Scoring Consensus of Multiple ECG Annotators by Optimal Sequence Alignment

Masoumeh Haghpanahi[1], Reza Sameni[2] and David A. Borkholder[3]

Abstract— **Development of ECG delineation algorithms has been an area of intense research in the field of computational cardiology for the past few decades. However, devising evaluation techniques for scoring and/or merging the results of such algorithms, both in the presence or absence of gold standards, still remains as a challenge. This is mainly due to existence of missed or erroneous determination of fiducial points in the results of different annotation algorithms. The discrepancy between different annotators increases when the reference signal includes arrhythmias or significant noise and its morphology deviates from a clean ECG signal. In this work, we propose a new approach to evaluate and compare the results of different annotators under such conditions. Specifically, we use sequence alignment techniques similar to those used in bioinformatics for the alignment of gene sequences. Our approach is based on dynamic programming where adequate mismatch penalties, depending on the type of the fiducial point and the underlying signal, are defined to optimally align the annotation sequences. We also discuss how to extend the algorithm for more than two sequences by using suitable data structures to align multiple annotation sequences with each other. Once the sequences are aligned, different heuristics are devised to evaluate the performance against a gold standard annotation, or to merge the results of multiple annotations when no gold standard exists.**

I. Introduction

The analysis of ECG signals is of great importance in the detection of cardiac anomalies. The advances in signal processing and machine learning techniques have resulted in numerous automated ECG beat detectors and classifiers in the past few decades. Evaluating the performance of such algorithms requires devise of suitable scoring functions that compare the resulting annotation sequences with a gold standard or with other existing annotation sequences. However, the existence of missed or erroneous determination of fiducial points in the results of different annotators, coupled with the finite accuracy of measurement systems, makes comparison of the annotation sequences a challenge.

In practice, root mean square error (RMSE) and Pearsons correlation coefficient are widely used measures [1] to compare the results of different annotators with the reference sequence, in the presence of a gold standard. On the other hand, different deterministic (such as mean, median, and majority voting) or probabilistic (such as bayesian voting) methods [1], [2] are applied to merge the results of different annotators and estimate the true (hidden) labels, in the absence of a gold standard.

The implicit assumption in all these methods is that the sequences are aligned with each other; hence, the problem is formulated as having a data set $D = \{y_i^{(1)}, y_i^{(2)}, \cdots, y_i^{(N)}\}_{i=1}^{L}$ consisting of N different annotation sequences $\{y_i^{(j)}\}_{i=1}^{L}, j = 1, \cdots, N$ of length L, with the goal of comparing the annotators with or estimating the gold standard sequence $\{y_i^*\}_{i=1}^{L}$.

In this work, we discuss how different annotation sequences can be optimally aligned using a sequence alignment technique similar to the one used for aligning biological sequences [3]. Once annotation sequences are aligned, a scoring function is devised to measure the similarity between annotation sequences with their corresponding reference responses. To show the importance of sequence alignment, the performance of the devised scoring function is further tested against an RMSE similarity measure in scoring the performance of automated fetal QRS (fQRS) annotators. The final results indicate that sequence alignment enables more robust evaluation of the performance of ECG annotators. Finally, the concept of sequence alignment is extended to multiple sequences by using a *union-find* data structure. This provides a means to merge the information provided by multiple annotators and estimate the true (hidden) annotations from the merged information of the annotation sequences.

In what follows, the notion of "optimal alignment" is first defined for two DNA sequences and the dynamic programming approach used to find an optimal alignment between two such sequences is described in Section II. In Section III, gene sequences are compared with ECG annotation sequences, and the alignment methodology used for aligning gene sequences is modified accordingly to find the optimal alignment of annotation sequences. Finally, the importance of sequence alignment prior to measuring the similarity between multiple annotation sequences is shown in Section IV.

II. Background

In this section, we define the concept of "global alignment" for two gene sequences and describe the methodology for computing their optimal alignment via dynamic programming. Next, we briefly mention how the pairwise sequence alignment methodology can be extended to multiple sequence alignment.

