Improving the Computational Complexity of the COOL Screening Tool

Mohamed Ghalwash
IBM Research, NY, USA
Ain Shams University, Cairo, Egypt

***Abstract*—Autoimmune disorder, such as celiac disease and type 1 diabetes, is a condition in which the immune system attacks body tissues by mistake. This might be triggered by abnormality in the development of biomarkers such as autoantibodies, which are generated by unhealthy beta cells. Therefore, screening of such biomarkers is crucial for early diagnosis of autoimmune diseases. However, one of the fundamental questions of screening is when to screen subjects who might be at a higher risk of autoimmune disorder. This requires an exhaustive search to find the optimal ages of screening in retrospective cohorts. Very recently, a comprehensive tool was developed for screening in autoimmune disease. In this paper, we improved the computational time of the algorithm used in the screening tool. The new algorithm is more than 100 times faster than the original one. This improvement would help to increase the utility of the tool among clinicians and research scientists in the community.**

***Keywords*—Software engineering; screening tool; autoimmune disorder**

I. Introduction

Autoimmune disorder is a condition in which the immune system mistakenly attacks healthy body tissues in different organs of the body. For example, in type 1 diabetes, the immune system destroys the insulin-producing cells of the endocrine pancreas, which leads to insulin deficiency [1]. In celiac disease, eating gluten – a protein found in wheat, rye, and barley – causes the autoimmune system to damage the small intestine [2]. There are many factors involved in causing such diseases such as genes, environmental factors, drugs and/or chemicals. However, autoimmune disorder is often associated with a few circulating autoantibodies, which are abnormal antibodies generated by pathogenic β-cells, when targeting a tissue [3]. Autoantibodies are often precede the onset of the disease and, therefore, considered as a clinical biomarker of the autoimmune disorder. In type 1 diabetes and celica diseases, there are four or five autoantibodies that are often used to assess the risk of developing the disease [4], [5].

Screening for autoantibodies – a group of serum tests to assess the presence of autoantibodies – is usually performed to detect the disease as early as possible so that a proper treatment or intervention can be administered. Therefore, frequent screening is of upmost importance to detect potential autoimmune disorder in subjects who in an apparently healthy population [6], [7]. Although frequent screening is beneficial for detecting subjects who are at a higher risk of the disease, it is cost inefficient and may also introduce harm for those who do not have the diseases by increasing the risk of overdiagnosis [8]. Therefore, one needs to find the *optimal* ages for screening in order to balance between the benefit and the harm of multiple screening.

To find a proper screening schedule, one needs to do cross-sectional experiments on retrospective cohort to find the optimal ages for screening. The authors of a recent paper [9] proposed a tool, called the Collaborative Open Outcomes tooL (COOL), that can be used to compute the quality performance of a given proposed screening schedule according to some measures that can be used to balance between the benefit and the harm of the screening schedule. However, computing these measures for a given schedule is a very time consuming task. In this paper, we propose to make these computations much faster. This proposed enhancement will increase the utility of the tool to compare multiple schedules to find the optimal (according to the given measures) screening schedule much faster.

II. Method

A. Data Structure

We explain the structure of the data used for defining the screening schedule. The data has biomarkers information for each subject. Each subject may visit the clinic multiple times and each time a blood sample is taken from the subject to assess the development of biomarkers. The value of each biomarker is either positive (the autoantibody is developed) or negative. It is worth mentioning that each subjects may have a different number of visits.

Notations: We use the upper case letter to define a matrix – a two dimensional array –, e.g. X, a boldface letter to define a vector, e.g. $\boldsymbol{x}$, and a italic letter to define an entry or element, e.g. x. $\boldsymbol{x}[i]$ represents the entry i of the vector $\boldsymbol{x}$. $X[i]$ represents the i^{th} row of the matrix X, and $X[i][j]$ represents the entry in the row i and column j.

Mathematically, let us define the data for a subject i as $\boldsymbol{x}_i = [(t_i^1, \boldsymbol{x}_i^1), (t_i^2, \boldsymbol{x}_i^2), \ldots, (t_i^{T_i}, \boldsymbol{x}_i^{T_i})]$ where T_i is the number of visits for the subject i, t_i^j is the subject's age at the visit j, and $\boldsymbol{x}_i^j \in \{0,1\}^M$ is the list of M biomarkers for the visit j. In addition, the information about whether and when the disease was developed is recorded. For simplicity, we assume that each subject either developed the disease within a predefined period of time from birth or the subject has been observed for the full period but has not developed the disease[1]. If the subject has

[1]The other case where the subject is partially observed and has not developed the disease (right censored subjects) can be handled using inverse probability censoring weights [10] but it is outside the scope of this paper.

developed the disease, the subject is not followed afterwards. y_i is the age when the disease was developed and -1 otherwise.

