BIOINFORMATICS

Vol. 00 no. 00 2005
Pages 1–7

Analysis of Nanopore Data using Hidden Markov Models

Jacob Schreiber and Kevin Karplus

Nanopore Group, Department of Biomolecular Engineering, University of California Santa Cruz, California, USA.

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Associate Editor: XXXXXXX

ABSTRACT

Motivation: Nanopore-based sequencing techniques can reconstruct properties of biosequences by analyzing the sequence-dependent ionic current steps produced as biomolecules pass through a pore. Typically this involves alignment of new data to a reference, where both reference construction and alignment have been performed by hand.

Results: We propose an automated method for aligning nanopore data to a reference through the use of hidden Markov models. Several features that arise from prior processing steps and from the class of enzyme used can be simply incorporated into the model. Previously, the M2MspA nanopore was shown to be sensitive enough to distinguish between cytosine, methylcytosine, and hydroxymethylcytosine. We validated our automated methodology on a subset of that data by automatically calculating an error rate for the distinction between the three cytosine variants, and show that the automated methodology produces a 2–3% error rate, lower than the 10% error rate from previous manual segmentation and alignment.

Availability: The data, output, scripts, and tutorials replicating the analysis are available at `https://github.com/UCSCNanopore/Data/tree/master/Automation`.

Contact: karplus@soe.ucsc.edu or jmschreiber91@gmail.com

1 INTRODUCTION

The use of hidden Markov models (HMMs) in analyzing biosequence data is widespread and is the backbone of such services as pFAM, TMHMM, and SAM-T08 (Sonnhammer *et al*, 1998; Krogh *et al*, 2001; Karplus, 2009). These services use HMMs to identify regions of interest in observed sequences. Briefly, HMMs model a sequence family with emission probabilities for each possible base or amino acid at each position of a reference. HMMs can perform an alignment between an observed sequence and the reference and can allow for features such as insertions or deletions compared to the reference (Eddy, 1998).

This HMM methodology can be extended to analyze nanopore data, which is ionic current sampled at a high frequency instead of a sequence of characters (Timp *et al*, 2012). The ionic current is the flux of ions through a tiny hole (the *nanopore*) in an insulating barrier as a voltage is applied. When biomolecules pass through the nanopore, they block the passage of ions, causing characteristic drops of ionic current (Kasianowicz *et al*, 1996). More recently, both single-nucleotide resolution (Manrao *et al*, 2011) and discrimination of three epigenetic variants of cytosine (Schreiber *et al*, 2013; Laszlo *et al*, 2013) have been achieved by using Φ29 DNAP (Lieberman *et al*, 2010; Cherf *et al*, 2012) to mediate the movement of DNA through a M2MspA nanopore.

The ionic current can be processed into segments, which summarize the ionic current while a DNA molecule is held in a particular position within the nanopore by Φ29 DNAP (Schreiber *et al*, 2014). Each segment, instead of representing each nucleotide individually, is influenced by a number of adjacent nucleotides which corresponds to the length of the narrowest region of the pore. For the pore M2MspA, the word length is approximately four nucleotides.

An important task in the use of nanopore devices is decoding the sequence of segments into information about the underlying biosequence that generated it, which is complicated because the DNA motor is not a perfect stepper motor and the segmentation algorithm is not perfect. However, the combined behavior of the enzyme, nanopore, and signal processing can be modelled with a somewhat more complicated HMM than the simple profile HMMs usually used in protein and DNA sequence modelling. The HMM presented here is specific to using Φ29 DNAP with a M2MspA pore and using our current segmentation algorithm. Different classes of enzymes may have different features that need to be modelled, and different segmenters may have different rates of over- or under-segmentation.

We propose an automated way to align nanopore data to a reference model using this HMM. We validate this approach using a three-class classification problem, in which the strands contain either a single cytosine (C), methylcytosine (mC), or hydroxymethylcytosine (hmC) in an otherwise-identical sequence. These single cytosine variants have been shown to produce different ionic current levels as they pass through the nanopore (Laszlo *et al*, 2013). Hand analysis of data from the DNA molecules tested in this paper resulted in a classification error rate of approximately 10%. (Schreiber *et al*, 2013). We show that this error rate can be reduced to 2–3% using our automated methodology on a small data set.