Consider the case of two gene sequences $X = X_1 \cdots X_m$ of length m, and $Y = Y_1 \cdots Y_n$ of length n, defined on a finite alphabet set of $\Sigma = \{A, C, G, T\}$; i.e., X_i and $Y_j \in$

[1] M. Haghpanahi is a postdoctoral fellow at the Electrical Engineering department, Rochester Institute of Technology, USA mxheen@rit.edu
[2] R. Sameni is a faculty member of the School of Electrical Engineering, Shiraz University, Shiraz, Iran rsameni@shirazu.ac.ir
[3] D.A. Borkholder is the Bausch and Lomb associate professor of Microsystems Engineering, Rochester Institute of Technology, USA david.borkholder@rit.edu

$\Sigma, \forall 1 \leq i \leq m$ and $1 \leq j \leq n$. Informally, an alignment of two sequences X and Y, denoted by $X^\diamond$ and $Y^\diamond$, is obtained by first inserting gaps either into or at the ends of X and Y such that the length of the sequences match (i.e., $|X^\diamond| = |Y^\diamond|$), and then establishing a one-to-one relation between the elements or gaps at the corresponding indices of the aligned sequences.

While many different alignments exist between two sequences, optimal alignment is defined with respect to an alignment quality and with the goal of minimizing a cost function. Let $\sigma(x, y)$ be the weight of the alignment of elements x and y in the set $(\Sigma \cup \{-\})$, where $\{-\}$ is the character used to denote gaps in the aligned sequences. The total cost of an alignment is then defined as

$$W(X^\diamond, Y^\diamond) = \sum_{l=1}^{|X^\diamond|} \sigma(X_l^\diamond, Y_l^\diamond).$$

The optimal alignment and the minimum cost of aligning X and Y, with respect to a given weight function σ, can then be formulated as

$$\begin{aligned} X_{\text{opt}}^\diamond, Y_{\text{opt}}^\diamond &= \underset{X^\diamond, Y^\diamond}{\operatorname{argmin}} \{W(X^\diamond, Y^\diamond) | (X^\diamond, Y^\diamond) \text{ is alignment of } \\ &\quad (X, Y)\}, \text{ and} \qquad (1) \\ D(X, Y) &= \min_{X^\diamond, Y^\diamond} \{W(X^\diamond, Y^\diamond) | (X^\diamond, Y^\diamond) \text{ is alignment of } (X, Y)\}, \end{aligned}$$

respectively.

For a given weight function $\sigma(\cdot)$, the algorithm known as the *Needleman-Wunsch* algorithm can be used to find the optimal alignments of equation (1) via dynamic programming [3]. The key insight to using the dynamic programming framework is that the optimal alignments can be computed in a step-wise fashion by making locally optimal choices based on the results from smaller subproblems. Specifically, the algorithm defines the minimum cost of subsequences $X_{1\cdots i} = \{X_1, \cdots, X_i\}$ and $Y_{1\cdots j} = \{Y_1, \cdots, Y_j\}$ as partial solutions and creates a matrix of these partial solutions, referred to as the alignment matrix A. Hence, the alignment matrix $A \in \mathbf{R}^{(|X|+1)\times(|Y|+1)}$, and $A_{i,j} = D(X_{1\cdots i}, Y_{1\cdots j})$. Arguing by contradiction, it can be shown that the following recursive relation holds for the alignment matrix A:

$$A_{i,j} = \min \begin{cases} A_{i-1,j-1} + \sigma(X_i, Y_j) \\ A_{i-1,j} + \sigma(X_i, -) \\ A_{i,j-1} + \sigma(-, Y_j) \end{cases}, \text{ for } \begin{array}{l} 1 \leq i \leq |X|, \\ 1 \leq j \leq |Y| \end{array}$$

with initial conditions: (2)

$$A_{i,0} = \sum_{k=1}^{i} \sigma(X_k, -), \text{ and } A_{0,j} = \sum_{k=1}^{j} \sigma(-, Y_k).$$

Once the alignment matrix is filled, $A_{|X|+1,|Y|+1}$ is, by definition, equal to the minimum alignment cost of X and Y. Furthermore, the optimal alignments $X_{\text{opt}}^\diamond$ and $Y_{\text{opt}}^\diamond$ can be found by backtracking on the alignment matrix A. To this end, pointers can be assigned to each matrix cell as their values are being computed. The direction of the pointer at each cell (i, j) indicates which neighboring cell increased the alignment cost of $A_{i,j}$ by the least amount, and determines the type of the alignment (occurrence of a match or insertion/deletion of a gap) at positions i and j of X and Y, respectively.

