Supplementary information: fast and accurate m6A calling using single-molecule long read sequencing

1 Supplementary Methods

1.1 Semi-supervised m6A calling algorithm

We derive m6A labels from GMM-filtered ipdRatios from ipdSummary. These labels can contain false positives. The lack of clean m6A labels makes the supervised training approach less suitable since it assumes that true labels are available. Therefore, we present a semi-supervised approach that makes a more reasonable assumption that our m6A class a mixed population of true and false positives and our non-m6A class is a clean set. Our training approach is derived from percolator and mokapot, proteomics tools for maximizing peptides identified at a target precision [1, 2].

Given a set of candidate m6A calls, our method aims to maximize the number of m6A calls at a target estimated precision. First, we split our dataset into training and validation sets stratified by class labels. Then our method proceeds in two phases. In the first phase, we use the inter-pulse distance (IPD) score of the central base as a classifier and generate an m6A classification score for all examples in the validation set. The validation examples are then ranked by the classification score, and estimated precision is computed at every score threshold. For a set of m6A (P) and non-m6A calls (N) meeting a score threshold, we define estimated precision, EP, as

$$EP = 1 - \frac{(1+c)*N}{P+N} \tag{1}$$

where c is the ratio of positive and negative labels in our validation dataset. The EP equation estimates that we have $(1 + c)*N$ total false positives, adding c unknown false positives for every known false positive found, where c is the ratio of m6A and non-m6A classes in a given dataset. Using a ranked list of EP at different score thresholds, we select the score threshold with target EP in the validation set. At the end of this phase, we have a score threshold for identifying m6A calls at the target estimated precision.

The second phase is iterative, and each iteration consists of three steps. The first step is selecting a high-confidence m6A, and non-m6A training set using a score threshold. The second step consists of training a CNN model on this training data. In the final step, a new score threshold is found by scoring the validation data using the trained CNN model from the second step as a classifier. We will now describe each step in more detail.

In the first step, a score threshold is used to select a putative m6A set. This score threshold is derived from the first iteration's IPD score classifier and a trained CNN in subsequent iterations. The training m6A set is then selected by filtering examples with non-m6A labels from the putative m6A class. These false positive examples are also removed from the non-m6A training set. At the end of this step, we have a training set selected using a given score threshold and classifier.

The second step is training a CNN model to discriminate between m6A and non-m6A examples. We initialized our model using the CNN model from the supervised approach. This transfer learning approach allowed fast convergence of our training procedure. We retained the same training hyper-parameters as the supervised approach. At the end of this step, we produce a CNN classifier trained on training data from step one.

The final step is finding a new score threshold by re-scoring the validation examples using trained CNN from step two. The initial process of estimating EP at different score thresholds is repeated, and a new score threshold with target EP is selected. In the case of a successful second phase training, the number of positives in the validation data identified at target precision increases with every iteration and plateaus when most m6A examples in the validation data have been identified. In practice, it took XX, XX and XX iterations for this curve to plateau for 2.2, 3.2 and Revio chemistry Fiber-seq experiments, respectively.

Algorithm 1 The SemiM6ACalling Algorithm The input variables are: $\mathcal{TD}$ = training data, $\mathcal{VD}$ = validation data, c = ratio of positive and negative m6A calls in $\mathcal{TD}$ and $\mathcal{VD}$, t = target precision threshold, *IPDClassifier* = m6A classifier using the IPD score of the central base, $\mathcal{I}$ = the number of iterations.

```
procedure SemiM6ACalling($\mathcal{TD}$, $\mathcal{VD}$, $c$, $t$, $IPDClassifier$, $\mathcal{I}$)
    $s_t \leftarrow$ scoreAtTargetPrecision($IPDClassifier, \mathcal{VD}, t, c$)    ▷ Compute score threshold $s_t$ at precision $t$
    for $i \leftarrow 1 \ldots \mathcal{I}$ do
        $\mathcal{TD}_t \leftarrow$ selectTrainingSet($\mathcal{TD}, s_t$)    ▷ Compute new training set $\mathcal{TD}_t$
        $\mathcal{CNN}_w \leftarrow$ trainCNN($\mathcal{TD}_t$)    ▷ Train the CNN classifier $\mathcal{CNN}_w$
        $s_t \leftarrow$ scoreAtTargetPrecision($\mathcal{CNN}_w, \mathcal{VD}, t, c$)    ▷ Recompute $s_t$
    end for
    return ($s_t, \mathcal{CNN}_w$)
end procedure
```

References

[1] William E Fondrie and William S Noble. mokapot: fast and flexible semisupervised learning for peptide detection. *Journal of Proteome Research*, 20(4):1966–1971, 2021.

[2] Lukas Käll, Jesse D Canterbury, Jason Weston, William Stafford Noble, and Michael J MacCoss. Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nature methods*, 4(11):923–925, 2007.