Cascade Support Vector Machines applied to the Translation Initiation Site prediction problem

Wallison W. Guimarães[1], Cristiano L. N. Pinto[2], Cristiane N. Nobre[1], Luis E. Zárate[1]

[1] Pontifical Catholic University of Minas Gerais, Brazil
wallisonsgp@gmail.com
{nobre, zarate}@pucminas.br
[2] School of Engineering of Minas Gerais, Brazil
cristiano@emge.edu.br

Abstract. The correct identification of the protein coding region is an important and latent problem of biology. The challenge is the lack of deep knowledge about biological systems, specifically the conservative characteristics of the messenger Ribonucleic Acid (mRNA). Thus, the use of computational methods is fundamental to discovery patterns within the Translation Initiation Site (TIS). In Bioinformatics, machine learning algorithms have been widely applied, among them we have the Support Vector Machines (SVM), which are based on inductive inference. However, the use of SVM incurs a high computational cost when applied to large data sets, and its training time scales up to quadratically in relation to the data set size. In this study, to tackle this challenge and analyse the algorithm's behavior, we employed a Cascade SVM approach to the TIS prediction problem. This strategy proposes accelerating the model training process and reducing the number of support vectors. The results achieved in our study showed that the cascaded SVM approach is able to significantly reduce model training times while maintaining accuracy and F-measure rates similar to the conventional approach (SVM). We also demonstrate the scenarios in which the cascade approach is more suitable for reducing training time.

Categories and Subject Descriptors: H.2 [**Database Management**]: Miscellaneous; H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: Translation Initiation Site, Cascade SVM, Data Mining, Machine Learning.

1. INTRODUCTION

The prediction of Translation Initiation Site (TIS) from a Ribonucleic Acid Messenger (mRNA) is a relevant and latent problem of molecular biology, which has benefited from the evolution of computational techniques. The correct prediction of TIS is an important task, and a high accuracy rate in its prediction may aid in the understanding of protein-coding from nucleotide sequences. However, this is not a trivial task, since lack of knowledge of conservative characteristics to identify the translation start site makes the TIS prediction problem complex.

There are several repositories like RefSeq. [Pruitt and Maglott 2001] which contain molecules of mRNA, DNA and proteins of various organisms, and the amount of these molecules is continuously growing. In this work, 113,011 mRNA molecules from six eukaryotic organisms are considered. From the molecules, typically nucleotide sequences are extracted to represent TIS and non-TIS classes to compose the vector of characteristics used by the classifiers. According to the different methodologies

We appreciate the financial support given by Foundation for Research Support of the State of Minas Gerais/Brazil (FAPEMIG), the Brazilian National Council for Scientific and Technological Development (CNPq), and the Coordination for the Improvement of Higher Education Personnel (CAPES). We also would like to thank the Supercomputing National Center (CESUP) of the Federal University of Rio Grande do Sul (UFRGS) for making available the computational resources for the execution of the experiments.

for obtaining the sequences and the number of nucleotides (N) of each of them, an amount of up to 4^N can be reached. Analyzing a large volume of sequences can have a high computational cost which would require strategies to overcome this problem and guarantee the quality of the models.

One of the first prediction works of TIS was developed by Stormo et al. [1982], where the authors used Artificial Neural Networks to predict TIS in prokaryotic cells. Pedersen and Nielsen [1997] also used neural networks to predict TIS using a database with 13,502 eukaryotic sequences. Zien et al. [2000] and Liu and Wong [2003] were first to use the Support Vector Machine (SVM). The authors used the same databases as Pedersen and Nielsen [1997] and demonstrated satisfactory performance of SVM. Works such as Guimaraes et al. [2017] and Pinto et al. [2017] explored the SVM in their experiments both with datasets with approximately 20,000 sequences. In recent work, Zhang et al. [2017] used *Deep Learning* with a data set of approximately 100,000 sequences. Note that the number of sequences considered for constructing the classifications models has increased significantly in order to improve their performance. For example, in our work we consider 100,000 sequences which were extracted with 1081 nucleotides (N) each.

However, it should be noted that SVM is an efficient classification technique, but it has drawbacks when applied to large datasets because its memory consumption can reach quadratic scales $O(N^2)$ in relation to the size of the dataset N [Graf et al. 2004], and cubic $O(N^3)$ in time to find a solution. To try to circumvent this problem, in Graf et al. [2004], the authors developed an approach capable of accelerating the SVM training process. This approach consists of dividing the training data typically into k inputs ($k = 2^n, n = 1, 2, 3, ...$, where 2^n is the desired number of divisions), and for each input an independent SVM is applied. The support vectors resulting from the previous SVMs are merged two by two, and these new datasets are used as input to new SVM's. Such a process creates a cascading structure that repeats itself until only one SVM remains. The Cascade SVM's execution framework enables the training to be done independently, distributed, and with fewer records for each SVM, greatly reduces the training time of the model, in addition to maintaining the quality of the results.

In this work, we applied the Cascade SVM structure to the prediction problem of TIS considering databases with more than 113,011 mRNA molecules. The sequences that compose the training sets, obtained by a 1081 nucleotide mRNA window as suggested by Pinto et al. [2017], represent the positive class (TIS) and the negative class (nTIS). Differently to what was proposed by Pinto et al. [2017], who considered as the nTIS negative sequences the *upstream* sequences out of reading phase (UPOP) in relation to the TIS of the molecule, in this work only the *downstream* sequences out of reading phase with the TIS were considered nTIS [Guimarães et al. 2017], see Fig. 1. A problem with the methodology proposed in Pinto et al [2017] is that it limits its applicability to organisms without the *upstream* region.