The weights $\sigma(x, y)$ used for comparing DNA sequences are derived from computing the log-likelihood of co-occurrence of different pairs of nucleotides (x, y) in a training set of aligned sequences [3]. However, in many cases, it is common to assume more simplified functions such as,

$$\sigma(x, y) = \begin{cases} \sigma_{\text{mis}} & \text{, if } x \neq y, x, y \in \Sigma \\ \sigma_{\text{gap}} & \text{, if } x \text{ or } y \in \{-\} \\ 0 & \text{otherwise} \end{cases}, \qquad (3)$$

for some positive constants σ_{mis} and σ_{gap}.

The above dynamic programming framework can be directly generalized to align $N > 2$ annotation sequences $\{X^{(i)}\}_{i=1}^N$. Under such scenario, the alignment matrix $A \in \mathbf{R}^{(|X^{(1)}|+1)\times(|X^{(2)}|+1)\times\cdots\times(|X^{(N)}|+1)}$, and finding the optimal alignments requires computation of the cost for all of the $\prod_{i=1}^N |X^{(i)}|$ different cells of the alignment matrix A. Moreover, calculating the cost of each cell requires determination of the minimum over $2^N - 1$ different combinations of gaps. Assuming all annotation sequences are of roughly the same length $\bar{L}$, the time complexity of the multidimensional dynamic programming alignment is $O(2^N \bar{L}^N)$ [3].

The Needleman-Wunsch algorithm is closely related to the Levenshtein distance [4], used in computer science for calculating the minimum "edit distance" between two strings, and to the Dynamic Time Warping (DTW) algorithm [5], primarily used in speech processing for measuring similarity of temporal signals.

III. Method

A. Case I: Two annotation sequences

While a DNA sequence consists of a set of characters from the finite set Σ, an ECG annotation sequence is comprised of a set of indices corresponding to the location of a fiducial point (e.g. the location of QRS complexes) in the ECG signal over a certain period of time. Hence, the time indices are from the infinite set of real numbers. However, due to the pseudo-periodic nature of the ECG signals, a positive time difference exists between consecutive fiducial points. This property allows defining appropriate weight functions for the alignment of ECG annotation sequences. One such weight function could be in the form of

$$\sigma(t_1, t_2) = \frac{|t_1 - t_2|}{tol/2}, \text{ and } \sigma_{\text{gap}} = 1, \qquad (4)$$

where tol is the maximum acceptable amount of mismatch (in seconds) between two aligned indices. In other words, any two time points which are matched in the aligned annotation sequences contribute to the total cost by a weight proportional to their time difference, and any time point reported by only one of the annotators and missed by the other contributes by a weight of $+1$ to the alignment cost. The maximum mismatch parameter, tol, depends on the type of the fiducial point and the properties of the underlying ECG signal. For example, in the case of annotating fetal QRS (fQRS) locations, a time difference of 20 ms between the correct location and the annotator's reported location might be acceptable, and time differences above 100 ms might not be acceptable. Hence, $tol = 100$ ms would be an appropriate choice for this example.

Once a suitable weight function $\sigma(\cdot)$ is defined, the Needleman-Wunsch algorithm can be used for the alignment of annotation sequences similar to aligning DNA sequences.

In order to show the importance of sequence alignment prior to using any heuristic for comparing the annotation sequences, consider the scoring function used in the PhysioNet/CinC challenge 2013 [6] for annotating the location (time indices) of fQRS complexes in one-minute fetal ECG (fECG) recordings obtained from a set of electrodes placed on the mother's abdomen. The reference fQRS sequences for a small set of signals were available to the competitors as part of a training set.