Example II.1. *Let us assume that there are four subjects,* 1*,* 2*,* 3*, and* 4*. The data for these four subjects can be represented as*

- $\boldsymbol{x}_1 = [(1,[0,1,0,1]),(2.3,[1,1,0,0]),(5.8,[0,1,0,1]),(7.1,[1,0,1,0])]$, $y_1 = 9$
- $\boldsymbol{x}_2 = [(2.4,[0,0,0,1]),(6,[1,0,0,1]),(9.2,[0,0,0,1]))]$, $y_2 = -1$
- $\boldsymbol{x}_3 = [(1.9,[0,0,0,0]),(7.4,[0,0,1,1]))]$, $y_3 = 8$
- $\boldsymbol{x}_4 = [(0.6,[0,1,0,0]),(4.7,[0,0,0,1]),(6.4,[0,0,1,1]),(10,[0,0,0,0]))]$, $y_4 = -1$

Subject 1 *has a sequence of* $T_1 = 4$ *visits. Each visit has measurements for* $M = 4$ *biomarkers. The first visit was measured at age* $t_1^1 = 1$ *year and the second and the fourth biomarkers were positives while the other two biomarkers were negatives. The second visit was sampled at age* $t_1^2 = 2.3$ *years. We can see that the fourth biomarker turned to negative in the second visit while the first biomarker became positive. The subject has developed the disease at age* $y_1 = 9$ *years. The second subject has* $T_2 = 3$ *visits at ages* $2.4, 6$*, and* 9.2 *years and has not developed the disease, i.e.* $y_2 = -1$*. We clearly see that each subject may have a different number of visits and these visits might be sampled at different ages. A graphical representation of these data is shown in Fig. 1.*

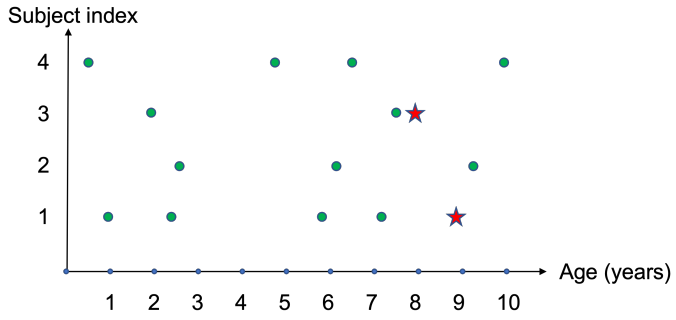

Fig. 1. A Graphical Representation for the given Data in the Example. The Green Points Represent Visits while the Red Stars Represent the Age at which the Subjects Developed the Disease. E.g., Subject 3 has 2 Visits and Developed the Disease at Age 8.

One simple data structure that can be used to store the data for all subjects is a 3-dimensional array, where the first dimension is the number of subjects N, the second dimension is the maximum number of visits $S = \max_i\{T_i\}$, and the third dimension is the number of biomarkers M, i.e. $\mathbb{R}^{N\times S\times M}$. However, there are two challenges to store the data in a three-dimensional array. The first challenge is that each subject may have a different number of visits. The second challenge is the irregularity in the biomarkers collections. As seen from the example, the biomarkers are collected at different and irregular time stamps. These two issues pose a challenge to store the data for all subjects in a 3-dimensional array, which assumes that the data are time-aligned. A better data structure for storing such information would be a 2-dimensional array (matrix) with a special structure.

Let us assume that $T = \sum_{i=1}^{N} T_i$ is the total number of visits across all N subjects. We construct a matrix X with dimensions $T \times M+2$, where each row represents one visit for a particular subject. The first column in the matrix represents the subject index, the second column is the age of the subject at the current visit, the other M columns are the values of the biomarkers. Data is sorted in ascending order by subject index and age. An additional array $\boldsymbol{y}$ stores the age at which the subject developed the disease, i.e. $\boldsymbol{y}[i]$ is the age when the subject i developed the disease and -1 otherwise.