Using the Cascade SVM allows us to analyse the behavior and applicability of the strategy in the scenario of large TIS prediction datasets. Another applicable approach would be specialized algorithms, typically based on gradient descent methods, that achieve impressive gains in efficiency, but still become impractically slow for problem sizes in the order of 100,000 training vectors (2-class problems)[Graf et al. 2004]. To validate the scalability of the cascade strategy for TIS prediction, the present work also proposes the training of a single model containing all the sequences of all organisms studied. In experimental results, training of this model containing about 100,000 sequences spent approximately 6 hours through the standard SVM method, whereas with our approach based on Cascade SVM the computational time was reduced 15 times. To the best of our knowledge, the use of this structure in this context of application has not been found in the literature. In addition, a new methodology for sequence extraction is proposed that reduces the aforementioned limitations of the Pinto et al. [2017] strategy.

By applying the cascade SVM strategy to TIS prediction, we sought to answer the following questions: What impacts are brought by the approach of dividing a large problem into several smaller problems? Is the quality of the final model inferior to the model constructed in the conventional approach? Does the reduction in training time always occur? The more divisions are made, the faster is the training of the model? Is the cascade strategy ideal for all analyzed organisms? Thus, there are several issues motivating our analysis of the behavior of the cascade SVM applied to the prediction of TIS.

This article is organized as follows: Section 2 is showed the TIS classification problem. Section 3 presents related works. Section 4 provides a discussion of the materials and methods used. The experimental results are shown in Section 5. Finally, Section 6 contains our conclusions and future work.

2. THE TRANSLATION INITIATION SITE PREDICTION PROBLEM

Molecular Biology is an area of biology that studies what happens in the cell at the molecular level, analyzing the relationship between DNA, RNA and protein synthesis. According to the Dogma of molecular biology, information is perpetuated through DNA replication and is translated through two processes: *transcription* which converts the DNA information into a complementary RNA strand, and through *translation*, which converts the information contained in RNA into proteins.

Thus, the translation and transcription processes of the mRNA sequences are used by cells to transmit and express their genetic information. However, only a few parts of the transcript sequence carry the information which is necessary to, in fact, encodes the proteins. These sequences are called *CoDing Sequences* (CDS). Determining whether a given mRNA "ribbon" does or does not contain the CDS region is considered a central problem of molecular biology [Zien et al. 2000].

In eukaryotes[1], the CDS region is delimited by flags named *start codon* and *stop codon* (see Fig. 2). The *start codon*, identified by the AUG triple, also known as Translation Initiation Site (TIS), is responsible for the beginning of the process of protein synthesis, which is one of the most important processes in the regulation of gene expression. The *stop codon*, identified through the occurrence of UAA, UAG or UGA, determines the end of the translation process of the protein [Pedersen and Nielsen 1997].

In the work developed by Kozak [1984] a statistical analysis was performed on mRNA sequences from eukaryote cells and showed that some positions of these sequences, relative to TIS, are conservative. The experiments identified a conservative pattern at the -3 and +4 positions of the mRNA. For reference the ATG (AUG) start codon corresponds to +1 through +3 (see Fig. 1). It was defined that position -3, that is, three nucleotides to the left of the TIS in the *upstream* region, conservatively presents a purine, nucleotide A (Adenine) or G (Guanine), 79% of these sequences corresponding to nucleotide A; and that at the +4 position the nucleotide G is found, thus establishing the Kozak consensus. Despite the existence of conservative information indicated by Kozak [1984], these positions are not sufficient for TIS identification. The correct identification of TIS in a sequence of nucleotides is part of an important previous step in the process of discovering the characteristics and functions of proteins.

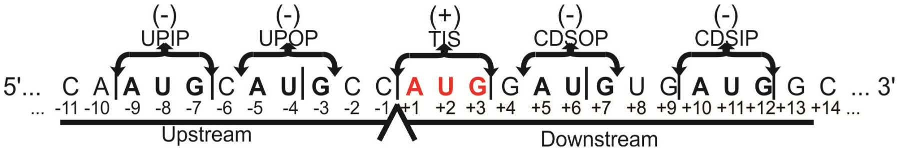


Fig. 1. mRNA sequence with identification of *upstream* and *downstream* regions and reading phases.

[1] Eukaryotes are all living beings made up of cells that have a nucleus.

The translation process often occurs at the first occurrence of an AUG codon[2] [Kozak 1984], but it can also start in different codons depending on the position and context of the sequence [Pedersen and Nielsen 1997]. According to Kozak [1984], in eukaryotes, the scanning model assumes that the link between the mRNA sequence and the ribosome for protein translation initially occurs in the 5' region and goes to the 3' region, see Fig. 2. The position of the beginning of the translation directly influences the produced protein, being able to alter its structure and function in the cellular environment.