Let X and R refer to a fQRS annotator and its corresponding reference sequence for a data set in the challenge training set, respectively. Furthermore, let X_{aligned} and R_{aligned} denote the aligned sequences using the Needleman-Wunsch algorithm with a weight function similar to (4). Then the following scoring function would provide a consistent and meaningful measure for comparing the two sequences:

$$S_{\text{consensus}} = f_s \cdot \left(\sqrt{\frac{1}{n_{\text{match}}} \sum_{\substack{\text{matched} \\ \text{indices } i}} (R_{\text{aligned}}[i] - X_{\text{aligned}}[i])^2} + \frac{n_{\text{gap}}}{|R|} \cdot k \cdot tol \right), \text{ for some } k > 1, \quad (5)$$

where n_{match} and n_{gap} refer to the total number of matches (two time indices matched together) and gaps (a time index matched with a gap) in the aligned sequences, and f_s denotes the sampling frequency of the underlying ECG signal. The first term in the above scoring function calculates the RMSE between the matched indices of X and R, and the second term assigns a fixed amount of penalty $k \cdot tol$ to each index that was matched with a gap. The coefficient $k > 1$, so that the penalty assigned for gaps is greater than the maximum allowable amount of mismatch between the matched indices. Note that by aligning the two sequences using the weight function (3), a matched pair of indices can have a time difference up to the mismatch tolerance tol, and indices in either sequence which do not find a corresponding index (in the other sequence) within their tol-margin are matched with a gap. Therefore, a gap either represents an index in the reference sequence that was not detected by annotator X, or an index that was erroneously detected in X and has no corresponding index in R.

The scoring function $S_{\text{consensus}}$ defined in equation (5) is used in Section IV to compare and score the similarity between fQRS annotation sequences and their corresponding reference responses. The scoring function $S_{\text{consensus}}$ is further compared with the scoring function $S_{\text{challenge}}$ which was used in the PhysioNet/CinC challenge 2013 to score the accuracy of the annotation sequences by computing the mean square error without prior sequence alignment [6].

B. Case II: Multiple annotation sequences

In the absence of a gold standard, comparison of multiple annotation sequences (from different annotators) can provide information about the underlying true labels or locations of fiducial points [1]. In order to avoid the high time complexity of the multiple sequence alignment technique outlined in Section III, we propose instead, to perform pairwise sequence alignment between any two annotation sequences in the set, and use a *union-find* data structure (also known as the *disjoint-set* data structure) to merge the results of pairwise matches together.

The union-find data structure keeps track of any partitioning of a set into a disjoint set of groups with fast implementation of the following operations,

- `find`(x): returns the name of the group element x is in; thus, x and y are in the same group if and only if `find`(x) = `find`(y),
- `union`(G_1, G_2): merges two groups G_1 and G_2, and
- `makeUnionFind`(S): returns a data structure where each element of set S is in a separate group.

Fast implementation[1] of the union-find data structure for a set S of size n has a time complexity of $O(n), O(1)$, and $O(\log n)$ for the `makeUnionFind`, `union`, and `find` operations, respectively [7]. Implementation details of the union-find data structure are out of the scope of this work and are not covered here.

In order to use the union-find data structure for the alignment of N annotation sequences, set S is initialized to be the union of all annotation indices reported by the N different annotators such that

$$S = \left\{ (i : X_j^{(i)}), \forall i \in \{1, \cdots, N\} \text{ and } j \in \{1, \cdots, |X^{(i)}|\} \right\}.$$

Next, pairwise sequence alignment is performed, using the Needleman-Wunsch algorithm, for each pair of annotation sequences and the matched indices of the aligned sequences are grouped together using the union-find operations previously described. This process is detailed in Algorithm 1.

Using Algorithm 1, a total of $O(N^2)$ pairwise sequence alignments must be performed; assuming all sequences have roughly same length $\bar{L}$, this corresponds to a time complexity of $O(N^2\bar{L}^2)$. Moreover, each union-find operation takes $O(\log(N\bar{L}))$, which amounts to a total time complexity of $O(N^2\bar{L}\log(N\bar{L}))$. Finally, the initialization of the data structure takes $O(N\bar{L})$. Therefore, the total time complexity of Algorithm 1 is $O(N\bar{L} + N^2\bar{L}\log(N\bar{L}) + N^2\bar{L}^2)$, which is mostly dominated by the cost of the pairwise alignments. Comparing the exponential time complexity of the multi-dimensional dynamic programming alignment (mentioned in Section II) versus the polynomial time complexity of the alignment via union-find data structures, the latter is a much more efficient algorithm for the alignment of multiple annotation sequences.

[1]The reported time complexities correspond to the *forest-based* implementation of the union-find data structure. Faster implementation via the path compression technique is not mentioned here.

