Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: 28 April 2018
Manuscript Category

OXFORD

Building a Profile Hidden Markov Model for the Kunitz-type protease inhibitor domain

Cortile Clelia [1],

[1] Bioinformatics, Bologna University, Bologna, 40121-40141, Italy

Abstract

Motivation: The recent high-through-put sequencing technologies generated an huge amount of the sequence data that remain unknown. With no information about the sequences, they are not useful so there is the need to use specific methods that allow the researchers to manipulate and classify the huge amount of possible protein sequences. This job became doable with the use of protein predictor that simplifies a job much. Indeed, it allows the researchers to analyze several sequences at the same time and understand which one could be classified in a specific protein family. Obviously all the predictors can make errors in the prediction but in general a good predictor tries to minimize them.
Results: We built a Profile Hidden Markov Model able to predict the presence of the Kunitz domain in a sequence. The predictor is trained using a set of Kunitz proteins (Pfam ID:PF0014) which are structurally similar. We tested the predictor using two benchmark sets from which we derived statical informations to evaluate in which E-value range it is able to do its optimal performance. The final results are supported by two statical coefficients: the Accuracy Coefficient and the Matthews Correlation Coefficient.
Availability: All the the scripts and the input files are available at this GitHub repository address: https://github.com/cleliacort/build-HMM-.
Contact: clelia.cortile@studio.unibo.it

1 Introduction

The Kunitz domain is an active short (~50-60 amino acids)[1] domain composed by alpha/beta fold, which contains 3 disulfide bonds, as we can see in Figure 1. This domain inhibits the function of a big enzymatic class, which is that of protease (EC 3.4). In detail, the proteins that contain this domain are classified as serine protease inhibitors (EC 3.4.21)[2]. The most common family that groups all the proteins having the Kunitz domain is the MEROPS inhibitor family[1]. The most studied are aprotinin (bovine pancreatic trypsin inhibitor, BPTI), Alzheimer's amyloid precursor protein (APP)[4] and tissue factor pathway inhibitor (TFPI)[1]. As we can easily understand, the proteolytic activity is essential in the organism, for example to retrieve the necessary amino acids or to obtain a sub-protein product that can interact in a different pathway. Though, when the proteases do not work in the correct way, they can damage the cells and consequentially cause serious problems into the organism, some of which can lead to diseases [3]. For the reason explained above, the development of biopharmaceutical studies is increasing. Their main aim is to design a drug able to recognize specific protein structures and interact with them to work as competitive protease inhibitors. The most common drug derives from the already mentioned bovine pancreatic trypsin inhibitor (BPTI) and it is called Trasylol [5]. Its main function is to reduce bleeding during complex surgery, such as heart and liver surgery. Our project focused on building a method for automatic classification of a specific family, composed by proteins containing the Kunitz domain.

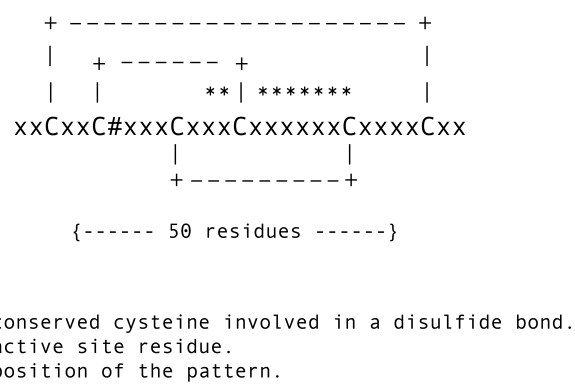

Fig. 1: Scheme of the consensus pattern that allow the formation of three disulfide bonds.

2 Methods

2.1 Database

We used the release of Protein Data Bank (`http://www.uniprot.org/`), submitted in April, to download the sequences which were part of the training set. It contains all the proteins having the Kunitz domain which are retrieved using the X-ray diffraction method with resolution less than 2.0. On this initial dataset, we apply other two main filters related to the sequence length and the chemical components. Regarding the former, we selected a length range that goes from 55bp to 99bp, in order to have sequences that contain a very little number of domains, which differ from the Kunitz domain. Choosing proteins with few domains allows us to manipulate them more easily. As for the latter, we chose the option that excludes from the result those proteins having mutated residuals (Supplementary Query 1). In the end, we found out that this training set contained 89 proteins. As we will demonstrate in the Results section, these filters are not enough to have an optimal starting data set. To obtain it, we added another filter related to the features sequence, able to remove from the result all those proteins having natural or engineered mutations compared to the wild type protein. The reader can find the final search query in the Supplementary Query 2. Therefore, the original training set was modified so that we had a training set containing 40 proteins.