2 MODULE STRUCTURE

A traditional character-emitting profile HMM models a reference sequence with a linear sequence of *match* states, each with probabilities for the characters dependent on the position in the sequence. They model insertions by adding another character-emitting state after each position and deletions by adding a silent state at each position (Fig. 1a). Transitions between the match, insert, and delete states indicate how an observed sequence may

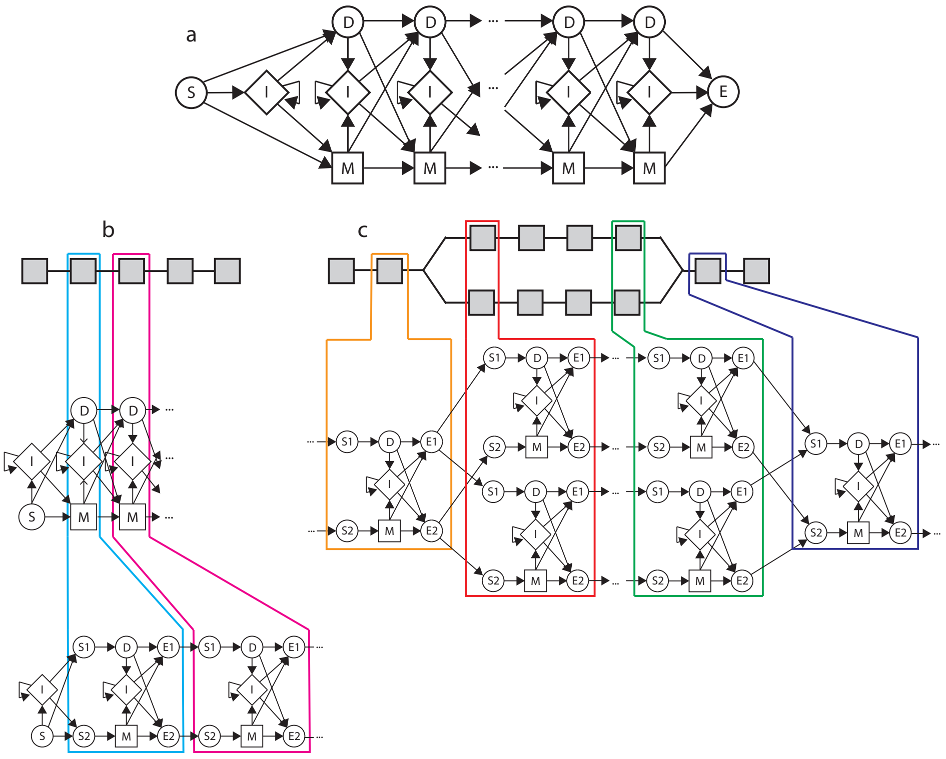

Fig. 1. An example global sequence alignment HMM. (a) The typical structure of a global sequence alignment HMM where each match represents a position in a reference. Insertions and deletions in an observed sequence are allowed through a symbol-emitting insert state and a silent delete state, and mismatches can be allowed for through pseudocounts on emission distribution of the match state. (b) A module structure can be made by taking slices from the global sequence alignment HMM, and adding silent states as ports, each modelling a single symbol in the reference sequence. (c) Creating a HMM with a fork is simple with this module format, as the modules can be attached at the ports using a small number of edges without having to change internal transition probabilities within the modules.

be aligned to the model, with transition probabilities indicating the likelihood of each possible transition.

We can view the HMM graph structure as composed of repeating subunits, consisting of a match, delete, and insert state and their associated edges. We can minimize inter-module connections by adding silent states to act as *ports* for the modules, with single transitions of probability 1 between the ports out of one module and into the next. (Fig. 1b) These extra silent states do not add to the computational complexity of the HMM and are automatically optimized out by the YAHMM software we used.

Alignment to simple profile HMMs suffices for many recognition tasks, but some classification tasks are better handled by forks within the HMM, where the different paths chosen at the fork determine the classification. Our module format with ports allows complicated forking paths without needing excessive numbers of edges. Fig. 1c shows an example fork containing two paths in an otherwise linear sequence. Note that only four edges are required on either side to create the fork structure. In general, $2nm$ edges are needed for a module structure with n silent-state ports and a fork with m paths, and the edges where the fork rejoins all have probability 1, thus not adding to computational complexity. The internal transition probabilities of each module are kept entirely separate from the presence of a fork.

To model nanopore data, we first perform event detection on the data, detecting all regions of ionic current which are longer than 500 ms, below 90 pA, and above 0 pA. We then segment each event by recursively splitting at the ionic current sample which best splits a region into two Gaussian distributions until a threshold in probabilistic gain is reached, representing each time interval as a *segment* with a mean current, a standard deviation, and a duration (Schreiber *et al*, 2014). Although all three parameters carry information, we built our HMM to model only the mean values to match more closely the previously done hand analysis—it is likely that using the other information would improve the alignments slightly. Each match state in our HMM uses a Gaussian distribution to assign emission probabilities to the segments, with parameters μ and σ having initial values derived from a hand-analyzed reference sequence. Insert states, which correspond to unpredicted currents, have a uniform distribution from 0 pA to 90 pA, which are the limits for event detection. Initial transition probabilities inside each module were estimated by hand from a small number of events.