Algorithm 1 Multiple sequence alignment using the union-find data structure

Input: a set of N annotation sequences $\{X^{(i)}\}_{i=1}^{N}$, and maximum acceptable mismatch tol for pairwise alignments
Output: All the matched elements between the annotation sequences in the form of a union-find data structure.
Comments: The `seqAlign` function takes two annotation sequences, together with a maximum acceptable mismatch parameter, and aligns the sequences using the Needleman-Wunsch algorithm.

```
1: $S \leftarrow \texttt{makeUnionFind}\Big(\Big\{(i : X_j^{(i)}), \forall i \in \{1, \cdots, N\}$ and $j \in \{1, \cdots, |X^{(i)}|\}\Big\}\Big)$
2: for $i$ in $\{1, \cdots, N-1\}$ do
3:   for $j$ in $\{i+1, \cdots, N\}$ do
4:     $X_{\text{aligned}}^{(i)}, X_{\text{aligned}}^{(j)} \leftarrow \texttt{seqAlign}(X^{(i)}, X^{(j)}, tol)$
5:     $match\_index \leftarrow \Big\{k \in \{1, \cdots, |X_{\text{aligned}}^{(i)}|\} \mid X_{\text{aligned},k}^{(i)} \neq \{-\}$ and $X_{\text{aligned},k}^{(j)} \neq \{-\}\Big\}$
6:     for $k$ in $match\_index$ do
7:       $G_1 \leftarrow \texttt{find}\big((i : X_{\text{aligned},k}^{(i)})\big)$
8:       $G_2 \leftarrow \texttt{find}\big((j : X_{\text{aligned},k}^{(j)})\big)$
9:       if $G_1 \neq G_2$ then
10:        $\texttt{union}(G_1, G_2)$
11:      end if
12:    end for
13:  end for
14: end for
15: return $S$
```

IV. Results and Discussion

Once the sequences are aligned, different heuristics can be devised to compare the aligned sequences with each other. The scoring function $S_{\text{consensus}}$ defined in equation (5), for example, uses both the RMSE measure and the number of gaps between the aligned annotation sequences to score the similarity of an annotator X with its reference sequence R.

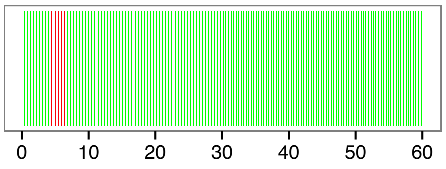
Fig. 1 and Fig. 2 compare the overall performance of $S_{\text{consensus}}$ (with parameters $tol = 100$ ms and $k = 2$) and the scoring function used in the PhysioNet/CinC challenge 2013 [6], denoted by $S_{\text{challenge}}$, to measure the similarity between a fQRS annotator and its reference response. As described in Silva et al. [6], $S_{\text{challenge}}$ was calculated by obtaining uniformly sampled fetal heart rate (FHR) estimates of the annotation and reference sequences using integral pulse frequency modulation (IPFM), and reporting the mean square error of the two estimates as the similarity score between the two annotation sequences.[2]

Both $S_{\text{consensus}}$ and $S_{\text{challenge}}$ aim to measure the closeness of an annotator sequence to a reference sequence and hence lower scores represent higher amount of similarity. Note that the two scoring functions have different ranges, and the scores assigned by them are not comparable with each other. The goal of this comparison, however, is to test the sensitivity and consistency of each scoring function with itself under different scenarios, and to show the importance of sequence alignment prior to measuring the similarity between annotation sequences.

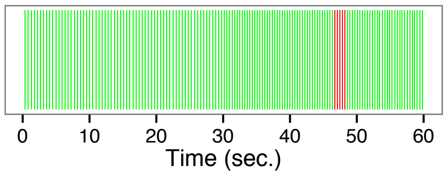
[2]This refers to the FHR time-series estimation method used in Events 1 and 4 of the Physionet/CinC challenge 2013. For more information, refer to Silva et al [6].

Fig. 1 depicts a very simple scenario where the annotation sequence X is exactly similar to the reference sequence R, except at five consecutive indices where annotator X was not able to detect the location of fQRS peaks, and hence those indices are not present in X. The location of gaps are different in Fig 1(a) and 1(b), but the total number of exact matches is the same in both cases. However, while $S_{\text{consensus}}$ assigns a similar score to both cases, $S_{\text{challenge}}$ assigns two very different scores; this sensitivity to location of gaps is not desirable for scoring functions which are devised to measure the overall similarity between two sequences.

