A Coprocessor Architecture for Fast Protein Structure Prediction

M. Marolia, R. Khoja, T. Acharya, C. Chakrabarti
Department of Electrical Engineering
Arizona State University,
Tempe, USA.

***Abstract*—Predicting protein structure from amino acid sequence is computationally very intensive. In order to speed up protein sequence matching and processing, it is necessary to develop special purpose VLSI architectures that exploit the underlying computational structures. In this paper, we present a coprocessor architecture for fast protein structure prediction based on the PSIPRED algorithm. The architecture consists of systolic arrays to speed up the data intensive sequence alignment and structure prediction steps, and finite state machines for the control dominated steps. The architecture has been synthesized using Synopsis DC Compiler using 0.18µm CMOS technology. The synthesized architecture requires 783,228 units of gate area and 226KB of memory, and can be clocked at 100 MHz. The architecture processes amino acid sequences extremely fast; for a database of 135,000,000 amino acids, the secondary structure of a query sequence of length ~150 amino acids can be predicted in ~11 seconds.**

I. INTRODUCTION

Protein structure analysis and prediction is a core area of research in bioinformatics. Determining the protein structure from amino acid sequence leads to better understanding of the functionality of the protein resulting in faster drug discovery. Protein structure prediction typically involves amino acid sequence matching with known structures. According to Moore's law of genomics, "The Protein Databank (PDB) of known structures will continue to double every 3 years and the number of sequences in want of a predicted structure will continue to double every few months" [1]. Thus in order to process the large volume of data, it is becoming increasingly important to develop faster and simpler techniques for protein structure prediction.

There are numerous methods [2-4] with varying degrees of accuracy for protein structure prediction. Most of these are based on homologous template matching and use various fold recognition techniques to predict the tertiary structural model. Exhaustive approaches such as the ab initio tertiary structure prediction [5] are also extremely slow. To overcome the computational bottleneck and to make protein structure prediction faster, secondary structure prediction methods are used. The secondary protein structure prediction method used here is the highly accurate PSIPRED algorithm [6] which is based on PSI-BLAST (Position Specific Iterated BLAST) [7]. It uses neural networks on the Position Specific Scoring Matrix (PSSM) generated by the PSI-BLAST algorithm to predict the secondary structure for the amino acid query sequence.

The computationally intensive nature of the PSIPRED makes it necessary to develop special purpose VLSI architectures that would serve as co-processors. In this paper, we have presented one such architecture for the PSIPRED algorithm that achieves several orders of speedup in execution time. The architecture is pipelined and consists of a systolic array to compute local alignment between the query and database, a neural network to predict the secondary structure, and finite state machines for the control dominated steps. The architecture is synthesized using Synopsis with 0.18µm CMOS technology; the gate area is 783,228 units, the memory is of size 226KB and the clocking speed is 100 MHz. The architecture can predict the secondary structure of an amino acid sequence of length ~150 in ~11 seconds using a database of 135,000,000 amino acids.

The rest of the paper is organized as follows. We first explain the protein structure prediction algorithm in section II. Then we present the details of the proposed architecture in section III. The synthesis results are highlighted in section IV and concluding remarks made in section V.

II. PROTEIN STRUCTURE PREDICTION TECHNIQUE

Protein is a biopolymer of amino acids arranged in a specific sequence that fold into a three-dimensional shape. The shape of a three dimensional protein molecule is determined by the sequential order of the amino acids. The folded structure can be constructed with three types of substructure (secondary structure): helix structure (α), sheet structure (β) or other coil structures (γ).

Of the various secondary structure prediction methods that exist today [2-4], PSIPRED is a method that demonstrated average accuracy of 77.3% at CASP3 (Third Critical Assessment of Structure Prediction) meeting [6]. We selected this method because of its simplicity and the capability of giving the highest level of accuracy published to date.