Example II.2. *The matrix for the data in Example II.1 can be represented as*

$$X = \begin{bmatrix} 1 & 1.0 & 0 & 1 & 0 & 1 \\ 1 & 2.3 & 1 & 1 & 0 & 0 \\ 1 & 5.8 & 0 & 1 & 0 & 1 \\ 1 & 7.1 & 1 & 0 & 1 & 0 \\ 2 & 2.4 & 0 & 0 & 0 & 1 \\ 2 & 6.0 & 1 & 0 & 0 & 1 \\ 2 & 9.2 & 0 & 0 & 0 & 1 \\ 3 & 1.9 & 0 & 0 & 0 & 0 \\ 3 & 7.4 & 0 & 0 & 1 & 1 \\ 4 & 0.6 & 0 & 1 & 0 & 0 \\ 4 & 4.7 & 0 & 0 & 0 & 1 \\ 4 & 6.4 & 0 & 0 & 1 & 1 \\ 4 & 10.0 & 0 & 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{T\times M+2},$$

$$\boldsymbol{y} = \begin{bmatrix} 9 \\ -1 \\ 8 \\ -1 \end{bmatrix}$$

As it can be seen, subject 1 *has 4 rows in the matrix representing 4 visits, and subject* 3 *has two rows.*

B. Single-Age Screening

Problem 1 (Single-age screening)**.** *At which age* a*, subjects with a positive test at that age will likely develop the disease within the observation period?*

The objective of screening at a single age is to assess the likelihood that a subject has the disease. Let us assume that the screening test is whether any biomarker is positive. The question would be how likely subjects with any positive biomarker at a given age will develop the disease within the observation period (e.g. within 10 years from birth). In order to compute the quality performance of the screening at a single age, we need to compute the following Table I:

TABLE I. SUMMARY OF THE SCREENING TEST RESULTS

Screening test	Developed the disease	Not developed the disease
Positive	# true positives (TP)	# false positives (FP)
Negative	# false negatives (FN)	# true negatives (TN)
No test	# no test and positives (NP)	# no test and negatives (NN)

Each subject will be placed in one of these six cells. If the subject was tested positive and developed the disease, the subject will be counted in the TP cell. FP is the number of subjects who were tested positive and have not developed the disease. Similarly, FN (TN) is the number of subjects

who were tested negative and developed (not developed) the disease, respectively. Finally, since not all subjects may not necessarily have a visit at a particular age, some subjects may have no screening test and therefore will be missing from the screening test. This is accounted for in the last row of Table I.

Using the information provided in Table I, the screening test is usually evaluated using the sensitivity and the specificity measures [11]. The sensitivity is the probability that the screening test is positive among those who have the disease. Specificity is the probability that the screening test is negative among those who do not have the disease [12]. These two measures can be computed as:

$$Sen = \frac{TP}{TP + FN} \tag{1}$$

$$Spc = \frac{TN}{TN + FP} \tag{2}$$

Example II.3. *If the sensitivity is* 80%*, it means that* 80% *of diseased subjects are identified as diseased (have a positive test). If the specificity is* 90%*, it means that* 90% *of non-diseased subjects have a negative test (correctly identified as non-diseased).*

These two measures are important as they measure the percentage of diseased individuals who have positive test results and the percentage of non-diseased individuals who have negative test results, respectively. Nevertheless, these two measures assume that the test result for each subject is known, i.e. they do not account for subjects with missing tests. Cumulative sensitivity ($CSen$) and dynamic specificity ($DSpc$) address this issue [13]:

$$CSen = \frac{TP}{TP + FN + NP} \tag{3}$$

$$DSpc = \frac{TN}{TN + FP + NN} \tag{4}$$

As it can be seen, $CSen$ and $DSpc$ require all subjects who do/do not have the disease, respectively. However, from the subject's perspective, these two measures do not give insights about the likelihood to develop the disease if the test results is positive or negative. Positive predictive value (PPV) and negative predictive value (NPV) answer this question.

$$PPV = \frac{TP}{TP + FP} \tag{5}$$

$$NPV = \frac{TN}{TN + FN} \tag{6}$$

PPV is the probability of having the disease among those subjects who tested positive. NPV is the probability of not having the disease among those subjects who tested negative.

So, in order to evaluate the performance of a screening test, we need to compute 4 measures $CSen$, $DSpc$, PPV and NPV. Algorithm 1 evaluates the performance of a screening at a given age a by computing these four measures.

Algorithm 1 takes as parameters the age a at which the screening will be evaluated, the data matrix X that encodes the age and the biomarkers information, and the label array $\boldsymbol{y}$ that encodes the age at which the disease was developed. The algorithm utilizes an array $found$ to mark whether the subject has a screening test at the given age a, i.e. $found[i] = 1$ if the subject i has a visit at the given age a, and 0 otherwise.