We used the release of Uniprot (`http://www.uniprot.org/`) submitted on April 25, 2018, to search sequences to make two benchmarking sets. The former, also called positive set, contains those reviewed sequences having the Kunitz domain which are 359. Since we used it to evaluate the performance of the method, we decided to remove from it the sequences used for the training set and those that were very similar. The reason is to avoid the possibility that those sequences can drive the results. We will explain better this step on the Result section. Therefore, the final positive set contains 337 sequences. The latter, also called negative set, include all those proteins that do not contain the Kunitz domain. The initial negative set was composed of about 550 thousand proteins. The reader can find the final search query in the Supplementary Query 3. For us, working with this huge amount of data was not convenient in terms of memory and time. Therefore, we decided to reduce it and we selected from the original set only the those proteins having length in range between 50bp and 100bp. The final negative set is composed of 43697 thousand of proteins.

2.2 Tools

We used the latest stable version 2.2.25 of local BLAST [6] submitted on March 31, 2011. In particular, we used two specific programs: `blastclust` and `blastall`. The tool `blastclust` [7] allowed us to cluster the proteins belonging to the training set. It computes all the possible pairwise matches between two proteins belonging to the training set. It uses the BLAST algorithm [8]. As an input, we passed a file containing the FASTA sequences of each protein in the training set. We obtained this file using a shell script [17]. It requires the customize table of the training set as an unique argument. The table comes direclty from the PBD website and reports the PDB ID, the chain ID, the Entity ID and the Sequence. We used the specific option `-S` which allows us to set a score coverage threshold equal to 95. Therefore, the proteins belonging to the same cluster have at least 95 % of identity with each other. The tool `blastall` allowed us to run a local blast search [9] which means to seek the possible matches of a proteins list against a customize database. We did this local search to find those proteins that were similar to the ones in our training set. If we found some significant mathches, we decided to remove those similar proteins from the positive benchmark set because they can misdirect the future statical analysis. We used the basic syntax of `blastall` which requires five options. The `-p` option allowes us to specify the type of comparison. In our case, we set it on `blastp` because we want to make a comparison between an amino acid sequence and a protein sequence database. The `-d` option is needed to specify the database file which must be a list of FASTA sequences. In our case, the database is the FASTA file containing all the Kunitz proteins in the UniProt website. The options `-o` and `-m` are used respectively to specify the name and the format of the output. We decided to set this last option equal to 8 because we wanted the results in tabular format in order to better manipulate them.

To compute the multiple sequences alignment, we used the on-line tool called MUSCLE (`https://www.ebi.ac.uk/Tools/msa/muscle/`). In comparison with the other tools for the multiple sequence alignment like CLUSTAL and T-Coffee, it has a better accuracy and speed [10]. As an input, it needed a unique file containing the FASTA sequences. We used MUSCLE with the data coming from the search with Query 1 and also with those coming from the Query 2. We did this work to check if all the proteins selected for the training set contained the typical consensus pattern of Kunitz domain (six Cys inside the sequence). We left all the parameters as default. As an output, we had the multiple sequence alignment in the ClustalW format.

We used the on-line tool called Skylign [11] to graphically visualize the conservation pattern of a set of sequences. As an input, we used the multiple alignment generated with MUSCLE tool. We used the current on-line version of PDBeFold (`http://www.ebi.ac.uk/msd-srv/ssm/cgi-bin/ssmserver`) to retrieve a multiple structural alignment of our training set. It uses an efficient SSM (Secondary Structure Matching) algorithm [13] that makes the multiple structure alignment possible among at most 100 sequences. It is based on the identification of residues that occupy geometrically equivalent position in all the aligned structures. It can be obtained properly rotating and translating the structures, until obtaining the optimal one, so the one that has a better alignment score, is found. As an input, we passed a text file containing the ids of all the proteins belong to our training set. Since the input file is a list of ids, we set the section Source on "List of PDB". As an output, the software produced a file containing the multiple structural alignment and gave us an HTML page reporting much information about it. We mainly focused on the overall RMSD value and on the matrix containing the RMSD for structural superimposition between all the possible pairwise among our training set.