The PSIPRED algorithm takes a sequence of amino acids as a query sequence input and predicts the corresponding secondary structure. It is implemented in two steps: (1) PSI-BLAST, for multiple sequence alignment

followed by updating a sequence profile iteratively to generate position specific scoring matrix (PSSM) and (2) secondary structure prediction and filtering using neural networks. The key components of the algorithm and their interactions are shown in fig. 1.

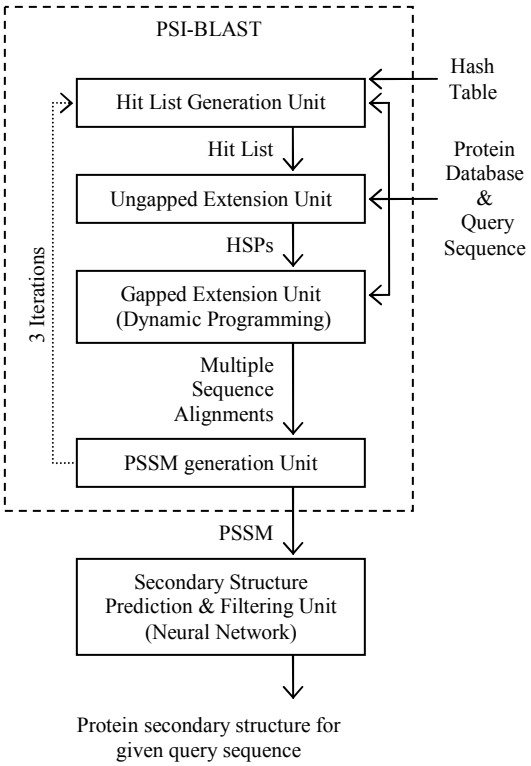


Figure 1. System Architecture for PSIPRED

A. *PSI-BLAST*

Basic Local Alignment Search Tool (BLAST) [8] is a heuristic tool based on dynamic programming [9] to find sequence similarities in protein and DNA sequences. The central idea of BLAST is that any statistically significant alignment between query and database sequences must have at least one word pair of length *w* scoring above a threshold *T*. Once the location of this word pair is known, it is extended on either side to obtain High Scoring Pair (HSP) of aligned sequences with a score S. This score is obtained from the BLOSUM-62 [10] substitution matrix by comparing the aligned amino acids in the HSP. PSI-BLAST [7] is an iterated version of BLAST which generates a Position Specific Scoring Matrix (PSSM) from gapped alignments between multiple sequences. Database searches using this matrix are more sensitive and are better able to detect weaker homologies than pairwise comparison methods [11]. The key steps of PSI-BLAST are summarized below:

1) A hash table is generated which contains a list of words scoring above threshold *T* when aligned with a word in the query sequence.

2) The protein sequence is scanned using the hash table and a hit list is generated containing the locations having two high scoring words in close neighborhood.

3) A list of High Scoring Pairs (HSP) is generated corresponding to the hit list locations having a high score S when extended. This score *S* is normalized as $S' = (\lambda S - \ln k)/\ln 2$, where, λ and k are statistically determined parameters.

4) The HSPs are extended using dynamic programming to generate local alignment with error $< E = mn/2^{S'}$ where *m* is the length of the query sequence, *n* is the length of the database sequence and *S'* is the normalized score.

5) The steps 2 to 4 are repeated for all the database sequences. A multiple sequence alignment is obtained which is used to calculate new scores for the PSSM matrix (of size 20 x length of query sequence).

6) Steps 1 to 5 are repeated three times after which the PSSM matrix is read by the neural network for secondary structure prediction.

B. *Secondary Structure Prediction*

The secondary structure prediction step of PSIPRED computes for each amino acid the confidence level that it is likely to be an α, β or γ structure. This stage consists of two neural networks connected in a feed forward configuration. The first neural network (trained using non-redundant set of protein sequences) scans a window of (2n+1) amino acids and predicts the secondary structure of the central element of window depending on the amino acid's 'n' preceding and 'n' succeeding amino acids. The second neural network filters the erroneously predicted and structurally improbable profiles.