Algorithm 1: Single-Age Screening (SS)

Input: Age a, encoding data matrix X, label array $\boldsymbol{y}$
Return: $CSen$, $DSpc$, PPV, and NPV.

```
// list for all N subjects
Initialize all N entries of the found list with false
  Initialize TP, TN, FP, FN, NP, and NN with zeros.
for each row in X do
    // row is a list of M+2 entries
    id = row[1]
    age = row[2]
    biomarkers = row[3 : M + 2]
    /* if the age is within 6 months of a     */
    if a − .5 ≤ age < a + .5 then
        // found a visit for the current subject
        found[id] = true
        if IsPos(biomrakers) then
            PositiveTest(y[id])
        else
            NegativeTest(y[id])
  /* loop over subjects with a missing screening
     test to compute NN and NP.                */
for each subject id do
    // if the test is missing
    if found[id] is false then
        MissingTest(y[id])
Compute CSen, DSpc, PPV, NPV using equations 3-6
```

Algorithm 2: Helper Functions

```
Function PositiveTest(label):
    if label ≥ 0 then
        TP = TP + 1 // diseased subject
    else
        FP = FP + 1 // non-diseased subject
    return
Function NegativeTest(label):
    if label ≥ 0 then
        FN = FN + 1 // diseased subject
    else
        TN = TN + 1 // non-diseased subject
    return
Function MissingTest(label):
    if label ≥ 0 then
        NP = NP + 1 // diseased subject
    else
        NN = NN + 1 // non-diseased subject
    return
Function IsPos(biomarkers):
    return
```

In line 1, the algorithm initializes the boolean array $found$ with false. In line 2, it initializes all counts with zero. Then, it loops over all rows in the data matrix X (line 3), and for each row it checks whether the age of the current visit is within a specified window of 6 months around the given age (line 7). If yes, it marks that the subject has been tested (line 8) and checks the results for the screening test (line 9) using the function $IsPos$. If the test result is positive (line 10), the algorithm calls the function $PositiveTest$ in Algrithm 2, which updates the number of true positives or false positive depending on whether the patient has developed the disease. Otherwise, it updates the number of false positives (line 12). If the test result is negative (line 11), the algorithm calls the function $NegativeTest$ which updates either false negatives if the patient developed the disease or true negatives if the patient has not developed the disease.

Finally, after iterating over the entire matrix X, the algorithm iterates over the $found$ array (line 13) to find those who have not been tested at the given age (line 14) and calls the function $MissingTest$ in line 15 to compute the number of subjects who missed the screening test and developed (NP) or did not develop the disease (NN). After computing TP, TN, FP, FN, NP, and NN counts, the algorithm uses equations (3-6) to compute $CSen$, $DSpc$, PPV, and NPV for the single-age screening at age a.

Time Complexity: The for loop in line 3 has $O(T)$ iterations. Let us assume that the function $IsPos$ in line 9 takes $O(M)$. The loop in line 18 takes $O(N)$. Hence, the total time complexity of Algorithm 1 is $O(T.M + N)$.

Example II.4. *We compute the quality performance of screening for any biomarker (if any biomarker is positive, the result of the test is positive) at age 2 using data provided in Example II.2. The summary statistics of screening at age 2 is given in the following Table II:*

TABLE II. SUMMARY STATISTICS FOR SCREENING OF ANY BIOMARKER AT AGE 2. THE SUBJECT ID COLUMNS INDICATES THE SUBJECTS USED FOR COMPUTING THE MEASURE OR THE COUNT

Screening test	Count	Subject ids
TP	1	1
FP	1	2
TN	0	
FN	1	3
NP	0	
NN	1	4
$CSen$	0.5	1,3
$DSpc$	0	2,4
PPV	0.5	1,2
NPV	0	3

The screening at a single age might not perform good as some subjects might miss the screening test and that will reduce the sensitivity and/or specificity of the test. To increase the quality performance of a screening, one can screen twice so that those subjects who missed the first screening can be covered by the second screening. This is discussed in the next section.

C. Two-Age Screening

Problem 2 (Two-age screening)**.** *How likely subjects with a positive test at either one of a pair of ages a and b will develop the disease within the observation period?*

The screening test can be performed at the first age a. If the result is positive then no need to screen again and the final result is positive. If the screening test is negative or the subject missed the first screening then another screening is required at the second age and the result of the second screening determines the final result. If the subject missed both screening then it will be counted either in NN or NP depending whether the subject developed the disease. The two-age screening can be visualized as in Fig. 2.

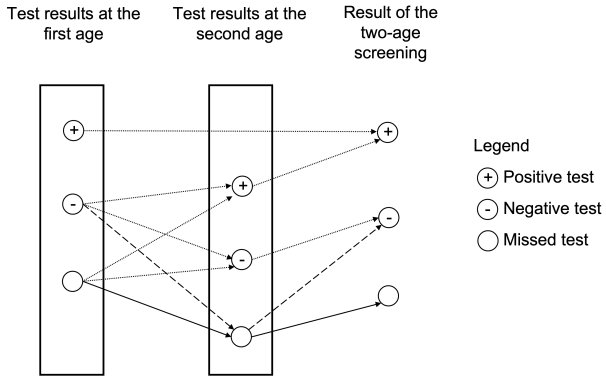

Fig. 2. Visualization of the Two-Age Screening Process. The Final Result of the Screening is Positive if and only if One of the Screenings at the First or the Second Age is Positive. The Final Result is Missing if both Screening are Missing. Otherwise, the Final Result is Negative.