Our HMM (Fig. 2) has a more complicated module than the standard modules for profile HMMs, in order to capture variations in the signals due to both signal-processing limitations and non-ideal behavior of the Φ29 DNAP motor moving the strand through the pore. Ionic current traces giving examples of what is being modelled by the various additions to the modules are shown color coded below

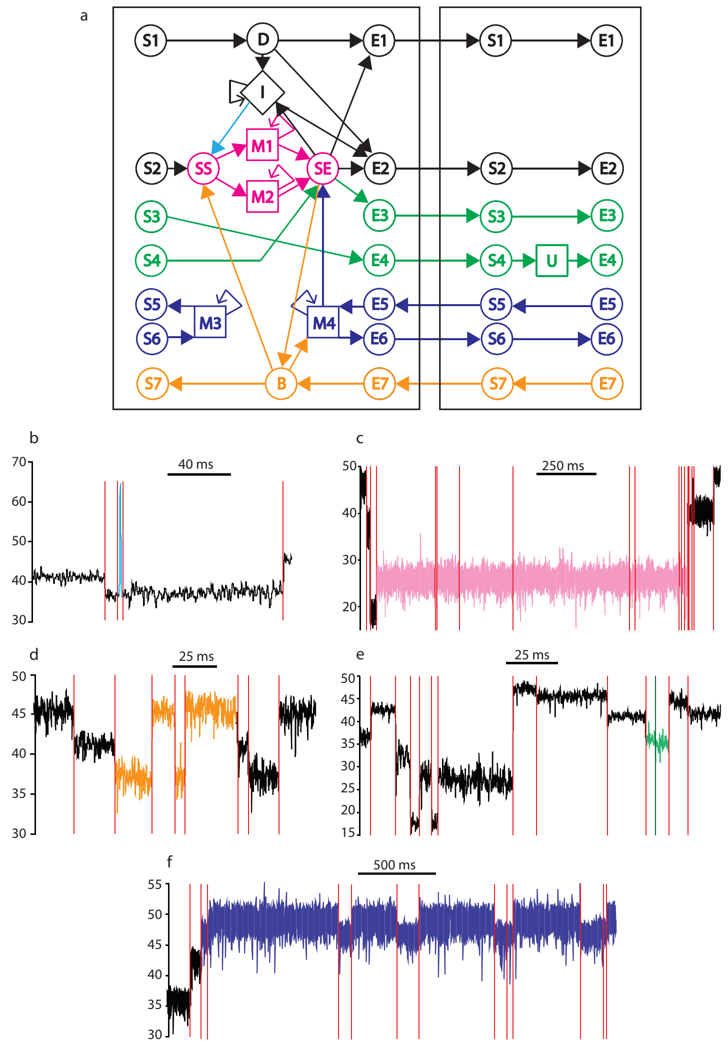

Fig. 2. The repeating structure of an HMM that incorporates features specific to DNA translocation by Φ29 DNAP. (a) The two modules that form the repeating structure, of which the left represents one position in a PSSM, and the right represents two adjacent positions in the reference profile. Five new features are proposed in comparison to the typical global sequence alignment HMM; (b) in teal, inserts can now transition back to the match state they came from, to allow for a segment to be split by a transient spike, (c) in magenta two match states, one with a low-probability self loop and one with a high-probability self loop, to handle oversegmentation, (d) in orange, the possibility of a backslip with an exponentially decreasing probability with the number of positions slipped, (e) in green, handling for undersegmentation by including a state with parameters derived from the two adjacent states that were to be modelled, and lastly (f) in blue the possibility of repeateded single-position backslips, usually associated with a temporary stall in the incorporation of a nucleotide on the part of the Φ29 DNAP motor.

the HMM module. The features that pertain to the signal processing and noise spikes in the nanopore are useful in all nanopore HMMs, but those that model Φ29 DNAP behavior on the M2MspA pore may need to be modified for different pores and other DNA motors.

The segmenter can *oversegment*, breaking an interval of current into multiple segments, even though the DNA is not advancing to the next nucleotide, or *undersegment*, failing to distinguish current values that correspond to different positions on the DNA strand. Oversegmentation is fairly rare, but tends to occur in bursts with either one or many segments. To approximate this distribution, we used two match states (M1 and M2 in magenta in Fig. 2), each with a self-loop. One models the initial peak of the distribution, while the other models the long tail. Undersegmentation is also fairly rare, and is modeled by creating a new state U (green in Fig. 2) corresponding to two adjacent match states, with intermediate μ and σ values. We treated it as a separate module for convenience in creating the HMM, since its emission values depend on two adjacent positions in the sequence. The extra silent states are optimized out by the YAHMM software, so this notational convenience does not cost us anything.

Nanopore signals often have unexplained short blips, where the current goes momentarily high or low before returning to the same mean current. These may be electronic artifacts or caused a small molecule transiting the nanopore along with the DNA. We model them by including a low-probability edge from the insert state I back to the silent state before the match states (teal in Fig. 2). A possible improvement to this model would be to include a special "blip" insert state that modeled duration as well as mean, given the short duration of these blips.