In the hardware implementation of the neural network, the synaptic weights are calculated using matrix multiplication. The input vector is multiplied by weights stored in the network and products are summed up to give outputs based on the sigmoid activation function. Back propagation algorithm is used for offline training.

III. PSIPRED ARCHITECTURE

In this section we describe the details of the coprocessor architecture to implement PSIPRED. The main processor provides the coprocessor with the protein database, the query sequence, the hash table, and the statistical constants. The hash table is a list of words with scores greater than threshold *T* when aligned with some word from the query sequence (using BLOSUM-62 scoring matrix). The hash table helps the coprocessor to scan the database at very high speeds. The coprocessor performs sequence matching and multiple sequence alignment using PSI-BLAST to generate PSSM. It then processes the PSSM using neural networks to predict the secondary structure in terms of α, β and γ structures. The architectural modules and interconnections between the modules of the coprocessor are shown in fig. 2.

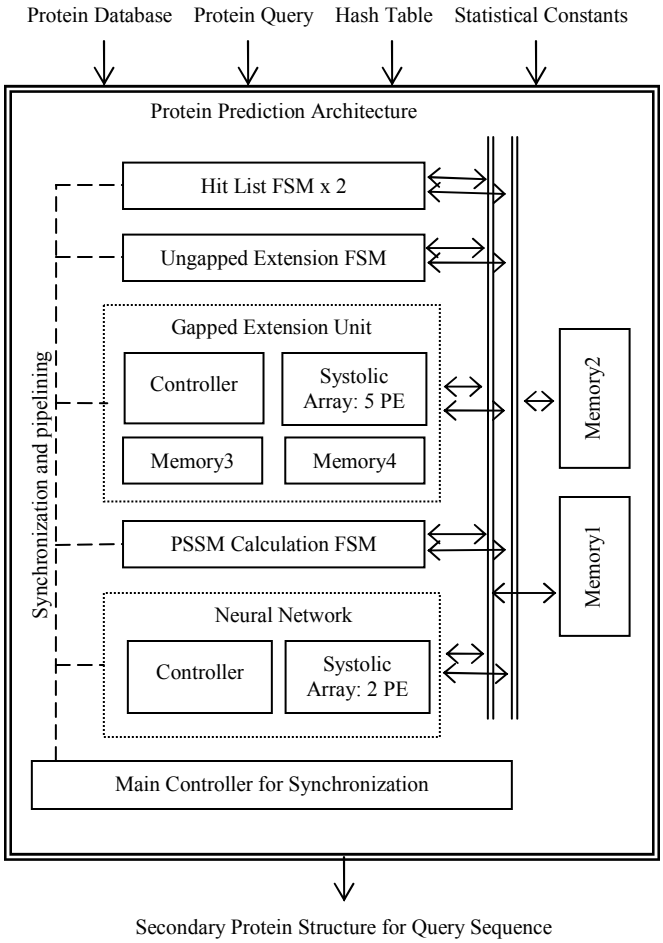


Figure 2. Proposed architecture for PSIPRED

A. *Hit List FSM*

Hit list is a list of most probable alignments between database and query sequence which could generate a High Scoring Pair (HSP). The Hit List FSM shown in fig. 2 reads the hash table and database sequence from the Memory1 and generates a list of pairs; each pair consists of a query sequence word (in Memory1) and the corresponding database sequence word locations which satisfy the two-hit criteria (in Memory2). This step is the most time consuming process and so we duplicate the hardware to process two sequences in parallel. The FSM can be summarized as follows:

1) Read a word from database sequence (for proteins, a word consists of three consecutive amino acids)

2) Read hash table entry of that database word to get pointer to list of hit locations