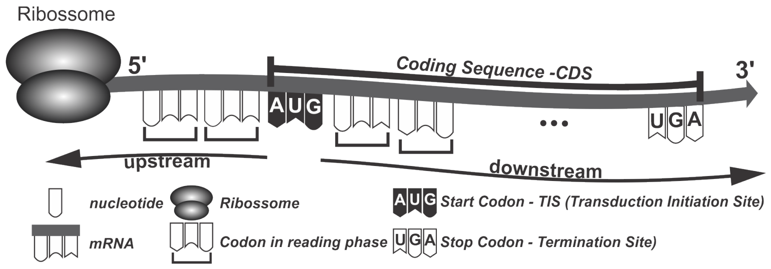


Fig. 2. Model of mRNA scanning in eukaryotes.

The Kozak consensus is used in problems of prediction or classification of TIS for the construction of methodologies based on the extraction of nucleotides, by means of windowing, when collecting sets of sequences for the application of Machine Learning techniques [Silva et al. 2011][Pinto et al. 2017]. As pointed out in [Silva et al. 2011], the window size definition has a direct influence on the quality of the prediction model. The authors establish that windows with an unsymmetric size between AUG codon are adequated for this kind of problem. In the classification methodology proposed in Pinto et al. [2017], the authors defined the beginning of the windowing from the nucleotide -9, that is, 9 positions in the *upstream* region, ensuring that the conservative patterns indicated by Kozak [1984] form part of the extracted window. The generated classification model's results were superior to the models created by *TISHunter*[3], *TIS Miner*[4], and *NetStart*[5], tools already known in TIS prediction. As mentioned, although this methodology has achieved superior results, it limits the number of extracted sequences as well as the number of organisms analyzed, since it discards molecules that do not contain 9 nucleotides in the TIS *upstream* region. In addition, it discards organisms that do not have this sequenced region, as is the case of the organism *Caenorhabditis elegans*, which has only the *downstream* region of the molecule.

Typically, a sequence extraction process begins with the identification of all AUG codons present in the molecule, both in the *upstream* and the *downstream* regions, with only one of them being the TIS and the others being negative sequences (nTIS). The nTIS sequences of the *upstream* region that are read in the same reading phase of the TIS are classified as *upstream in phase* (UPIP), and those that are read out with the reading phase are called *upstream out of phase* (UPOP). The sequences located in the CDS region that are in the same reading phase of the TIS are called *CDS in phase* (CDSIP), and those that are out with the reading phase are classified as *CDS out of phase* (CDSOP), as shown in Fig. 1. The different methodologies proposed in the literature vary according to the window size, the number of nucleotides before and after the AUG codon, and the use of the UPIP, UPOP, CDSIP and CDSOP sequences. The task of computationally identifying the AUG codon depends on a method capable of accurately predicting both the positive class (TIS) and negative class (nTIS) examples.

[2] Codon is a sequence of three nucleotides that encodes a given amino acid or indicates the end point of translation.
[3] Available at http://tishunter.ucr.edu/
[4] Available at http://dnafsminer.bic.nus.edu.sg/Tis.html
[5] Available at http://www.cbs.dtu.dk/services/NetStart/

3. RELATED WORKS

In Silva et al. [2011], the authors present a methodology for the SVM-based TIS prediction problem and propose an *undersampling* method, called *M-Clus*, to address the problem of class imbalance, characteristic of this type of problem. This method consists of grouping majority class samples (nTIS) and selecting the most significant examples from each *cluster* to represent this class. In this way, the number of *clusters* considered corresponds to the number of samples available in the minority class (number of TIS sequences). The results obtained show that the proposed methodology improves the accuracy, sensitivity, specificity and adjusted accuracy metrics, with values higher than 93% for *Mus musculus* and *Rattus norvegicus* and ranged from 72.97% and 97.43% for *Arabidopsis thaliana*, *Caenorhabditis elegans*, *Drosophila melanogaster*, *Homo sapiens* and *Nasonia vitripennis*.

In their study, Pinto et al. [2017] compared the supervised and semi-supervised approaches through the *Inductive Support Vector Machine* (ISVM) and *Transductive Support Vector Machine* (TSVM), for predicting TIS. The authors use the 235, 518, 800, 1081, 1365, and 1650 nucleotide sills in two distinct scenarios. In Scenario 1 the 10-fold cross-validation method was applied, using 90% of the training base and 10% for validation. In Scenario 2, the same cross-validation method was applied, however, using 10% for model training and 90% for validation. According to the authors, the latter scenario is appropriate for transductive learning because it has fewer molecules sequenced. The windowing adopted was asymmetrical with the extraction of sequences always starting at position -9, to include the conservative positions, as presented by Kozak [1984]. The obtained results show that the TSVM method can be applied to solve the TIS prediction problem, mainly for organism with smaller number of sequences, and that the windowing with 1081 nucleotides resulted in a greater accuracy and sensitivity in the prediction, for both approaches TSVM and ISVM.

Regarding the studies that applied the cascade architecture based on SVM, we can cite the work of Garg and Gupta [2008]. The authors applied a two-layered structure in the context of prediction of virulence[6] in bacterial pathogens. The first layer is composed of SVM classifiers trained and optimized with different characteristics of the protein. The results produced by the first layer are used as input for the training of a new SVM in the second layer, which produces the final model. Employing this methodology, the authors reached an accuracy of 81.8%.

