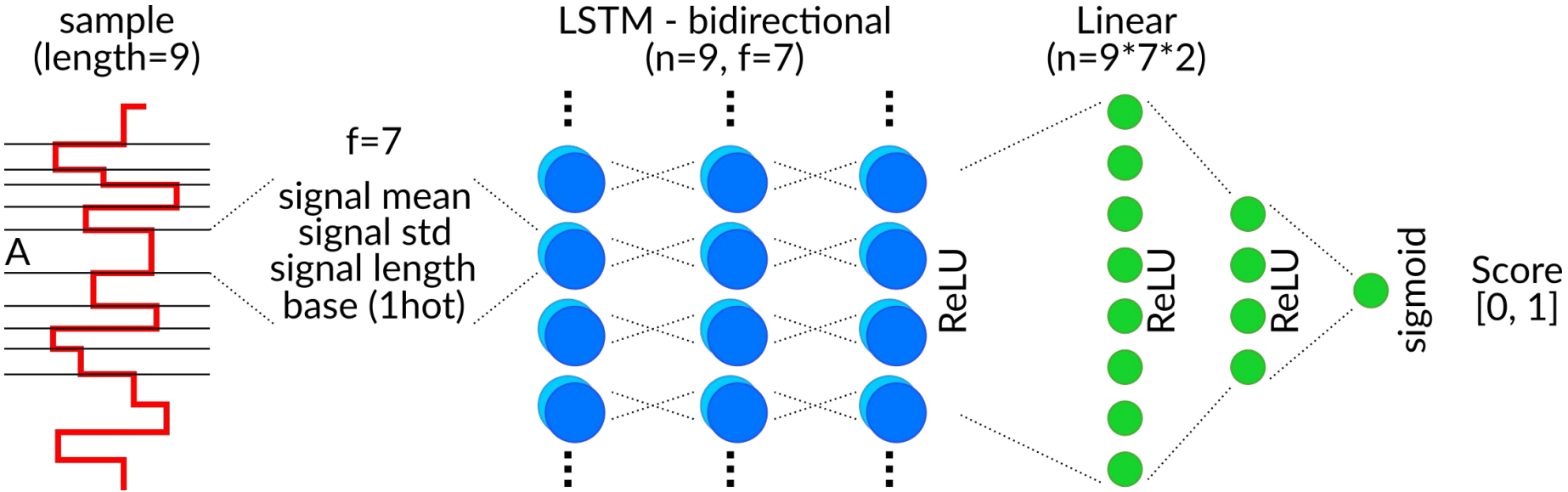
- **5mC** 0.99 average precision
- **m6A** 0.9 average precision

Liu et al. *Nature Communications*. 2019

Datasets - In vitro transcripts

m6A data	# reads canonical	# reads m6A	# samples (7mers)
Epinano replicate 1	46,628	8,739	15,990,176
Epinano replicate 2	603,275	94,046	TBD
Our own m6A data	in progress	26,621	TBD