3) Read the list of hit locations and corresponding alignments

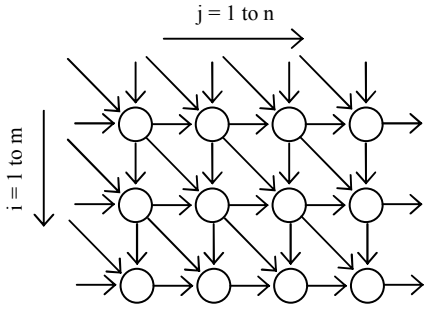
4) Check two-hit criteria; if the new hit locations are within distance *A* (generic constant) of the previous hits on same alignment then it is most probably a HSP

5) Report the database locations and query locations that match two-hit criteria

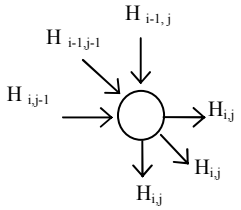
B. *Ungapped Extension FSM*

The ungapped extension unit processes the hit list to generate a list of statistically significant HSPs (refer to fig. 1). The hits which are reported by hit list FSM as the most probable HSP locations are now extended in both directions till the score falls below the best score found for shorter extensions (in any direction) by a certain value *X* (generic constant). The pair-wise score is calculated by comparing the database amino acid sequence with the query amino acid sequence using BLOSUM-62 matrix [10].

C. *Gapped Extension Unit*



(a)



$$H_{(i,j)} = \max\{H_{(i-1,j-1)} + S_{ij}, H_{(i-1,j)} + g, H_{(i,j-1)} + g\}$$

(b)

Figure 3. (a) Systolic array used to calculate edit distances (b) Individual node description

Gaps are introduced in the local alignment using the dynamic programming algorithm of [12]. The edit distance H_{ij} between the amino acid query sequence *i* and the database sequence *j* is calculated by a systolic array configuration. Fig. 3 (a) shows the data flow graph of the systolic array. Here *m* is the length of the query sequence and *n* is the length of the database sequence. The edit distance defined as the number of insertion/deletion and substitution operations required to convert one string into the other, is described in fig. 3 (b). Here S_{ij} is the replacement cost obtained from BLOSUM-62 which can be high for amino acids that share a common ancestor and low (negative) for those which are a result of several mutations.

Also g = -4 is the cost of inserting/deleting a character from the string.

In the systolic array architecture, one query character from Memory1 is mapped onto each PE along with the corresponding row from BLOSUM-62 (which is stored in Memory3). The database sequence from Memory2 is then scanned through the array to compute the alignment scores. If *k* is the number of available PEs and k < m, then the data flow graph is partitioned using the Locally Parallel and Globally Sequential (LPGS) scheme into *k* blocks. The switching of blocks is controlled by a state machine which stores the intermediate signals in Memory4. Other state machine controllers supporting this systolic array architecture include a FSM to fetch a query character every cycle from the Memory1, a FSM to fetch database characters from Memory2, and a FSM to fetch substitution scores from Memory3 every cycle.

Once the query and database sequences have been scanned through this array, backtracking is initiated. This is done by finding the largest score calculated in the matrix and tracing backwards. At the end of backtracking, the pairwise alignment between the two sequences is obtained. This procedure is repeated for several database sequences and a multiple alignment is thus obtained.

D. PSSM Calculation Unit

The multiple sequence alignment obtained from BLAST has *m* columns and *x* rows, where *m* is the length of the query sequence and *x* is the number of database sequences present in the multiple alignment. The PSSM matrix calculated from the multiple alignment also has *m* columns but 20 rows, one for each amino acid. For *m* different columns, the substitution score in each column should be different. This is because the scores depend not only upon amino acids appearing in that column but also appearing in other columns. The procedure is implemented in the following way

1) First a state machine is used to prune the multiple alignment so that all rows that are >98% identical to the query are purged, and only one copy is retained.

2) A simpler reduced multiple alignment Mc is constructed for each column. This is a set of sequences that have an amino acid present in that particular column.

3) The weighted frequency of an amino acid f_i is calculated as the observed frequency of the amino acid in a particular column averaged with the weight of the sequence [13].