The Φ29 DNAP enzyme often slips backwards due to the tension applied by the voltage across the nanopore. The *backslips* are modeled by edges to and from silent state B (orange in Fig. 2). Note that the transitions along this bottom path are right-to-left instead of left-to-right.

We also occasionally observe a rapid *flicker*, switching back and forth between the current values of two adjacent states. We believe that this results from the enzyme repeatedly attempting and failing to incorporate a nucleotide. The match states M3 and M4 (blue in Fig. 2) correspond to flickers with the previous and next state respectively.

The four match states in each module initially have the same underlying distributions—they all represent the same position being read in the nanopore, and the separation into different states handles the different transition probabilities. However, the underlying distributions are not tied, meaning that distributions can deviate from each other after training.

3 CLASSIFICATION OF METHYLATION STATUS

The nucleotide cytosine (C) undergoes a methylation cycle as a part of biological cell regulation. During the cycle, cytosine is enzymatically converted to methylcytosine (mC), then to hydroxy-methylcytosine (hmC), formylcytosine, or carboxylcytosine by Tet proteins (Shinsuke *et al*, 2011). Our group has previously shown that DNA nanopores can distinguish between a single C, mC, or hmC present in a CG dinucleotide using hand feature extraction and classical machine learning methods with an estimated error rate of 2% to 10% depending on the context of the flanking nucleotides (Schreiber *et al*, 2013). The $5'$-CCGG-$3'$ context had the highest error rate, at 10%.

Briefly, all three DNA hybrids (see Supp. Fig. 1) are added to the well of solution above the nanopore. These hybrids are composed of the template strand to be read, a hairpin primer with a free $3'$ hydroxyl, and a blocking oligomer designed to allow enzyme binding to the template strand, but block access to the $3'$ hydroxyl. When the DNA-enzyme complex reaches the nanopore, the $5'$ end of the template threads through the nanopore, and the blocking oligomer is pulled off one nucleotide at a time by the voltage using the enzyme as a wedge—moving the template strand through the porin. This causes the unzipping fork seen in Fig. 3. When the blocking oligomer is fully removed, the $3'$ hydroxyl becomes available to the enzyme, and strand replication begins, pulling the template strand back up through the nanopore one nucleotide at a time, at which point the second reading of the cytosine occurs, as well as the reading of the label and the remainder of the strand. The sequence of the template strand is a poly-$3'$-CAT-$5'$ sequence, with a $3'$-CCGG-$5'$ 4mer inserted into it, and labels consisting of either an abasic residue, 'GG' dinucleotide, or no modification 9 residues downstream from the 4mer insertion.

We validated our automated methodology on new data collected for this difficult $5'$-CCGG-$3'$ context. The data was collected from running a mixed pool of strands through the nanopore. The strands bore a single C, mC, or hmC at the target CG dinucleotide and the corresponding downstream label in an otherwise identical sequence. The first 27 events were identified by hand (5 C, 11 mC, and 11 hmC) and used to create initial profiles for the three cytosine variants. These events were not used in any further analysis.

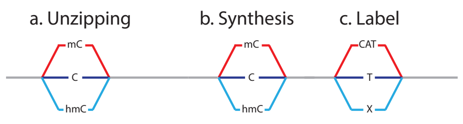

Fig. 3. The general structure of the hidden Markov model used, which models 54 segments. (a) The first 18 segments arise as Φ29 DNAP unzips the blocking oligomer on the hybrid, during which time the cytosine variant passes through the nanopore once. (b) The remaining segments arise as Φ29 DNAP performs synthesis on the growing daughter strand, pulling the cytosine variant back up through the nanopore, and (c) later the corresponding label. The label name refers to the sequence inserted into the repeating CAT background sequence, with an abasic residue denoted by an 'X' corresponding to hmC, a thymine reside corresponding to an unmodified cytosine, and an unmodified CAT trinucleotide indicated a methylated cytosine.

The profiles from the hand-curated sequences were merged to create an HMM with three forks (Fig. 3), with each fork containing three paths. The first fork models the three cytosine variants (C, mC, hmC) passing through the nanopore during unzipping (Fig. 3a), the second the cytosine variants during synthesis (Fig. 3b), and the third the labels (designated X, T, or CAT) passing through the nanopore (Fig. 3c) (Schreiber *et al*, 2013). Each fork merges before the next fork splits in order to ensure that the calls for each fork are approximately independent of each other.

After excluding the 27 events used for constructing the initial model, 423 more events were automatically detected, segmented, and reduced to a sequence of means of the ionic current segments. Because we expect the HMM approach to handle oversegment-ation better than it handles undersegmentation, extra preprocessing was done to reduce the probability of undersegmentation. This preprocessing consisted of calculating the Viterbi path of each sequence in the untrained HMM, and splitting segments that aligned to an undersegmentation state U (the splitting was done by the same segmenter as the initial segmentation, just forcing a further split in the segment, rather than using the normal termination condition). The align-and-split-undersegmented-segments process was repeated until either the number of undersegmentation states in the path did not decrease from the previous iteration, or none were present.