To visualize the multiple structural alignment produced by PDBeFold, we used the version 2.7.5 of RasMol [14]. It is a program that allows us to see molecular graphics.

We used the version v3.1b2 of HMMER submitted on February 22, 2015. It is a free software implementation of profile HMMs, which allows us to analyze biological sequences [15]. It is a collection of programs and we used two of them: `hmmbuild` and `hmmsearch` [16]. The `hmmbuild` allows us to build a Profile-HMM (Hidden Markov Model) starting from a multiple alignment. The making of the Profile-HMM comes from the capturing of position-specific information about how conserved each column of the alignment is, and which residues are likely. As an input, it needed a multiple alignment that could be based on the sequences or on the structures. In this work, we used the multiple structural alignment. We ran the `hmmbuild` program using the basic syntax which includes name of the multiple alignment and the name of the output file which will be the HMM profile. The `hmmsearch` allows us to search the specific profile against a sequence database in FASTA format. Its basic syntax consists in two input files, which are the HMM-Profile and the sequence databse, and the name of the output file. We also used this program with specific options like the `-noali` that allowed us to remove all the pairwise alignments from the ouput file in order to reduce its volume size of it; the `-E` option that returns us an output file containing only the sequences having an E-value lower than the one chosen; the `-max` option that turns off all the default filters.

To measure the performance of our model, we did a benchmark test. We ran the `hmmsearch` program against the positive and negative test set.

We used the option `-E` and assigned it a high value to have in the output file all the proteins that are present in the input file with the corresponding E-value. Then we manipulated this output file to retrieve the information in which we were interested. Using a specific shell script [18], we retrieved a confusion matrix for ten different E-value thresholds. In our work, the number of the TP (True Positive) corresponds to those proteins of the positive test set that are predicted as Kunitz proteins; the number of the FN (False Negative) corresponds to those proteins of the positive test set that are not predicted as Kunitz protein; the number of FP (False Positive) corresponds to those proteins of the negative test set that are predicted as Kunitz proteins but do not being; the number of TN (True Negative) corresponding to those proteins of the negative test set that are correctly predicted as not Kunitz proteins.
To support the results of the confusion matrices, we computed the Accuracy Coefficient(AC) and the Matthews Correlation Coefficient(MCC) using a specific python script [19] . The former is a measure of statistical bias which is a quantification of how much the results differ form the estimated value. A good level of accuracy is equal to have AC as close to one. The latter measures the correlation between observed and predicted results. If the MCC is close to one, it means that the two pieces of data are correlated. The reader can find the formula used to compute these coefficients in the specific python script [19].

3 Results

3.1 Improvement of the Traning set

Using `blastclust` on the training set we found 13 clusters. We chose one representative protein for each cluster to build the set with which train the HMM profile. We decided to catch the protein having the biggest length as the most representative one for each cluster. The reader can find the IDs list on Supplementary Table 1. We did the MUSCLE multiple alignment on this protein set to check if all of them had the main characteristic of a Kunitz protein which is having six Cys to make the three disulfide bonds. To easily understand this, we used the output to generate a logos representation with Skylign. The logos allowed us to visually discover if there were some cysteines which were not conserved among among all the proteins. As we can see in the Figure 2, four cysteines are not conserved in all the sequences. Analyzing the multiple alignment, we discovered that two proteins (3AUB:A, 2ZVX:A) do not contained all the six cysteines. The reader can easily identify these two proteins in the Supplementary Figure 1.

Fig. 2: Logos representation of the multiple sequences alignment of proteins coming from the search with Query 1.