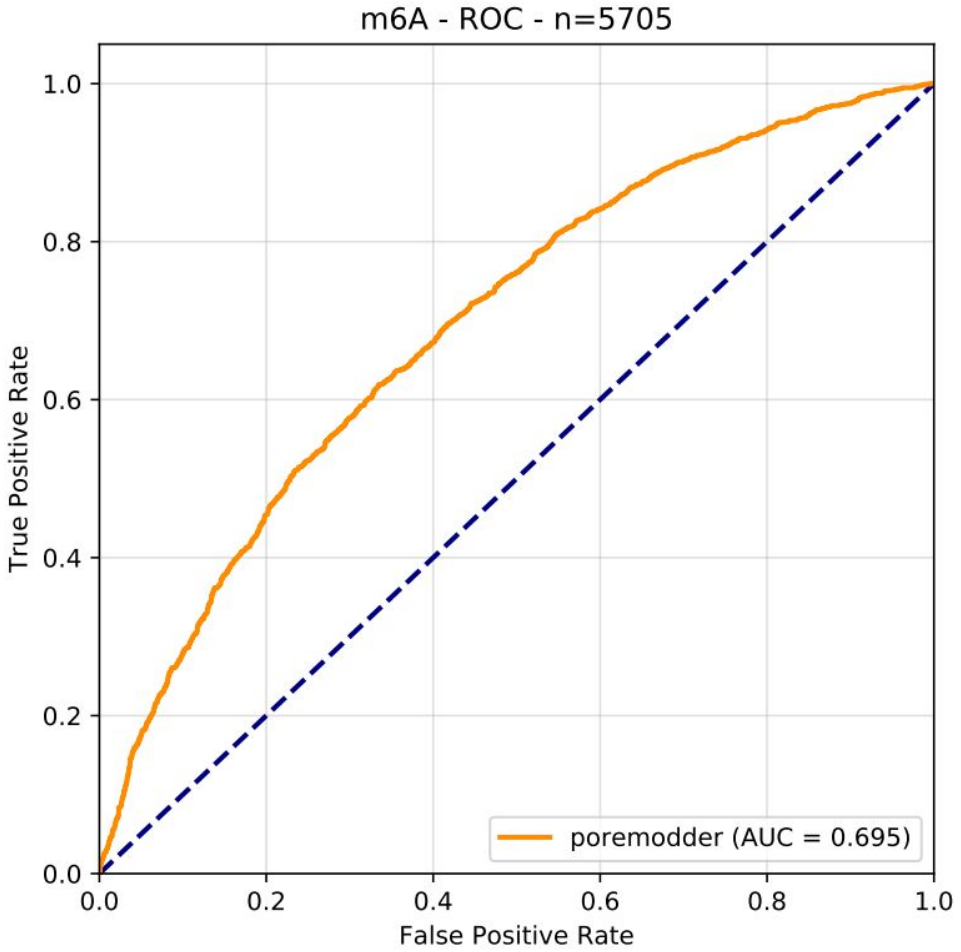
Prototype architecture using PyTorch



Results

- Trained on ~30k samples (9mers)
- Training/test split: 90/10
- Accuracy: 0.661

	Prec.	Recall	F1-score	#
Nomod	0.819	0.687	0.747	4157
m6A	0.413	0.592	0.487	1548



Results

```
[1, 3409500] loss: 0.244
              precision    recall  f1-score   support

       Nomod      0.668     0.674     0.671    778134
         m6A      0.638     0.631     0.634    707394

    accuracy                          0.653   1485528
   macro avg      0.653     0.652     0.653   1485528
weighted avg      0.653     0.653     0.653   1485528

Traceback (most recent call last):
  File "train_lstm_model.py", line 307, in <module>
    means = data['means'].T.unsqueeze(2)
  File "/home/ya86gul/miniconda3/envs/poremodder/lib/python3.7/site-packages/
    _error_if_any_worker_fails()
RuntimeError: DataLoader worker (pid 24510) is killed by signal: Killed.
```

Problems and Further Ideas

- Slow/bad data loading
- Data insufficient, e.g. k-mers with multiple As

- Data preprocessing?
- Data filtering?
- Feature selection?
- Network architecture?
- Activation functions?

Conclusion and Outlook

- Available data is problematic
- Task appears learnable

- Architecture prototyping
- Make more data!
- Punish those GPUs!