In the context of Bioinformatics, the SVM technique has offered high accuracy rates when compared to commonly used methods, such as *Random Forest* and *Linear Discriminant Analysis.* In Mazo et al. [2017], the authors analyse heart problems through images of heart cells using these three methods. The results confirm that the SVM application, specifically the Cascade SVM, can perform better when compared to the other methods. At a 98% area rate under the ROC curve (Receiver Operating Characteristic), Cascade SVM was the best performing method compared to *Random Forest* and *Linear Discriminant Analysis methods.*

In Sun and Fox [2012], the authors applied the Cascade SVM in a distributed and parallel programming model based on *MapReduce*. The authors used an iterative architecture called *Twister* which allows the calling of the *Map* and *Reduce* processes successively within a loop until a pre-set stop condition is satisfied. The experiments showed that the Cascade SVM can significantly reduce computational time, although partitioning training data in many parts does not imply a proportional reduction of training time. This is due to the cost of managing the cascading structure files, which can become more expensive than the efficiency provided by the cascade structure.

In Papadonikolakis and Bouganis [2012], the authors explored the parallelism and scalability inherent to the Cascade SVM architecture through a *Field Programmable Gate Array* (FPGA) implementation. The authors highlight the computational power of the architecture and claim that their

[6] The virulence of a bacterial pathogen is its relative ability to cause a disease, generally described in terms of the number of infecting bacteria.

implementation on a hardware like FPGA can outperform even parallel solutions using *Graphic Processor Unit* (GPU). To validate the FPGA implementation, the authors used a handwritten digits recognition base, the *MNIST Dataset*. The accuracy of the created model remained the same as the other implementations, but with a *speedup* of 25 times when compared to the implementation of the conventional SVM.

Problems related to computer vision are present in the automotive industry, in robotics and in visual recognition systems. According to Baek et al. [2015], the Cascade SVM architecture has features favourable to pedestrian detection in applications for autonomous vehicles. In this work the authors apply the architecture for real-time rejection of negative examples of pedestrian. Their results demonstrated that the Cascade SVM architecture can be applied to real-time applications in the context of computer vision. Recently, the Cascade SVM architecture got great scientific interest in contexts typically tackled with Deep Neural Networks (*DeepLearning*).

In this work, we applied the Cascade SVM architecture to the prediction problem of TIS showing their performance and limitations. We also propose a new methodology to obtain the set of training that allows to consider organisms that do not have the *upstream* region.

4. MATERIALS AND METHODS

In this section, we describe the materials and methods used in our study, including a description of the databases, the sequence extraction process, data coding, database balancing, definition of the SVM parameters, and the metrics and the validation environment.

4.1 Datasets description

The datasets used in our study correspond to those used in Pinto et al. [2017]. Data were extracted from the public database *RefSeq* [Pruitt and Maglott 2001] from NCBI[7] on 22 April 2014[8]. The data refers to the organisms *Rattus norvegicus*, *Mus musculus*, *Homo sapiens*, *Drosophila melanogaster*, *Caenorhabditis elegans* and *Arabidopsis thaliana*, representing 96.07% of all molecules available in the *RefSeq* database. The other 3.93% were the molecules disregarded due to the fact that there is little representativeness in their sequence sizes, considering a window of 1081 nucleotides, as used in this study. Therefore, the organisms *Nasonia vitripennis*, *Gallus gallus*, *Macaca mulatta*, *Pan troglodytes*, *Bos taurus*, *Capra hircus*, *Bubalus bubalis*, *Susscrofa*, *Danio rerio*, *Orcinus orca*, *Lipotes vexillifer*, *Oryctolagus cuniculus*, *Peromyscus maniculatus bairdii*, *Macaca fascicularis*, *Paniciscus*, *Gorilla gorilla*, *Callithrix jacchus*, *Chrysochloris asiatica*, *Trichechus manatus latirostris* and *Vicugna pacos* were not considered.

Table I contains the number of molecules extracted from the *RefSeq* database, for each of the considered organisms. It is important to emphasize that the molecules of the *RefSeq* database have different levels of inspection, and are classified as *Model*, *Inferred*, *Predicted*, *Provisional*, *Reviewed*, *Validated* and *WGSk*[9]. In this work, only mRNA molecules with a reviewed inspection were considered, since the molecules received a more rigorous revision process.

In addition to the datasets of the aforementioned organisms, a set of data composed of all organisms studied in this work was used, which has more than 100,000 sequences and is referenced in the course of the article as *ALL-ORG*.

[7] Available at http://www.ncbi.nlm.nih.gov

[8] The databases are available at http://icei.pucminas.br/projeto/licap2/download/cascadesvm-bio/

[9] The description of each status is available at http://www.ncbi.nlm.nih.gov/books/NBK21091/

Table I. Number of reviewed mRNA molecules by organism.

Organism	Number of molecules
Arabidopsis thaliana	35,173
Caenorhabditis elegans	26,066
Drosophila melanogaster	27,764
Homo sapiens	21,528
Mus musculus	1,097
Rattus norvegicus	1,383
Total	**113,011**

4.2 Sequence extraction

During the sequence extraction process, we only considered windows where the AUG is, at most, at the end of the CDS region. That is, the sequences whose AUG is after this region were disregarded. In this way, we ensure that all the sequences used to obtain the model have at least a portion of the CDS region, which is supposed to contain the pattern for TIS prediction we are interested in [Li and Jiang 2004].

For the extracted sequences, we performed the pre-processing step of removing duplicate sequences. Table II shows the number of sequences extracted per organism and the quantity that were duplicates.