In previous hand analysis, we distinguished between *on-pathway* events, which consist of Φ29 DNAP moving a DNA strand moving in a stepwise manner through the nanopore and terminating with the complex dissociating from the nanopore at the end of synthesis, and *off-pathway* events, where either something is passing through the nanopore that is not DNA, or the Φ29 DNAP has stalled in some manner. Our analysis consists of two classification tasks: deciding which events are on-pathway and then classifying the cytosine variant for the on-pathway events. We use the same HMM in diffferent ways for the two tasks.

In order to filter and classify events, we calculate an $m \times m$ matrix T for a model with m states, where $T_{s,t}$ is the expected number of transitions from state s to state t given the input event, as calculated by the forward-backward algorithm. The score of each path at a fork S_p is calculated as the minimum number of expected transitions to the magenta states SE shown in Fig. 2 for the modules on the path p:

$$S_p = \min_{i \in p} \sum_s T_{s,\mathrm{SE}_i} \tag{1}$$

Observed sequences that match a path in its entirety score high, while observed sequences that omit segments in the path will have a lower score. Soft classification of the cytosine residue is done by normalizing the S_c values for the three paths $\{C, mC, hmC\}$ of the synthesis fork. We do not use the path taken at the unzipping fork for classification or for filtering, to allow a direct comparison to previous work where only synthesis data was used, even though using the information in the unzipping fork might improve the classification.

To distinguish between on-pathway events and off-pathway events, a filter score F is calculated as

$$F = \sum_{c \in \{C,mC,hmC\}} S_c \sum_{l \in \{X,T,CAT\}} S_l \tag{2}$$

which requires that a path c has high probability at the cytosine synthesis fork, and a path l has high probability at the label fork, but does not require that the paths match.

To determine the accuracy of our soft calls, we compute a soft accuracy score A as the dot product between the score of a cytosine variant path and its corresponding label, normalized by the filter score for that event, which can be loosely interpreted as the probability that an alignment path through the HMM takes a path with consistent cytosine and label choices:

$$A = \frac{(S_C, S_{mC}, S_{hmC}) \cdot (S_T, S_{CAT}, S_X)}{F} \tag{3}$$

For the data in our experiments, the accuracy A is usually near 1 (a clearly correct call) or 0 (a clearly incorrect call), with few ambiguous results (see Fig. 6).

The 423 events were split into three sets: 207 (49%) in a training set, 89 (21%) in a cross-training set, and 127 (30%) in a test set. The test set was set aside and all training and model development done on the training and cross-training sets.

The first step of training was to determine a threshold for the filter score to remove off-pathway events from training. For a variety of thresholds ranging from 0.01 to 0.9, events in the training set that scored higher than the threshold were used for 10 iterations of Baum-Welch training, with edge pseudocounts equal to the initial probability of that edge.

The events in the cross-training set were then scored using the trained model and sorted by filter score, so that $F(\mathrm{Cross}_i) \geq F(\mathrm{Cross}_{i+1})$. We define cumulative average soft call accuracy on the cross-training set as

$$\mathrm{Acc}_i = \frac{1}{i} \sum_{j=1}^{j} A(\mathrm{Cross}_j) \tag{4}$$

All the thresholds produce similar plots of average accuracy (Fig. 4), indicating a robustness of the HMM training to inclusion of some off-pathway events in the training set. The thresholds $F \geq 0.1$ and $F \geq 0.05$ had the highest areas under the curve and were very close to each other, and so 0.1 was chosen for subsequent training due to its better performance on good events (those with high filter scores).

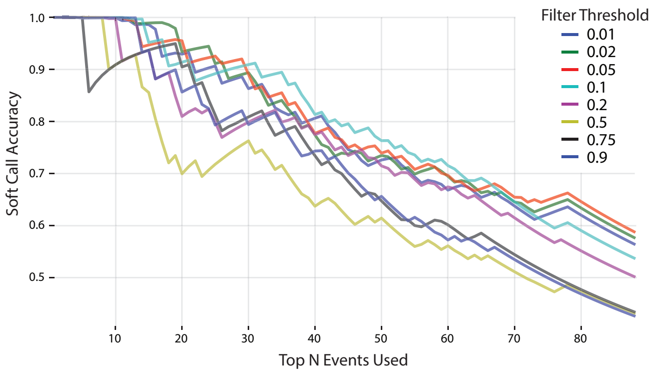

Fig. 4. Accuracy measurements using the cross-training set with different thresholds on F of events used to train from the training set. A range of filter thresholds were used on F calculated by the untrained model on the training set to determine which events to use for training. The accuracy of using the trained model on a cross-training set is shown. While all thresholds seem to perform similarly, those above 0.5 seem to perform worse than those under it, indicating some contamination of the model by off-pathway events in the training set when too high a threshold is used.