Fig. 2 compares the consistency of the two scoring functions in measuring the similarity between an annotator sequence with its corresponding reference sequence. As indicated in the figure, while there are a total of 10 gaps in the aligned sequences of Fig. 2(a), $S_{\text{challenge}}$ has assigned a worse score for this annotator compared to Fig. 2(b) and 2(c), which have a total of 33 and 59 gaps in their aligned sequences, respectively. Also note that while there is a huge discrepancy between the annotator of Fig. 2(c) and its reference sequence, $S_{\text{challenge}}$ has assigned a better score for this annotator than those of Fig. 1. However, $S_{\text{consensus}}$ correlates with the total number of gaps associated with missed and erroneously detected peaks in the fECG signals.



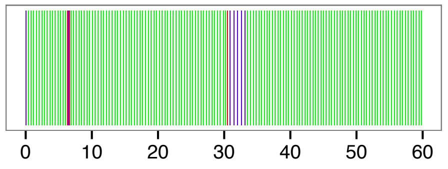
(a) n_{gap}=5, $S_{\text{challenge}} = 41.35, S_{\text{consensus}} = 6.90$



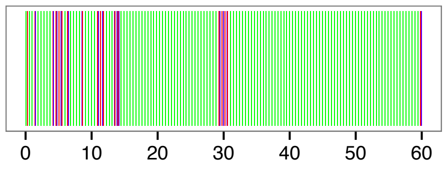
(b) n_{gap}=5, $S_{\text{challenge}} = 118.32, S_{\text{consensus}} = 6.90$

Fig. 1. Comparing the *insensitivity* of the two scoring functions $S_{\text{challenge}}$ and $S_{\text{consensus}}$ to the location of gaps between an annotator sequence X and its reference sequence R. The reference sequence is the same in both cases (a) and (b), but the annotator sequence $X = R \setminus \{R_{10}, \cdots, R_{14}\}$ in (a), and $X = R \setminus \{R_{110}, \cdots, R_{114}\}$ in (b). In both figures, the location of matches between R and X (along the time axis) are highlighted by green lines, and the location of gaps (where the annotator was not able to detect the peaks) are highlighted by red lines. Although the number of matches and gaps between R and X are the same in both cases, $S_{\text{challenge}}$ assigns two very different scores to them. However, $S_{\text{consensus}}$ is the same for both cases.

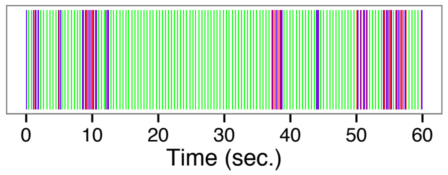
To show the performance of the union-find data structure in aligning multiple sequences in the absence of a gold standard, consider the following example in which the set of integers between 1 and 9 was used to generate five different sequences. In each sequence, some numbers were removed from the sequence to resemble the missing indices of annotation sequences. Moreover, a random noise between $[-0.2, 0.2)$ was added to each remaining number to simulate



(a) n_{gap}=10, $S_{\text{challenge}} = 118.60$, $S_{\text{consensus}} = 24.91$



(b) $n_{\text{gap}} = 33$, $S_{\text{challenge}} = 6.26$, $S_{\text{consensus}} = 72.31$



(c) n_{gap}=59, $S_{\text{challenge}} = 44.34$, $S_{\text{consensus}} = 122.69$

Fig. 2. Comparing the *consistency* of the two scoring functions $S_{\text{challenge}}$ and $S_{\text{consensus}}$ in scoring the similarity between an annotator sequence X and its reference sequence R. In all figures, the location of matches between R and X (along the time axis) are highlighted by green lines, the location of gaps where the annotator was not able to detect the peaks are highlighted by red lines, and the locations of gaps where the annotator has erroneously detected a peak is highlighted in blue. While it can be visually confirmed that the similarity between X and R decreases from (a) to (c), this is not reflected in the scorings of $S_{\text{challenge}}$.