Table II. Number of sequences extracted per organism and number of duplicate sequences,

Organism	**SIT**		**nSIT**							
			UPIP		**UPOP**		**CDSIP**		**CDSOP**	
	Unique	**Duplicated**	**Unique**	**Duplicated**	**Unique**	**Duplicated**	**Unique**	**Duplicated**	**Unique**	**Duplicated**
Arabidopsis thaliana	11,152	2,168	5,150	604	12,111	1,720	74,308	17,205	160,575	38,395
Caenorhabditis elegans	10,602	1,149	0	0	0	0	65,313	28,221	131,373	60,023
Drosophila melanogaster	9,896	6,793	12,252	6,264	26,053	13,639	81,905	109,764	123,776	177,920
Homo sapiens	9,602	3,544	6,852	1,761	16,602	5,188	65,514	48,550	125,919	91,139
Mus musculus	392	168	261	109	645	300	2,954	2,290	5,273	3,903
Rattus norvegicus	69	22	59	24	102	44	546	824	976	1,285

In preliminary tests, the UPIP and CDSIP sequences were used as negative sequences (nTIS) as input to the classifier, but the results expressed relatively lower accuracy values than the results obtained using the UPOP and CDSOP sequences as nTIS sequences. This confirms a previous observation by Li and Jiang [2004] and Nobre et al. [2007] that the UPIP sequences have a biological context very similar to the sequences containing the TIS and can thus degrade classifier performance. These sequences may even initiate the protein translation process and be stopped early by the presence of a *stop codon* [Luukkonen et al. 1995]. In the experiments, the CDSOP sequences were used to represent the negative class (nTIS) and the TIS sequences to represent the positive class in the training sets applied as input to the Cascade SVM. As observed in Guimarães et al. [2017], the CDSOP sequences may represent the negative class better than the UPOP sequences, in addition to providing training for organisms lacking the sequenced *upstream* region, as in the case of the *Caenorhabditis organism elegans*. Thus, for all AUGs found (TIS and nTIS), whether they were in the *upstream* or *downstream* region, the 1081 (*downstream*) nucleotide window was extracted starting at the AUG codon.

At the end, all nucleotides of the positive and negative sequences were converted to a 4-bit binary chain, with A, C, G, and U being encoded as: 1000, 0100, 0010, and 0001, respectively. This coding is also used in Stormo et al. [1982], Hatzigeorgiou [2002], Silva et al. [2011], Pinto et al. [2017] and Guimarães et al. [2017].

4.3 Balancing and encoding the dataset

The prediction context of TIS induces a natural imbalance in the dataset, since for each mRNA molecule there is only 1 (one) AUG codon identified as *start codon* (TIS), while all other AUG codons are identified as non-TIS (nTIS). The imbalance ratios for *Mus musculus* and *Rattus norvegicus*, for example, is 1:23 and 1:131, respectively, as observed by the authors in Silva et al. [2011]. In our dataset, the use of the CDSOP sequences for model training generated a slight imbalance, in which the number of negative sequences is higher than the number of positive sequences, being on average a ratio of 1:14 from one class to another.

Two approaches, *oversampling* and *undersampling*, are usually adopted to handle class imbalance in datasets. The *oversampling* approach consists in artificially generating minority class records in order to balance the representativeness of both. The *undersampling* technique removes instances of the majority class [Morais et al. 2016], [Liu et al. 2009], in a heuristic or random fashion, in order to reduce imbalance.

It is further noted that, due to the modification of the biological context, there are problems in both the *oversampling* and *undersampling* approaches. The first method generates artificial samples of the minority class, allowing the creation of sequences possibly inconsistent with the class, and also increases the number of sequences to be analyzed by the classifier. Similarly, the second approach may disregard majority class sequences that may be relevant to the model, but provides a smaller number of instances to be analyzed by the classifier.

As mentioned, Silva et al. [2011] proposed a heuristic method of *undersampling* called *M-Clus*, which performs the grouping of the samples contained in the majority class and selects the centroid to represent this class. In view of the results found, it is possible to observe that although the heuristic method *M-Clus* presents results slightly higher than those reached by the method of random *undersampling* (where the sequences are randomly selected), its computational cost is greater since in the random method there is no need to use clustering techniques to select the records that best represent the class.

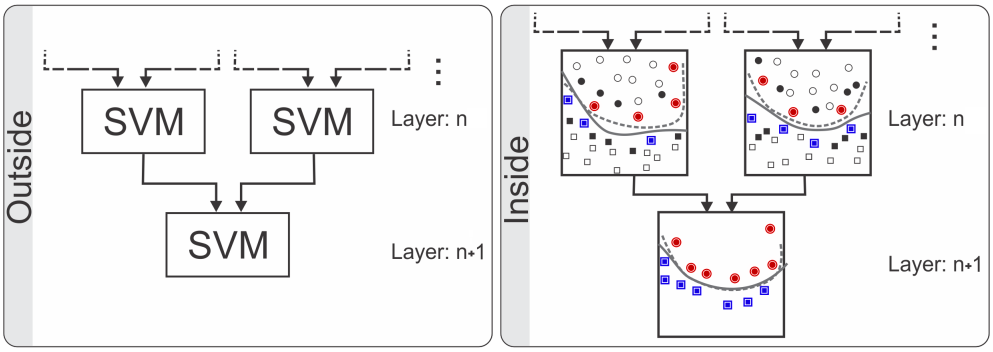
Therefore, to handle the class imbalance in the CDSOP sequences in our dataset, we chose to employ random *undersampling* in entire dataset (training and validation sets), which got satisfactory results and did not incur any considerable computational cost increase in our strategy.