4) The pseudo-count frequency g_i of an amino acid is calculated as

$$g_i = \sum_j \frac{f_j}{P_j} q_{ij}$$

where P_j is the background probability of amino acid *j* occurring in a sequence and q_{ij} is the target frequency which depends upon substitution scores between amino acids *i* and *j* in BLOSUM-62 matrix. Amino acids favored by the substitution matrix get higher pseudo-count frequencies.

5) The position specific score is then calculated as log (Q_i/P_i) where P_i is the background probability and Q_i is the estimated probability of i^{th} amino acid in the desired position

$$Q_i = \frac{\alpha f_i + \beta g_i}{\alpha + \beta}$$

where α = Nc – 1 (where *Nc* is the mean number of different amino acid types in various columns of the reduced multiple alignment Mc), and β is an arbitrary weight, empirically calculated as 10.

The scores obtained from the one iteration are used in the next iteration of PSI-BLAST with minor modifications. This process is repeated for three iterations after which the scores are sufficiently sensitive to be used by the neural network for secondary structure prediction.

E. Secondary Structure prediction and filtering unit

The neural network is used in two capacities: for secondary structure prediction from PSSM and for filtering the prediction. The first neural network (containing one hidden layer) is a feed forward neural network which looks at a window of 15 amino acids to predict the secondary structure for the central element. For 20 amino acids plus an entry to mark end of sequence, there are 315 input neurons (15x21) and three output neurons corresponding to the three types of secondary structures. According to the heuristics in [6], 75 hidden neurons give sufficiently accurate results and hence we program the architecture to implement a 315 x 75 x 3 layer neural network.

The second feed forward neural network contains one hidden layer and looks at a window of 15 secondary structures (predicted by the first neural network) to give a filtered output for the central element. For three types of secondary structures and an end marker, we require 60 input neurons (15x4) and three output neurons. For this neural network, 60 hidden neurons are found to give sufficient accuracy and hence the architecture is configured to implement a 60 x 60 x 3 layer neural network [6].

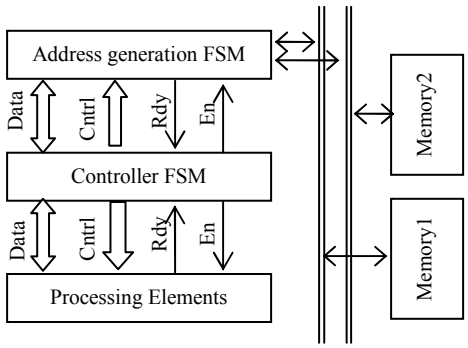


Figure 4. Neural Network Architecture

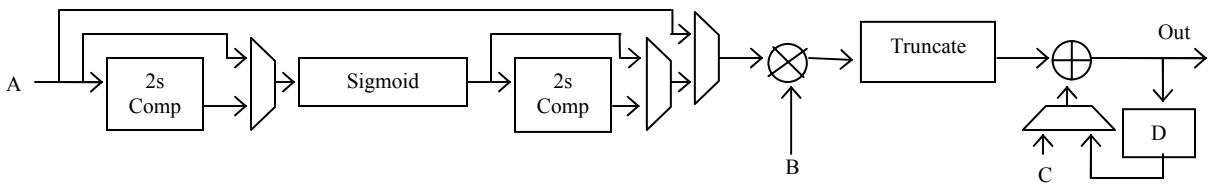


Figure 5. Processing Element for Neural Network

The neural network used for secondary structure prediction can be implemented as shown in fig. 4. The operation. To provide high speed data access, two memories are provided as shown in fig. 4. The controller FSM synchronizes the operations between the processing elements and the memory interface.

The neural network has two modes of operation; prediction and training. Since the training is an offline routine (using back propagation algorithm), the online prediction of protein secondary structures can be made faster by increasing the number of processors in the systolic array. The processing element described in fig. 5 can be configured to carry out MAC operation during the prediction run and can carry out addition and multiplication operations during training run. It has sigmoid processing available on one of the inputs (A): F(A)=A or sigmoid(A). This simple processing element can be used to implement multiple operations on inputs A, B, C and register D as shown below.