Algorithm 3 describes the two-age screening process. The algorithm takes a pair of ages a and b to compute the screening results where $a < b$. For each row in X, it tests whether the current visit are withing the window of 6 months of age a (line 7). If the subject has a visit within that window, the algorithm applies the screening test (line 8) and if the result is positive, it marks that the subject id has a positive test at age a (line 9) and then updates TP and TN in line 10. If the result is negative, it marks that the subject has a negative first screening (line 12).

If the current visit is not within the window of 6 months around a, the algorithms checks for the second screening (note that the matrix X is sorted in ascending order by age). If the visit is within the window of 6 months around the second age b and if the subject has no positive results in the first screening (line 13), then it checks the results of the screening at the second age (line 14). If the screening at the second age is positive, the algorithm marks that the second screening is positive (line 15) and updates the counts TP and FP (line 16). If the second screening is negative, it marks that the second screening is negative (line 18).

After iterating over all rows in X, the algorithm iterates over all subjects who missed the first and the second tests (line

Algorithm 3: Two-Age Screening (TS)

Input: Ages a, b where $a < b$, encoding data matrix X, label array $\boldsymbol{y}$
Return: $CSen$, $DSpc$, PPV, and NPV.

```
// assume all subject missed both screenings
1  Initialize all N entries of found1 and found2 with −1
2  Initialize TP, TN, FP, FN, NP, and NN with 0
3  for each row in X do
4      id = row[1]
5      age = row[2]
6      biomarkers = row[3 : M + 2]
       /* if the age is within the window of a   */
7      if a − .5 ≤ age < a + .5 then
           // found a visit for the current subject
8          if IsPos(biomarkers) then
9              found1[id] = 1            // 1st test is positive
10             PositiveTest(y[id])
11         else
12             found1[id] = 0            // 1st test is negative
13     else if b − .5 ≤ age < b + .5 ∧ found1[id] ≠ 1 then
14         if IsPos(biomarkers) then
15             found2[id] = 1            // test is pos
16             PositiveTest(y[id])
17         else
18             found2[id] = 0            // test is neg
   /* loop over subjects with a missing test    */
19 for each subject id do
       // if both tests are missing
20     if found1[id] = −1 ∧ found2[id] = −1 then
21         MissingTest(y[id])
       // if 1st test is neg and 2nd is missing
22     else if found1[id] = 0 ∧ found2[id] = −1 then
23         NegativeTest(y[id])
24 Compute CSen, DSpc, PPV, NPV using equations (3-6)
```

20) to update the counts NN and NP (line 21) and iterates over subjects who tested negative in the first screening and missed the second screening (line 22) to update the FN and TN counts (line 23). Finally, the screening quality measures are computed in line 24.

Time Complexity: The time complexity of Algorithm 3 is $O(T.M + N)$.

Although the time complexity of Algorithm 3 is $O(T.M + N) \approx O(T)$, but the actual running time is very large, especially if the algorithm needs to be executed multiple times. For example, in almost all cases in medical context, a confidence interval for each measure (sensitivity, specificity, PPV and NPV) is required. To compute the confidence interval [14], the algorithm needs to be run thousands of times on different samples of the matrix X. In addition, to compare different screening schedules, we compute the confidence interval for each schedule and compare them to see how statistically significant the difference between the screening schedules is [15]. Therefore, it is preferred that the algorithm that computes the quality performance of the screening needs to be fast enough so that all these experiments can be run in a reasonable time.

To do that, we perform a data pre-processing that needs to be done only once, and then we will devise Algorithm 3 to make it faster which can be run multiple times and obtain the results much faster than using Algorithm 3.

D. Improved Two-Age Screening

We start with the improved algorithm for the two-age screening which can be easily modified for single-age screening. To improve the computational time of the two-age screening algorithm, we preprocess the data in a different data structure so that the computation becomes faster. The preprocessing step needs to be executed only once for the data and then each application of the two-age screening uses the preprocessed data and returns the results faster than the original algorithm.

For now, let us assume that we have already constructed a matrix B that contains the biomarker information, which will be used by the screening schedule algorithm (the construction of this matrix is explained in Section II-E). $B \in \mathbb{Z}^{N \times A}$ where A is the number of all possible distinct ages in the data that the screening are to be evaluated at, and N is the number of subjects. The entry $B[id][a] \in \{-1, 0, 1, 2, \ldots, 2^M\}$ has the *encoding* of the biomarkers for the subject id at age a. Since the biomarkers are binary, then the number of all possible cases of biomarkers values is 2^M (note that the number of biomarkers is usually small in these applications as explained in the introduction section). The value -1 indicates that the subject id missed the test at age a.