4.4 Cascade SVM and parameter definition

The SVM is a machine learning (ML) technique based on the field of statistical learning [Vapnik 1995]. The SVM creates an optimal hyperplane capable of dividing the data into two classes, trying to maximize a margin as they separate. For problems that are not linearly separable, such as the TIS prediction problem, it is necessary for the constraints to solve the optimization problem, allowing some classification errors to occur. The use of a *kernel* function allows for the mapping of the training data to a characteristic separation space.

The SVM aims to separate supporting vectors from the rest of the training data, and performs this as a quadratic programming problem whose solution can reach the cubic order $O(N^3)$, regarding the number of training vectors N. In order to reduce the computational effort, one might consider parallel programming of the SVM algorithm, but that may not be feasible in this case due to the high dependency of computational calculations [Graf et al. 2004]. However, there are other strategies to accelerate the quadratic programming solution. One of them is based on "Chunking" [Boser et al. 1992], which iteratively searches for a well-defined and unrelated "Chunk" subset of support vectors. Another strategy known as "Shrinking" is used to previously identify *non-support vectors* [Joachims 1999] and save unnecessary computational effort. The strategy known as "Digesting" [Decoste and Schölkopf 2002] optimizes subsets of support vectors next to the final solution before adding new data, thus saving storage resources.

Graf et al. [2004] proposed the distributed architecture named *"Cascade Support Vector Machine"* (CSVM). As mentioned, in the CSVM the training data is initially divided into small subsets that will be used as input for several SVMs. The support vectors (SV) resulting from the first training are combined two by two, creating a single set that will be the input to a new SVM. This cascade



(a) Outside

(b) Inside

Fig. 3. Cascade SVM's Filtering Process.

process continues until only one set of SV remains. Each SVM serves as a filter, eliminating vectors that do not make up the best set of support vectors at each level ("Shrinking" strategy). This strategy is effective in decreasing the training time of SVM [Platt 1999], because it greatly reduces the number of records to be analyzed at each step.

Fig. 3 illustrates the cascading process. In the structure shown in Fig. 3, (a) represents the scheme of the externally viewed Cascade SVM, (b) represents how the cascade structure works internally. Observing the "inside" region, it is verified that the first layer is represented by two disjoint subsets and that they were selected from a training set. Each of them is passed individually as input to an SVM, which results in two subsets of support vectors (highlighted in the image). The support vectors of each SVM are combined resulting in a final classifier, represented here by the second layer of the "inside" region.

Formally we can describe the process as follows: consider the problem of classifying two classes from a training set of N examples (x_i, y_i), where $x_i \in R^d$ (d-dimension) and $y_i = \pm 1$ is the class label.

The solution developed by the SVM consists of maximizing a quadratic optimization function, expressed in its dual formulation, Equation 1.

$$L(\alpha) = \sum_{i}^{N} \alpha_i - \frac{1}{2} \times \sum_{i}^{N} \sum_{j}^{N} \alpha_i \alpha_j y_i y_j K(x_i x_j) \tag{1}$$

Respecting the following restrictions: $\forall_i\ 0 \leq \alpha_i \leq C$ and $\sum_i^N \alpha_i y_i = 0$, where α_i corresponds to the Lagrange coefficients to be determined by the optimization process. $K(\cdot) = (x_i)^T x_j$ corresponds to the matrix of *kernel* values between the patterns x_i and x_j, and C is a penalty factor imposed by the regularization scheme.

Let T denote the full training set, and Q a family of subsets of training examples $Q = \{S_1, \cdots, S_M\}$ where $S_i, S_j \subset T$ and $S_i \bigcap S_j = \{\quad\}$.

To the family Q it is possible to define the objective functions $\{L(S_i) \cdots L(S_M)\}$, Equation 1.

Since that $S_i \subset T$, it is possible to verify that $\forall\ S_i \subset T,\ L(S_i) \leq L(T)$ and as result of the optimization process of each funcition $L(S_i)$, the support vectors, $\mathrm{SV}(S_i) \subset S_i$ can be obtained.

Definition 1. A cascade corresponds to a sequence $(Q^{(t)})$ of families of subsets of T wich satisfy:

For $t = 1$,

$$Q^{(t)} = \{S_1^{(t)}, S_2^{(t)}, \cdots, S_{2^n-1}^{(t)}, S_{2^n}^{(t)}\}, n = 1, 2, 3, \cdots$$

$$P^{(t)} = \{SV(S_1^{(t)}), SV(S_2^{(t)}), \cdots, SV(S_{2^n}^{(t)})\}$$

For $t = 2, \cdots, n+1$

$$Q^{(t)} = \{p_{2i+1}^{(t-1)} \cup p_{2i+2}^{(t-1)}, i = 0, \cdots, n^{(t-1)} - 1 \quad and \quad p^{(t-1)} \in P^{(t-1)}\}$$

The solution is given by:

$$P^{(n+1)} = \{SV(Q^{n+1})\}$$

In the other words, the Cascade SVM architecture defines a sequence of families $Q^{(t)}$ in order to $L(T)$ in a finite time, i.e:

$$\exists\ t^{*}, \forall\ t > t{*}, L(Q^{(t)}) = L(T)$$

The demonstration of the convergence of this structure in given in more detail in the work of Graf el al. [2004].