To analyze the structure of the HMM, the events from the training set with $F \geq 0.1$ were used to train the model and events from the cross-training set was characterized using it. Both the untrained and trained model are included in the supporting information. The average expected number of transitions to each insert, delete, backslip, and undersegmentation state was calculated across the HMM using the same transition matrix T (Fig. 5). Three interesting phenomena are shown: (1) a large number of inserts at position 9 indicating that we are probably missing a state here or that the enzyme commonly stalls at this position, (2) a large number of backslips at position 17 followed by an increase in deletes at positions 18 and 19, which is where the enzyme's catalytic activity begins, indicates that our hand-selection of the sequence of states here may be poor, and (3) a high prevalance of all non-match states at the end of the HMM, due to strands dissociating from the nanopore at different times after the majority of the daughter strand has been replicated. We decided not to change the structure of the HMM, despite this analysis of possible improvements, as the HMM had already been evaluated on the test set, which can only be used once.

We evaluated the HMM method by training an HMM on the combined training and cross-training sets and computed the mean cumulative soft call (MCSC) accuracy on the test set. The filter score F was calculated for each of the 296 events in the combined training set using the untrained HMM, and events with $F < 0.1$ were removed, leaving 135 events. The HMM was trained on these events using a maximum of 10 iterations of Baum-Welch training with edge pseudocounts equal to the initial probability of the transitions. All events in the test set were scored, and both MCSC accuracy Acc_i and filter score F on the test set were plotted vs. rank of filter score F (Fig. 6a).

In hand analysis of similar data, 26% of all detected events were chosen as on-pathway. If we set a threshold on F to select the top 26% of events as on-pathway, we get an accuracy of approximately 98%, compared to a hand accuracy of approximately 90% (Schreiber *et al*, 2013). Note: this improvement may in part

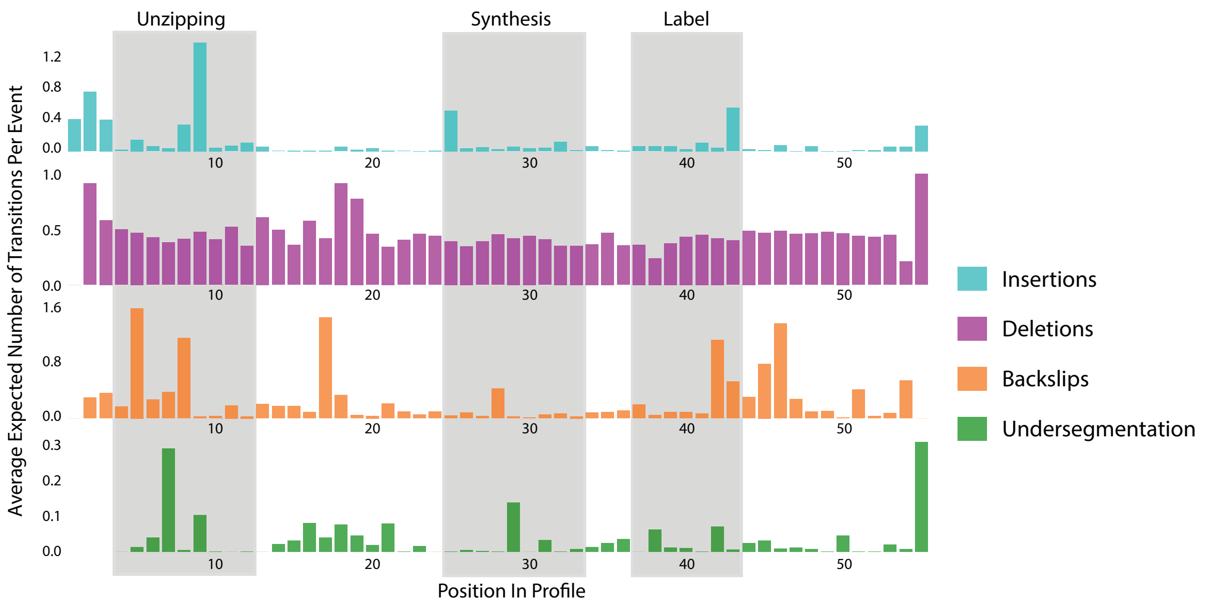

Fig. 5. The expected number of deletions, insertions, backslips, and undersegmentations in each position across the HMM. Each value is calculated by summing the transition matrix calculated using the forward-backward algorithm for each respective state in each module in the HMM. Three phenomena of note are (1) the spike of inserts at position 9, indicating either our need to include another state, or a location that the enzyme is likely to stall briefly at, (2) the spike of backslips at position 17 followed by increased deletes at position 18 and 19, indicating that the observed sequences had a slight preference to align to states before position 17 than positions 18 or 19, and (3) the general uptick of these occurances at the end of the sequence, indicative that the end of the strand may be ambiguous due to variable dissociation times.

reflect reversion to the mean, as the context chosen was the one with the worst results in the hand analysis, so new data would likely not be as difficult. But overall results from all contexts in the hand analysis had only 93.6% accuracy using the best classifier, and our new results are better even than that. It is not clear whether the improvement comes from better selection of on-pathway events than hand curation or from better alignment and classification, but either way there is much less risk of unconscious confirmation bias with the automated method. The automated method tends to call off-pathway events as hmC with a random label, perhaps because the hmC path is closest to the mean ionic current, allowing noise to align to it better (Supp. Fig. 2).