Example II.5. *Given Example II.2, there are* $2^4 + 1 = 17$ *possible values for each entry in the matrix* B*. The encoding matrix* B *is shown here:*

$$B = \begin{bmatrix} 10 & 3 & -1 & -1 & -1 & 10 & 5 & -1 & -1 & -1 \\ -1 & 8 & -1 & -1 & -1 & 9 & -1 & -1 & 8 & -1 \\ -1 & 0 & -1 & -1 & -1 & -1 & 12 & -1 & -1 & -1 \\ 2 & -1 & -1 & -1 & 8 & 12 & -1 & -1 & -1 & 0 \end{bmatrix}$$

Column j *encodes the biomarker information at age* j*. For example, the entry* $B[2][6]$ *encodes the biomarker information for subject* $id = 2$ *at age* 6*. The biomarker for subject* 2 *at age* 6 *were* $[1, 0, 0, 1]$ *which can be encoded as* $2^1 + 2^0 + 2^0 + 2^1 = 9$*. Similarly, the biomarker of subject* 3 *at age* 2 *is encoded as* $B[3][2] = 2^0 + 2^0 + 2^0 + 2^0 = 0$*. All entries with* -1 *indicate that the subject has no visit at that age, e.g.* $B[2][5] = -1$ *because the subject* 2 *has no visit at age* 5*.*

Note that all ages are rounded given the window of interest. For example, visits at ages 2.4 *are considered at age* 2 *(this is similar to line 6 in Algorithm 1)*[2]

Given the matrix B, the improved algorithm for two-age screening is re-written in Algorithm 4. The algorithm iterates over all subjects (line 1), and for each subject id it checks if screening at age a and age b are missing (line 2) then it marks

[2]If there are multiple visits within the window around the given age, we can consider either the closest visit, the first visit, the last visit, or any other visit based on the application. This is outside the scope of this paper.

that the final result is missing (line 3). If one of the tests is positive (line 4) it marks that the final result is positive (line 5). Otherwise, it marks that the final results is negative (line 7).

Algorithm 4: Improved Two-Age Screening (ITS)

Input: Ages a, b where $a < b$, biomarkers matrix B, label array $\boldsymbol{y}$
Return: $CSen$, $DSpc$, PPV, and NPV.

```
/* loop over all subjects                                   */
1 for each subject id do
      // if both tests are missing
2     if B[id][a] = −1 ∧ B[id][b] = −1 then
3         MissingTest(y[id])
      // one of tests is positive
4     else if IsPos(B[id][a]) ∨ IsPos(B[id][b]) then
5         PositiveTest(y[id])
      // (both tests are negative) or (one is
         negative and the other is missing)
6     else
7         NegativeTest(y[id])
8 Compute CSen, DSpc, PPV, NPV using equations (3-6)
```

Time Complexity: The running time for Algorithm 4 is $O(N)$.

E. Data Preprocessing for ITS

We preprocess the data only once to construct the biomarker encoding matrix B which makes the algorithm runs faster as evident by our experiments. The algorithm for constructing the matrix B is shown in Algorithm 5. The algorithm iterates over all rows of the matrix X (line 2). For each row, it maps the age to the closest age (line 6), encodes the biomarkers (line 7), and stores the value in the matrix B (line 8). To encode the biomarker information into one integer value (line 9), we multiple the biomarker vector into the encoding vector (line 10) to obtain the code value (line 11).

Algorithm 5: Biomarker Matrix

Input: matrix $\boldsymbol{X}$
Return: biomarkers matrix B

```
1  Initialize all entries of B with −1
   /* loop over all subjects                                */
2  for each row in X do
3      id = row[1]
4      age = row[2]
5      biomarkers = row[3 : M + 2]
6      a = Round(age) // map it to the closed age
7      code = encode(biomarkers)
8      B[id][a] = code
9  Function encode(biomarker):
10     e = [1  2  4  16  ...  2^M]^T // column vector
11     code = biomarker × e // matrix multiplication
12     return code
```

Time Complexity: The time complexity of Algorithm 5 is $O(T)$ but this process is executed only once not for each application of screening.

F. Improved Single-Age Screening

The improved algorithm for a single-age screening is shown in Algorithm 6.