The cascade architecture[10] used in this study was proposed by Wen and Lu [2004]. In this structure, illustrated in Fig. 4, unlike the structure proposed by Graf et al. [2004], there is no feedback from last to the first layer of the structure, in which the process is executed repeatedly until the stop criterion is reached.

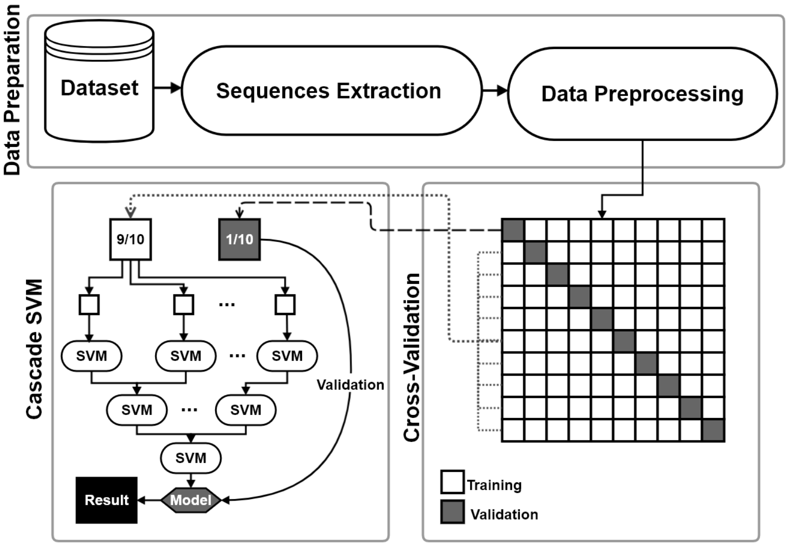


Fig. 4. Cascade SVM training flow.

To analyse the behavior of the CSVM, we proposed variations in the number of inputs and layers of the CSVM. In this proposal, experiments were performed dividing the input into k parts, considering

[10] The *code* is available at http://icei.pucminas.br/projeto/licap2/download/cascadesvm-bio/

$k = 2^n$ and $n = 1, 2, 3, 4, 5$ and 6. The number of layers used for each set of entries was $n+1$. Each k part passed as input to CSVM is constructed from the pre-processed dataset through *10-fold* cross-validation and whose randomness of data is guaranteed.

The performance of the SVM classifier depends on the proper selection of the parameters of the *kernel* function used and the smoothing parameter of the separation margin of the hyperplane, represented by the C symbol. We employed the gaussian kernel function RBF (*Radial Basis Function*), defined by Equation 2. The function requires the adjustment of the σ parameter, which represents the standard deviation in the Gaussian curve and controls the width of the curve. For simplicity, the SVM implementation used in our study employs a γ parameter, represented by $-\frac{1}{2\sigma^2}$, where σ^2 corresponds to variance.

$$K(x_i, x_j) \quad = \quad exp^{-\frac{1}{2\sigma^2}||x_i - x_j||^2} \tag{2}$$

For the adjustment of the parameters C and γ we employed the *Grid Search* algorithm[11] [Chang and Lin 2011].

In preliminary experiments, it was observed that for execution of the *Grid Search* method using the complete set of *Homo sapiens* organism sequences (approximately 20,000 sequences), it took about 360 hours to find the best pair of parameters. This computational time spent makes the methodology of using the whole set of the organism sequence unviable in this procedure. Thus, the procedure was repeated, considering only 10% of the sequence set. The results showed that the parameters C and γ were adequate to maintain satisfactory performance measures of the classifier.

The values of parameters C and γ (presented in the Table III) were determined from a sample of 10% of the dataset for the organisms with a large number of sequences. It is important to note that this procedure is a limitation of the CSVM structure. Although, this motivates future work such as proposing a parallel solution based, for example, on GPUs in order to determine the C and γ parameters. The parameters of Table III were used in each cascade structure for a given organism.

Table III. Parameters used in SVM.

Organisms	***Gamma(γ)***	**C**
Rattus norvegicus	$1.220703125 \times 10^{-4}$	8
Mus musculus	$1.220703125 \times 10^{-4}$	8
Homo sapiens	4.8828125×10^{-4}	8
Drosophila melanogaster	$3.0517578125 \times 10^{-5}$	8
Arabidopsis thaliana	$3.0517578125 \times 10^{-5}$	32
Caenorhabditis elegans	4.8828125×10^{-4}	2
ALL-ORG	$3.0517578125 \times 10^{-5}$	32

4.5 Validation and evaluation metrics

The evaluation of the results was done using the metrics of precision, sensitivity and *F-measure.*

The precision metric, described by Equation 3, evaluates among all the sequences classified as a given class, those which are truly of the class.

$$Precision \quad = \frac{TP}{TP \quad + \quad FP} \tag{3}$$

[11] Available in https://www.csie.ntu.edu.tw/˜cjlin/libsvm/.

The sensitivity measures the correctness of classification in each class, and is calculated using Equation 4.

$$Sensitivity = \frac{TP}{TP + FN} \tag{4}$$

Finally, the model can also be evaluated by the *F-measure* metric, described by Equation 5, which considers the precision and sensitivity metrics to calculate the model quality, calculating a harmonic average between them.

$$F - measure = 2 \times \frac{Precision \times Sensitivity}{Precision + Sensitivity} \tag{5}$$

being TP, TN, FP and FN, the number of *True Positive*, *True Negative*, *False Positive* and *False Negative* examples, respectively.