If instead of taking a cumulative average, we use a moving-window average, we can see that the events with a high filter score are all classified correctly and that ones with a low filter score are mostly classified incorrectly (worse than random calls) (Fig. 6b). The filter score does a good job of distinguishing classfiable on-pathway events from unclassifiable off-pathway ones, and the HMM does a good of classifying the on-pathway events.

To ensure that our estimation of accuracy was not just a lucky train-test split, further 5-fold cross-validation was performed. The events were divided randomly into five sets, and an HMM was tested on each set after being trained on the the events in the other four that had $F \geq 0.1$. The 5-fold cross-validation was repeated 10 times, shuffling the order of events each time, and the mean and range are shown in Fig. 7. The mean accuracy is 97% when using the top 110 events (26%) by filter score, which is consistent with the accuracy reported by the pure train-test split.

The full pipeline of event detection, segmentation, and classification is fast. The process of detecting 52 events, segmenting them,

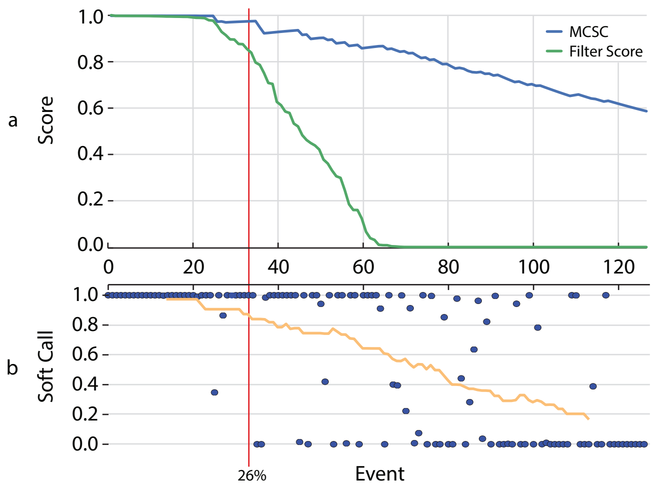

Fig. 6. Soft call accuracy measurements using the trained model on testing data. (a) The MCSC accuracy and filter scores are shown plotted for each event, ordered by their filter score. The MCSC accuracy represents the accuracy expected using events with this filter score or higher. As more events are included, the accuracy goes down. The top 26% of events (to the left of the red line) give ≈98% accuracy. (b) Soft call accuracy for each event is plotted, sorted by the filter score just as in part a. Only a few events have a soft call accuracy between ≈0 and ≈1, and sorting by filter score gives good discrimination between events with high accuracy and those with low accuracy. The orange line is the rolling mean using a 31-event window with 15 events on each side, and indicates the expected soft call accuracy of events with that filter score.

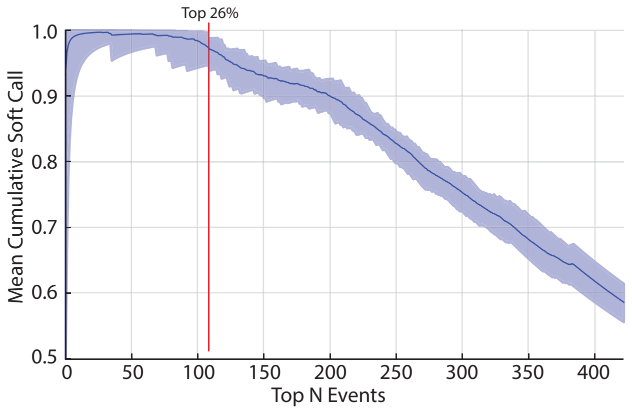

Fig. 7. The mean and range of cumulative soft call accuracies when running 5-fold cross validation 10 times. The mean is shown by the blue line, and the shaded area represents the maximum and minimum accuracy when using the top N events by filter score. The red line indicates using the top 26% of data (the fraction of similar data selected as on-pathway in hand-curated experiments), which gives a mean accuracy of 97%.

preprocessing the data using the untrained model, and scoring each event using the trained model took ≈48 seconds on a Intel Core i5-3470 CPU clocked at 3.2GHz for ≈461 seconds of event data, or ≈0.9 seconds per event that on average lasted ≈8.5 seconds. This makes the entire process a little over 9× faster than real time. Running just forward-backward on all 423 preprocessed events, calculating a filter score and a soft call score, took ≈95 seconds on ≈2736 seconds of data, making that component by itself ≈ 29× faster than real time. Training the HMM using 10 iterations of Baum-Welch on 155 events takes ≈700 seconds, making it unlikely to be consistently faster than real time, but training is usually not required to be done in real time.