Algorithm 6: Improved Single-Age Screening (ISS)

Input: Ages a, biomarkers matrix B, label array $\boldsymbol{y}$
Return: $CSen$, $DSpc$, PPV, and NPV.

```
/* loop over all subjects                                   */
1 for each subject id do
      // if the test is missing
2     if B[id][a] = −1 then
3         MissingTest(y[id])
      // the test is positive
4     else if IsPos(B[id][a]) then
5         PositiveTest(y[id])
      // (the test is negative)
6     else
7         NegativeTest(y[id])
8 Compute CSen, DSpc, PPV, NPV using equations (3-6)
```

III. Experiments

We evaluated the performance of the SS, TS, ISS and ITS algorithms on datasets with different number of subjects and visits. The description of the datasets is shown in Table III. The experiments were run on a Mac laptop with processor 2.7 GHz Quad-Core Intel Core i7 and 16 GB of memory. The screening test used for these experiments is to test for any positive biomarker, i.e. if any biomarker is positive the result of the screening test is positive. The code is written in Python [16]. Python has a data structure called pandas dataframe [17] which can be used store information in the matrix X. Using the dataframe, the SS and TS algorithm can be even run faster if we filter the dataframe on rows where the age is within the 6 months window of the given age a. This is done using the command

```
df[(df['age'] >= a-0.5) & (df['age']<a+0.5)]
```

In all our experiments for the SS and the TS algorithms we used the above command.

TABLE III. Each Subject has on Average 30 Visits. There are 3 Binary Biomarkers. Such as Distribution of Visits, Number of Subjects, etc

# subjects	# total visits	average # visits
9.170	169,530	19
13.383	219,276	16
15.747	240,917	15
18.984	262,233	14

A. Single Age Screening

We compared the running time for the single-age screening algorithms SS and ISS on different datasets. The experiments were run multiple times and the median and quartiles of the running times are reported as shown in Fig. 3.

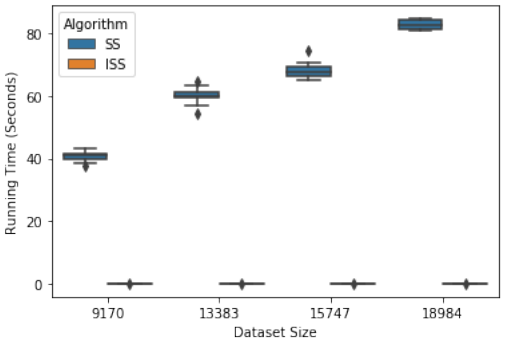

Fig. 3. Running Time Comparison between SS and ISS.

It is clear that the running time of the SS algorithm increases linearly with the dataset size. It takes about 80 seconds for Algorithm 1 to compute the quality performance of screening at a single age on data that has about 19,000 subjects, while the improved algorithm ISS takes only a fraction of a second to get the results. The running time for the ISS algorithm is shown in Fig. 4.

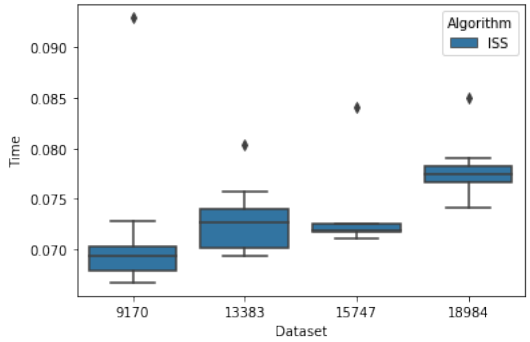

Fig. 4. Running Time for the ISS Algorithm.

B. Two Ages Screening

We compared the running time for the TS and ITS algorithms to compute the performance of the two-age screening. The results are shown in Fig. 5. A very similar behavior is observed. The TS algorithm scales linearly with the dataset size. The ITS algorithm is much faster than the TS algorithm. The running time for the ITS algorithm is shown in Fig. 6. ITS takes only 0.1 seconds to compute the quality performance of screening at a given two ages while TS takes 175 seconds.

C. Data Preprocessing for ISS and ITS

The additional overhead that the improved algorithms add on top of the original algorithms is the data preprocessing, i.e. the construction of the biomakers encoding matrix B. This step is required only once for each dataset. The running time for Algorithm 5 is shown in Table IV.

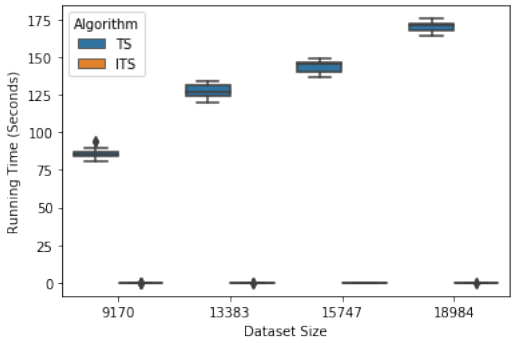

Fig. 5. Running Time Comparison between TS and ITS.