$X^{(1)}$: 0.993,3.928,4.845,5.994,6.954

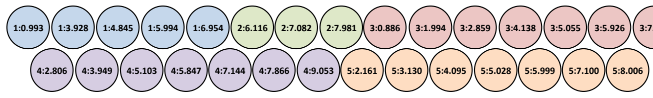
$X^{(2)}$: 6.116,7.082,7.981

$X^{(3)}$: 0.886,1.994,2.859,4.138,5.055,5.926,7.139

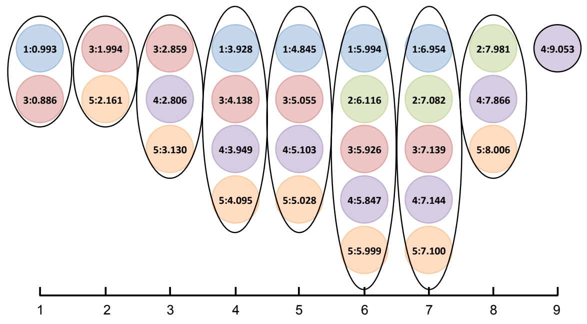
$X^{(4)}$: 2.806,3.949,5.103,5.847,7.144,7.866

$X^{(5)}$: 2.161,3.130,4.095,5.028,5.999,7.100,8.006,9.053

(a) Set of sequences



(b) Initialization



(c) Final result

Fig. 3. An example of using a union-find data structure to merge the information of multiple annotation sequences and cluster the corresponding indices together. (a) A set of five sequences with noisy readings from 0 to 9. (b) Initialization of the union-find data structure. At first, each annotation index is in a separate group. (c) Pairwise alignments establish index correspondences, and the corresponding indices are clustered in the same group.

the existence of noise in the annotations. Fig. 3(a) shows the resulting set of sequences. Next, the sequences were aligned via pairwise sequence alignment with mismatch tolerance $tol = 0.2$, and the matched results were grouped together using a union-find data structure. Fig. 3(b) and Fig. 3(c) show the data structure before and after the alignments, respectively. After the alignments, the size of each group in the data structure represents the amount of consensus between the annotators about existence of a true (hidden) annotation point $x^* \in (x_{\text{min}} - tol, x_{\text{max}} + tol)$, where x_{min} and x_{max} refer to the minimum and maximum values reported by annotators inside that group.

Once the sequences are aligned, different "voting" heuristics (such as mean, median or Bayesian voting) can be applied to merge the results of the aligned sequences and/or assess the "reliability" of each annotation algorithm.

V. Conclusions

In this work, we discussed the importance of sequence alignment prior to performing any comparison between a set of annotation sequences. This can show up in the form of scoring the "closeness" of an annotation sequence to a gold standard, or in the form of performing consensus or inference on a set of annotation sequences in the absence of a gold standard. We particularly discussed how the Needleman-Wunsch sequence alignment technique can be applied to different annotation sequences by devising proper weight functions. Finally, we compared the scoring function used in the PhysioNet/CinC challenge 2013 with a scoring function devised based on the sequence alignment results and discussed why the latter provides more consistent and meaningful scores for comparing ECG annotation sequences with their corresponding gold standards. A more detailed analysis and discussion of possible applications of the devised approach for measuring consensus among multiple ECG annotation sequences are left for future work.

References

[1] V. C. Raykar, S. Yu, L. H. Zhao, G. H. Valadez, C. Florin, L. Bogoni, and L. Moy, "Learning from crowds," *The Journal of Machine Learning Research*, vol. 11, pp. 1297–1322, 2010.

[2] T. Zhu, A. E. Johnson, J. Behar, and G. D. Clifford, "Bayesian voting of multiple annotators for improved QT interval estimation," in *Computing in Cardiology Conference (CinC)*, pp. 659–662, IEEE, 2013.

[3] R. Durbin, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*. Cambridge university press, 1998.

[4] M. J. Atallah, *Algorithms and theory of computation handbook*. CRC press, 1998.

[5] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[6] I. Silva, J. Behar, R. Sameni, T. Zhu, J. Oster, G. D. Clifford, and G. B. Moody, "Noninvasive fetal ECG: the PhysioNet/Computing in Cardiology challenge 2013," in *Computing in Cardiology Conference (CinC)*, pp. 149–152, IEEE, 2013.

[7] *http://www.cs.illinois.edu/class/fa07/cs473ug/Lectures/lecture4.pdf (last checked on 06/2014).*