4 CONCLUSION

The complexities of nanopore data have made hand curation of data popular, leading to questions of unconscious bias in data selection. By modeling many of these complexities explicitly with HMMs, we were able to build an automated classifier with performance at least as good as with previous hand-curated data sets.

In addition to the typical insertions and deletions of profile HMMs, we modeled the backslipping and enzyme stalling that are common occurrences when using an enzyme motor, and we incorporated segmenter-specific features, such as oversegmentation and undersegmentation, that are independent of the attributes of the DNA motor enzyme.

We proposed a modular structure for an HMM that handles each of these features, making construction of new HMMs for known sequences straightforward.

This modular structure was validated by automating the classification of DNA strands as bearing a single cytosine, methylcytosine, or hydroxymethylcytosine and calculating an error rate by comparing the call at the cytosine to an independent downstream marker. The results were substantially better than analysis of hand-curated data for the same sequence context (2% error rate instead of 10%), without the risk of unconscious bias in selection of on-pathway events, and the whole pipeline works significantly faster than real time.

Future areas of research include an automated way of building HMMs to recognize known sequences and classify cytosine variants in them. Our current HMM-building techniques require data collected from the molecule being studied, but we should be able to combine data from previously studied DNA molecules to predict the mean current values needed for the states of an HMM for known DNA sequences. These current values depend on both the pore and the DNA motor, so need to be collected for each new enzyme-pore combination used.

An automated base-caller HMM for sequencing unknown DNA would be approximately twice the complexity as an HMM for a sequence of length 4^n, where n is the word length of the pore-motor combination being used. For $n = 4$, such an HMM would be about five times the size of the HMM used in this paper, and so should still be runnable in real time.

Doing methylation calls at specified loci on a known reference sequence can be done by extending the model used here, which had three variable loci, allowing the methylation landscape to be probed on a single-molecule basis. Both base-calling and methylation calling on known references would involve using the modular structure we proposed, attached to each other in different ways.

ACKNOWLEDGEMENT

The authors would like to acknowledge Zachary Wescoe, who ran the nanopore experiments to produce the data analyzed in this paper.

Funding: This work was partially supported by the National Human Genome Research Institute (grant HG006321-02)

REFERENCES

Cherf,G. *et al.* (2012) Automated forward and reverse ratcheting of DNA in a nanopore at 5-Å precision, *Nature Biotechnology*, 30, 344-348.

Eddy,SR. (1998) Profile hidden Markov models., *Bioinformatics*, 14(9), 755-763.

Karplus,K. (2009) HMM-based Protein Structure Prediction, *Nucl. Acids Res.*, 37(Suppl.2), W492-E497

Kasianowicz,J. *et al.* (1996) Characterization of individual polynucleotide molecules using a membrane channel, *Proc Natl Acad Sci*, 93(24), 13770-13773.

Krogh,A. *et al.* (2001) Predicting transmembrane protein topology with a hidden Markov model: application to complete genomes, *J. Mol. Biol.*, 305(3), 567-580.

Laszlo,A. *et al.* (2013) Detection and mapping of 5-methylcytosine and 5-hydroxymethylcytosine with nanopore MspA, *Proc Natl Acad Sci*, 110(47), 18904-18909.

Lieberman,K. *et al.* (2010) Processive replication of single DNA molecules in a nanopore catalyzed by phi29 DNA polymerase, *J. Am. Chem. Soc.*, 132(50), 17961-17972.

Manrao,E. *et al.* (2011) Nucleotide discrimination with DNA immobilized in the MspA nanopore, *PLoS One*, 6(10), e25723.

Schreiber,J. *et al.* (2013) Error rates for nanopore discrimination among cytosine, methylcytosine, and hydroxymethylcytosine along individual DNA strands, *Proc Natl Acad Sci*, 110(47), 18910-18915.

Schreiber,J and Karplus,K. (2014) Segmentation of noisy signals generated by a nanopore, *Bioinformatics*, (SUBMITTED).

Shinsuke,I. *et al.* (2011) Tet proteins can convert 5-methylcytosine to 5-formylcytosine and 5-carboxylcytosine, *Science*, 333(6047), 1300-1303.

Sonnhammer,E. *et al.* (1998) Pfam: multiple sequence alignments and HMM-profiles of protein domains, *Nucl. Acids Res.*, 26(1), 320-322.

Timp,W. *et al.* (2012) DNA base-calling from a nanopore using a Viterbi algorithm, *Biophys J.* 102(10), L37-9.