Reset : A = 0 B = 0 C = 0 => F(A) * B + C = 0
MAC : A = in1 B = in2 => F(A) * B + D => D
Add : A = 1 B = in1 C = in2 => B + C
Multiply: A = in1 B = in2 C = 0 => A * B

The sigmoid activation function scales input in the range -8 to +8 non-linearly into a value between 0 and 1. For simplifying the hardware, the sigmoid function can be calculated using 15 segment piecewise linear approximation (PWL). To approximate sigmoid function for numbers between -8 to 0, the number is first converted to a positive number and then approximated using the PWL method and then converted back to the negative number [14].

F. Main Controller

The main controller serves the purpose of coordinating and synchronizing the protein structure prediction algorithm from PSI-BLAST to updating PSSM to neural network processing. The controller synchronizes the units shown in fig. 2 using handshaking signals, and provides parameters such as memory offsets, constants, input dataset and dimensions.

IV. RESULTS

The VHDL model of the architecture described in section III was compiled and simulated using Cadence NcLaunch and SimVision 05.10-s006. The architecture was synthesized using Synopsis DC Compiler v2002.05 and 0.18µm CMOS technology. Table I summarizes the area processing elements of the neural network are configured as a systolic array and are used for matrix multiplication and memory for each unit. The total area is 783,228 units and the memory requirement is 225.4KB.

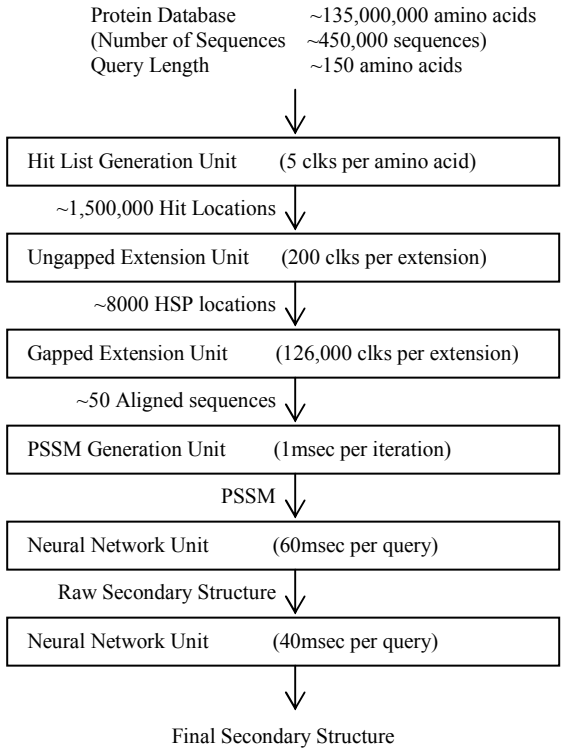


Figure 6. Data and timing details for processing a 150 amino acid sequence on a 100 MHz coprocessor

Next we consider protein sequence matching of a query sequence of length ~150 amino acids with a protein database with ~135,000,000 amino acids or ~450,000 sequences. The time consumed in each architectural module when clocked at 100 MHz is given in fig. 6. The total time taken is ~11 seconds which is a small fraction of what it takes on a high end general purpose processor.

Before the final architecture was designed, a dataflow analysis was done to identify the bottlenecks. For instance, since the hit list generation was very computationally intensive, we implemented two hit list generation units to speed up the calculation. Similarly, the gapped extension unit was implemented with a systolic array consisting of

five processing elements. Since the volume of data used by the neural networks was quite small, we implemented the neural network architecture with only two processing elements. Thus careful consideration of both the area and timing was done in the design of this architecture.