After this, we tried to understand the reason why these two proteins were in our training set. Consulting their PDB page, we figure out that they are classify as Kunitz proteins but they contain engineered mutations on some Cysteines. For our purpose, these proteins were not useful. Therefore,we refined the search to avoid the presence of these kind of proteins form our results. We added a new restriction on the previous query and repeated the same work explained above on this new proteins set. Using `blastclust` on this new set, we found 13 clusters. We did the multiple alignment and used it to generate the logo representation with Skyling that we can see in the Figure 3. Since the two clusters had different size, they cannot be comparable in terms of probability values related to the conserved cysteines. Therefore, we did a sort of general visual comparison to understand if the conservation of cysteines were not maintained such as in the previous proteins training set. We found that also in this new set some proteins did not contain all the six cysteines. Analyzing the multiple alignment, we saw that only one protein (3WNY:A) did not have all the conserved cysteines. So the two alignment differ form one protein. Since the difference in size set was not significant and we noticed that by using the second set we could remove one protein that does not hold all the cysteines, we decided to work with this last set. It was the set with which we make the HMM profile. The reader can find the IDs list on Supplementary Table 2.

Fig. 3: Logos representation of the multiple sequences alignment of proteins coming from the search with Query 2.

3.2 Structural Alignment

Before starting to build the HMM profile, we did a structural alignment which allow us to check if all the proteins in the training set were also structurally similar. We did the structural alignment using the on-line PDBeFold. From its output, we noticed that the overall RMSD was about 0.7008. This means that all the proteins are structurally similar among them. We can graphically see the superimposition among all the proteins in the Figure 4, which is obtained through Rasmol tool. The structural similarity is also confirmed from the RMSD of each possible pairwise structural superimpositions among all proteins belonging the training set. The reader can find the RMSD values for each combination into the Supplementary Figure 3. The noticed structural similarity, allow us to conclude that the absence of the whole consensus pattern in the 3WNY:A do not cause any changes in the function. The conservation of the structure and consequently the conservation of the function among all the proteins belonging to the training set are very important to build an optimal HMM profile. Indeed, it is obtained when we train the HMM profile on proteins having the same function.

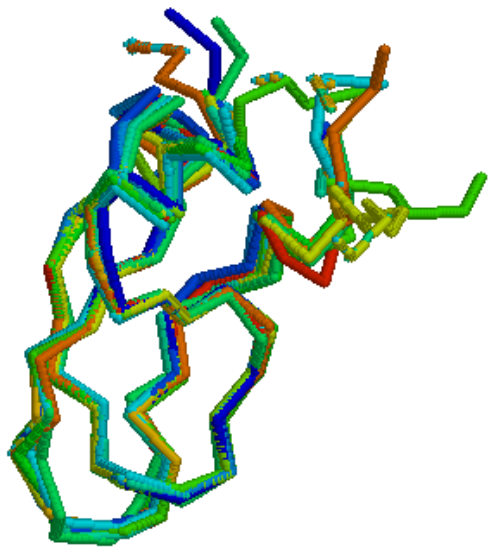

Fig. 4: Superimposition of all the sequences belonging to the training set.

3.3 Building the Profile-HMM

The most important step was building the HMM model which was made using the `hmmbuild` program. In a quick way, we checked the quality of it doing a local search with `hmmsearch` program. The input file contained all the proteins belonging to the training set. As we could expected, the model predicted all those proteins being a Kunitz protein.

3.4 Computation of the Evaluation Scores

We made the confusion matrices for ten different E-value and we reported the six most significative in the Table 1. From these data, we understood that our HMM-Profile can perform an optimal work if the E-value threshold is set to $10e^{-5}$. Using the HMM-Profile with this E-value the amount of FN and FP predicted is minimized, as we can read from the same Table 1. Based on the confusion matrix, we computed two other statistical measures which are the AC and the MCC to better support the chosen threshold. At $10e^{-5}$ E-value, these coefficients are very close to one so our obtained data are very close to those expected. The coefficients are very close but not equal to one because we there is the presence of one FN.

Table 1. Confusion Matrix with related AC and MCC for six selected E-value threshold.