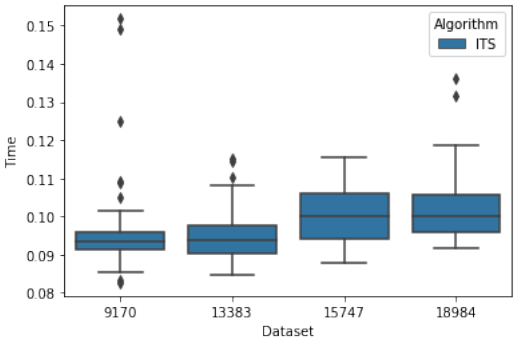

Fig. 6. Running Time for the ITS Algorithm.

TABLE IV. RUNNING TIME FOR CONSTRUCTING THE MATRIX B

Dataset size (# subjects)	Time (mins)
9170	5.3
13383	6.7
15747	7.5
18984	8.4

IV. CONCLUSION

Screening of biomarkers is of atmost importance to assess the risk of developing autoimmune diseases such as type 1 diabetes and celiac diseases. To improve the quality performance of the screening test, screening more than one time is required. Algorithms to compute the quality performance of a screening schedule were developed as part of a screening tool. However, the running time of these algorithms are large which hinders the utility of the tool on large applications. We improved the running time of the screening algorithms by more than 800 times at an additional cost of preprocessing the data only once. We evaluated the running time of these screening algorithms on datasets with different sizes.

REFERENCES

[1] S. A. Paschou, N. Papadopoulou-Marketou, G. P. Chrousos, and C. Kanaka-Gantenbein, "On type 1 diabetes mellitus pathogenesis," *Endocrine connections*, vol. 7, no. 1, pp. R38–R46, 2018.

[2] P. H. Green and C. Cellier, "Celiac disease," *New england journal of medicine*, vol. 357, no. 17, pp. 1731–1743, 2007.

[3] Z. X. Xiao, J. S. Miller, and S. G. Zheng, "An updated advance of autoantibodies in autoimmune diseases," *Autoimmunity Reviews*, vol. 20, no. 2, p. 102743, 2021.

[4] C. E. Taplin and J. M. Barker, "Autoantibodies in type 1 diabetes," *Autoimmunity*, vol. 41, no. 1, pp. 11–18, 2008.

[5] S. Caja, M. Mäki, K. Kaukinen, and K. Lindfors, "Antibodies in celiac disease: implications beyond diagnostics," *Cellular & molecular immunology*, vol. 8, no. 2, pp. 103–109, 2011.

[6] W. H. Organization *et al.*, "Screening programmes: a short guide. increase effectiveness, maximize benefits and minimize harm," 2020.

[7] L. Frommer and G. J. Kahaly, "Type 1 diabetes and associated autoimmune diseases," *World journal of diabetes*, vol. 11, no. 11, p. 527, 2020.

[8] N. Gilbert, "The pros and cons of screening," *Nature*, vol. 579, no. 7800, pp. S2–S2, 2020.

[9] M. Ghalwash, E. Koski, R. Veijola, J. Toppari, W. Hagopian, M. Rewers, and V. Anand, "Simulating screening for risk of childhood diabetes: The collaborative open outcomes tool (cool)," in *AMIA Annual Symposium Proceedings*, vol. 2021. American Medical Informatics Association, 2021, p. 516.

[10] D. M. Vock, J. Wolfson, S. Bandyopadhyay, G. Adomavicius, P. E. Johnson, G. Vazquez-Benitez, and P. J. O'Connor, "Adapting machine learning techniques to censored time-to-event health record data: A general-purpose approach using inverse probability of censoring weighting," *Journal of biomedical informatics*, vol. 61, pp. 119–131, 2016.

[11] S. Nissen-Meyer, "Evaluation of screening tests in medical diagnosis," *Biometrics*, pp. 730–755, 1964.

[12] R. Trevethan, "Sensitivity, specificity, and predictive values: foundations, pliabilities, and pitfalls in research and practice," *Frontiers in public health*, vol. 5, p. 307, 2017.

[13] A. N. Kamarudin, T. Cox, and R. Kolamunnage-Dona, "Time-dependent roc curve analysis in medical research: current methods and applications," *BMC medical research methodology*, vol. 17, no. 1, pp. 1–19, 2017.

[14] B. Efron, *The jackknife, the bootstrap and other resampling plans*. SIAM, 1982.

[15] P. Armitage, G. Berry, and J. N. S. Matthews, *Statistical methods in medical research*. John Wiley & Sons, 2008.

[16] G. Van Rossum and F. L. Drake Jr, *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[17] T. pandas development team, "pandas-dev/pandas: Pandas," Feb. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3509134