nvidia.com, kxly.com

FRIEDRICH-SCHILLER-
UNIVERSITÄT
JENA

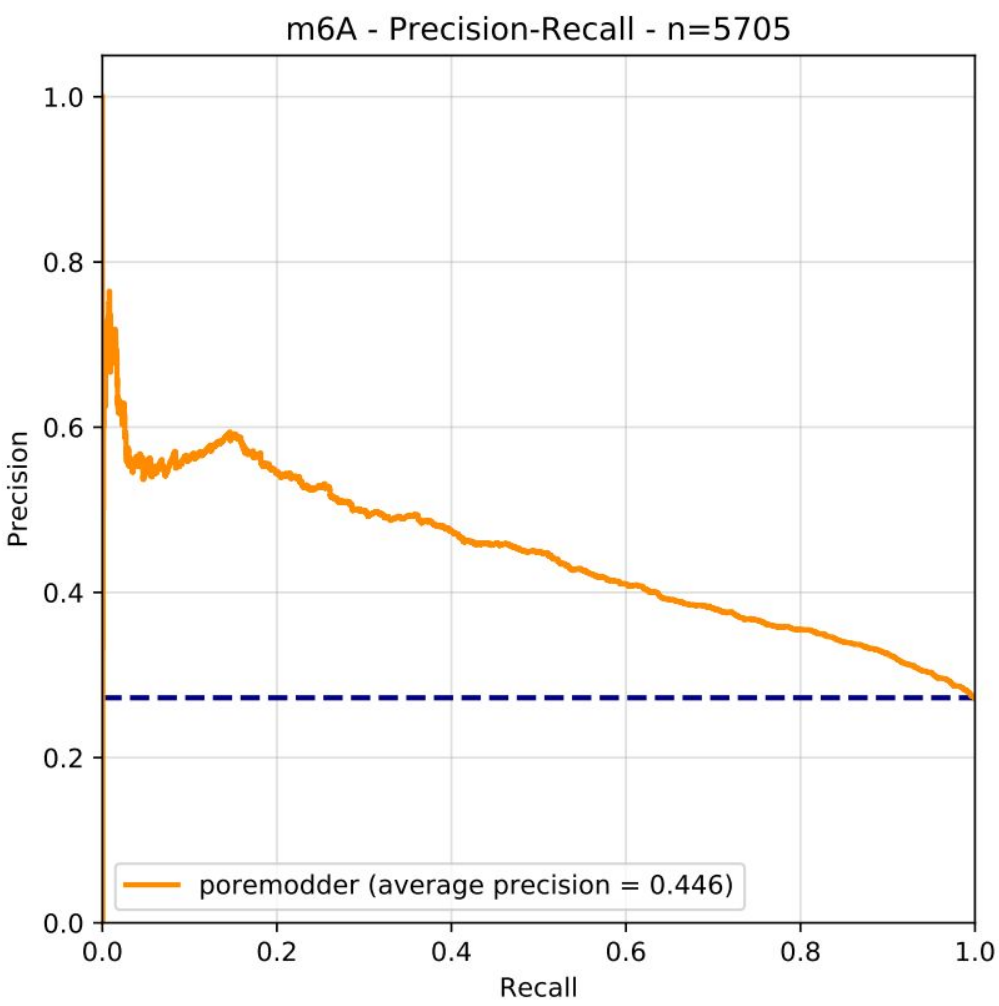
European
VIRUS BIOINFORMATICS
Center

RNA
BIOINFORMATICS

Celia Diezel Florian Mock
Manja Marz

Thank you!

SK supported by
Oxford Nanopore
Technologies bursary


m6A - Precision-Recall - n=5705
Precision
Recall
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
poremodder (average precision = 0.446)

sensitivity, **recall**, **hit rate**, or **true positive rate** (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = 1 - \mathrm{FNR}$$

specificity, **selectivity** or **true negative rate** (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{FPR}$$

precision or **positive predictive value** (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} = 1 - \mathrm{FDR}$$

negative predictive value (NPV)

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}} = 1 - \mathrm{FOR}$$

miss rate or **false negative rate** (FNR)

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}} = 1 - \mathrm{TPR}$$

fall-out or **false positive rate** (FPR)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} = 1 - \mathrm{TNR}$$

false discovery rate (FDR)

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}} = 1 - \mathrm{PPV}$$

false omission rate (FOR)

$$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}} = 1 - \mathrm{NPV}$$

```
6   def network(insize=1, size=256, winlen=19, stride=2, alphabet_info=None):
7       return Serial([
8           Convolution(insize, size, winlen, stride=stride, fun=tanh),
9           Reverse(GruMod(size, size)),
10          GruMod(size, size),
11          Reverse(GruMod(size, size)),
12          GruMod(size, size),
13          Reverse(GruMod(size, size)),
14          GlobalNormFlipFlopCatMod(size, alphabet_info),
15      ])
```


Open Assembly
Import Features
Import Enzymes
Data
Read Packing
Tag Variants
Read Colors
Visual
Zoom:
Variants:
Adjust
Page Left
Page Right
Jump to Base
Prev Feature
Next Feature
Prev View
Next View
Navigate
Read Info
RS Off
Show Cigar
Show Bases
RS Center
Read Names
RS Custom
Overlays
Contigs: 1 26.78 k reads (more)
Co... Le... Re... Fe... Mis...
cu... 1,2... 26,... 4,8... 18.5
1 to 1,297 (1.3 Kb)
1 to 419 (419 bp)
CIGAR-I
CIGAR-D
CIGAR-RIGHT-CLIP
CIGAR-LEFT-CLIP
1 U1
418 U418
Filter by:
Name

https://www.nvidia.com/content/dam/en-zz/Solutions/geforce/geforce-rtx-turing/2080/gallery/geforce-rtx-2080-gallery-c.jpg

https://www.kxly.com/content/uploads/2019/12/generic-fire_1562686082475-jpg_38950375_ver1-0-1024x576.jpg