Exp	TP	FN	FP	TN	AC	MCC
$10e^{-4}$	336	1	2	43695	0.9999	0.9955
$10e^{-5}$	336	1	0	43697	0.9999	0.9985
$10e^{-6}$	335	2	0	43697	0.9999	0.9970
$10e^{-7}$	335	2	0	43697	0.9999	0.9970
$10e^{-8}$	335	2	0	43697	0.9999	0.9970
$10e^{-9}$	334	3	0	43697	0.9999	0.9955

4 Conclusion

The aim of our work was to make an HMM-Profile able to predict the Kunitz proteins. We started from a training set composed by 89 Kunitz proteins. We cleaned this dataset and removed the redundancy by clustering all the sequences and choosing the one with the best resolution. Grouping all of these selected proteins, we obtained the final training set which is composed by 40 proteins. This was to make the HMM-Profile which is the predictor for the Kunitz proteins. Once we built it, we tested the performance of the this predictor. Therefore, we made two benchmarking sets, one containing all the Kunitz proteins (positive test set) and the other with all those non Kunitz proteins(negative test set) having a specific length. Using these two benchmarking sets, we computed the confusion matrix for ten different E-values and we found the relative value of TP,FN,FP,TN. The goal of doing that was to understand what could be the range/the unique value in which the number of errors was minimized which means having the lower number of FN and FP predicted. At the conclusion of our work, we can declare that we created an HMM-Profile that correctly predicts the Kunitz proteins when the E-value is $10e^{-5}$. At this E-value threshold, the relative AC and MCC are close to one so the obtained data are very similar to those expected.

5 Additional file

Supplemental Materials

References

[1]Pfam website: `https://bit.ly/2KmDMET`

[2]InterPro website: `https://bit.ly/2FMoupz`

[3]N.D. Rawlings, D.P. Tolle, A.J. Barrett; *Evolutionary families of peptidase inhibitors. Biochemical Journal* (2004). 378(Pt 3). Pages 705âĽ716.

[4]K. Ikeo, K. Takahashi, T.J. Gojobori; *Evolutionary origin of a Kunitz-type trypsin inhibitor domain inserted in the amyloid beta precursor protein of Alzheimer's disease*(1992). Mol Evol. 34:536.

[5]Wikepedia page of Aprotinin: `https://bit.ly/2wpoegP`

[6]Information about local BLAST url: `https://bit.ly/2HUDfMI`

[7]Blastclust usage website: `https://bit.ly/2HzJqWz`

[8]F.A. Stephen, W. Gish, W. Miller, E.W. Myers, D.J. Lipman; *Basic local alignment search tool* (1990). Journal of Molecular Biology. Volume 215. Issue 3. Pages 403-410.

[9]Blastall usage website: `https://bit.ly/2FNFuvw`

[10]RC. Edgar; *MUSCLE: multiple sequence alignment with high accuracy and high throughput* (2004). Nucleic Acids Research. 32(5):1792-1797.

[11]T.J. Wheeler, J. Clements, R.D. Finn; *Skylign: a tool for creating informative, interactive logos representing sequence alignments and profile hidden Markov models.* (2014). BMC Bioinformatics. Volume 15. pag. 7.

[12]E. Krissinel and K. Henrick; *Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions* (2004). Acta Cryst. D60, 2256—2268.

[13]E. Krissinel, K. Henrick; *Multiple Alignment of Protein Structures in Three Dimensions* (2005). In: R. Berthold M., Glen R.C., Diederichs K., Kohlbacher O., Fischer I. (eds) Computational Life Sciences. CompLife 2005. Lecture Notes in Computer Science, vol 3695. Springer, Berlin, Heidelberg.

[14]Rasmol website: `https://bit.ly/2ro159v`

[15]P. Vinuesa; *Bioinformatics exaplained: HMMER* (2007), `https://bit.ly/2KFvXdF`.

[16]E. Sean; *HMMER2 User's Guide (PDF)* , `https://bit.ly/2HRn9TX`.

[17]Script to obtained a file containing the FASTA sequences starting from the customize table of UniProt: `http://doi.org/10.5281/zenodo.1257913`

[18]Script to make the confusion matrix: `https://doi.org/10.5281/zenodo.1258020`

[19]Script to compute the AC and the MCC: `https://doi.org/10.5281/zenodo.1258024`