TABLE I. SYNTHESIS RESULTS

Hardware Unit Description	Area (Unit Gates)	Memory (KB)
Hit List Generation Unit	93,158	10
Ungapped Extension Unit	55,028	16
Gapped Extension Unit	329,270	97.8
PSSM Calculation Unit	143,930	1.6
Neural Network Unit	161,842	100

V. CONCLUSION

This paper presents a coprocessor architecture for implementing one of the most popular protein structure prediction algorithms, PSIPRED. We achieve several orders of speedup in algorithm execution time by pipelining the operations, exploiting the pipelining and parallelism capability of systolic arrays, and optimizing the FSM implementations. We have implemented the PSIPRED algorithm in VHDL and synthesized using Synopsis with 0.18µm CMOS technology. The synthesized architecture requires 783,228 gate area and 226KB memory. The coprocessor is clocked at 100 MHz and can predict protein structures of average sized query sequence in a fraction of a minute.

Future work will include aggressively exploiting parallelism to reduce prediction time by up to 50% with less than 25% increase in area. It will also include optimizing the synthesized architecture so that it can be clocked at 150 MHz. Our final goal is to develop a programmable architecture that is capable of supporting multiple protein structure prediction algorithms.

REFERENCES

[1] R. H. Kretsinger, R. E. Ison, S. Hovmoller, "Prediction of Protein Structure," Methods in Enzymology, Academic Press 2004, vol. 383, pp. 27.

[2] P. Y. Chou, & G. D. Fasman, "Conformational parameters for amino acids in helical, sheet, and random coil regions calculated from proteins," Biochemistry, vol. 13, pp. 211-222, 1974.

[3] B. Rost, & C. Sander, "Prediction of protein secondary structure at better than 70% accuracy", J. Mol. Biol., vol. 232, pp. 584-599, 1993.

[4] C. Geourjon, & G. Deleage, "SOPMA: significant improvements in protein secondary structure prediction by consensus prediction from multiple alignments," Comp. Appl. Biosci., vol. 11, pp. 681-684, 1995.

[5] D. T. Jones, "Successful ab initio prediction of the tertiary structure of NK-lysin using multiple sequences and ecognized supersecondary structural motifs," Proteins: Struct. Funct. Genet., vol. S1, pp.185-191, 1997.

[6] D. T. Jones, "Protein Secondary Structure Prediction Based on Position-specific Scoring Matrices," J. Mol. Bio., vol. 292, pp. 195-202, 1999.

[7] S. F. Altschul, T. L. Madden, A. A. Schaffer, J. H. Zhang, Z. Zhang, W. Miller, & D. J. Lipman, "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs," Nucl. Acids Res., vol. 25, pp. 3389-3402, 1997.

[8] S. F. Altschul, W Gish, W Miller, E. W. Myers, & D. J. Lipman, "Basic Local Alignment Search Tool," J. Mol. Bio., vol. 215, pp. 403-410, 1990.

[9] R. A. Wagner, & M. J. Fischer, "The string to string correction problem," J. of the ACM, vol. 21, pp. 168-173, 1974.

[10] S. Henikoff, & J. G. Henikoff, Natl. Acad. Sci. USA, vol. 89, pp. 10915-10919, 1992.

[11] R. L. Tatusov, S. F. Altschul, & E. V. Koonin, "Detection of conserved segments in protiens: Iterative scanning of sequence databases with alignment blocks," Proc. Natl. Acad. Sci. USA, vol. 91, pp. 12091-12095, December 1994.

[12] T. F. Smith, & M. S. Waterman, "Identification of common molecular subsequences," J. Mol. Biol., vol. 147, pp. 195-197, 1981.

[13] S. Henikoff, & J. G. Henikoff, "Position based sequence weights," J. Mol. Biol., vol. 243, pp. 574-578, 1994.

[14] P. Murtagh, A. C. Tsoi, "Implementation issues of sigmoid function and its derivative for VLSI digital neural networks," IEEE Proceedings - E, vol. 139, no. 3, pp. 207-214, May 1992.