For the validation of the classifiers, the cross-validation technique was employed: *k-fold* [Kohavi 1995]. This process consists of dividing the total set of data into k mutually exclusive subsets, where k-1 subsets are intended for training the model and the remaining subset, k, is reserved for its validation. The error estimate is calculated by averaging the k folds to get the full effectiveness of the model. In our study, we set $k = 10$.

5. EXPERIMENTS AND RESULTS ANALYSIS

Our experiments started with the pre-processing of the databases explained in the Sections 4.1 to 4.3 with the steps of extracting, coding and balancing the sequences. After pre-processing, we searched for the best C and γ parameters for the *RBF Kernel* setting of our SVM. The parameters were applied in the CSVM structure to the TIS prediction problem, and the results obtained are presented in Table IV. The table presents the mean values of the evaluation metrics, the standard deviation, and the number of support vectors (SV) that make up the respective models.

For better understanding of the results, they are presented according to each variation of the performed experiments. Thus, the numbers 1, 2, 4, 8, 16, 32 and 64 refer to the number of subsets in which the dataset was divided, with 1 being equivalent to the conventional SVM application without cascade training.

Analysing the evaluation metrics obtained with the cascade approach, it is observed that they remained practically unchanged, regardless of the number of variations to which the dataset was submitted. It can be observed in Table IV that the precision and sensitivity metrics are high, indicating a low number of false positives and false negatives, respectively, in the obtained models. This reflects on the *F-measure* metric, which is a harmonic mean between these two measures. The best results are highlighted in bold.

These results show that for the problem addressed, dividing it into several smaller problems did not detract from the solution or the quality of the models. In contrast, in addition to maintaining the quality of the evaluation metrics, the number of support vectors of the models constructed with the cascade method was reduced, in the best case, to 9% fewer support vectors than the conventional SVM model. This occurs for datasets with the highest number of sequences as the ALL-ORG that is about five times greater than the biggest organism‘s dataset (see Table II). For organisms with smaller training sets, the reduction was 3% and 2% for *Mus musculus* and *Rattus norvegicus*, respectively.

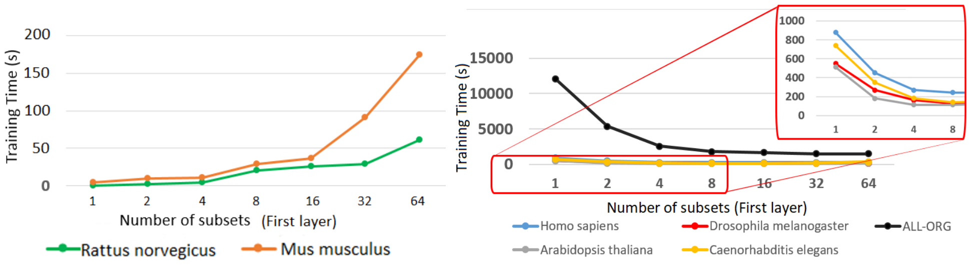
Unlike the previous results, for *Caenorhabditis elegans* organism there was an increase in support vectors. This can be explained by the following reason: as the number of subsets increases and the

Table IV. Evaluation metrics - Conventional SVM *versus* Cascade SVM.

Organism	***Precision***						
	nº Subsets						
	1	2	4	8	16	32	64
Rattus norvegicus	**95.40 ± 8.14**	**95.40 ± 8.14**	**95.40 ± 8.14**	**95.40 ± 8.14**	**95.40 ± 8.14**	**95.40 ± 8.14**	**95.40 ± 8.14**
SV of Model	107	108	105	105	105	105	105
Mus musculus	**98.29 ± 1.88**	**98.29 ± 1.88**	**98.29 ± 1.88**	**98.29 ± 1.88**	**98.29 ± 1.88**	**98.29 ± 1.88**	**98.29 ± 1.88**
SV of Model	434	423	425	429	429	422	422
Homo sapiens	98.85 ± 0.44	98.87 ± 0.45	98.87 ± 0.44	98.87 ± 0.44	**98.88 ± 0.44**	**98.88 ± 0.44**	**98.88 ± 0.44**
SV of Model	3351	3153	3100	3087	3087	3086	3083
Drosophila melanogaster	98.42 ± 0.32	**98.80 ± 0.36**	98.78 ± 0.36	97.69 ± 0.37	97.68 ± 0.36	**98.80 ± 0.36**	**98.80 ± 0.36**
SV of Model	2635	2565	2525	2535	2533	2537	2537
Arabidopsis thaliana	99.66 ± 0.17	**99.66 ± 0.16**	99.66 ± 0.19	99.63 ± 0.20	99.64 ± 0.18	99.63 ± 0.20	99.63 ± 0.20
SV of Model	1779	1680	1615	1629	1634	1632	1634
Caenorhabditis elegans	97.81 ± 0.42	97.72 ± 0.29	97.68 ± 0.27	97.69 ± 0.28	97.68 ± 0.26	97.65 ± 0.26	**97.81 ± 0.26**
SV of Model	2932	2747	2682	2668	2685	2678	3643
ALL-ORG	97.74 ± 0.13	97.76 ± 0.15	**97.81 ± 0.33**	97.73 ± 0.12	97.72 ± 0.16	97.72 ± 0.12	97.72 ± 0.13
SV of Model	6964	6591	6432	6372	6317	6311	6305

Organism	***Sensitivity***						
	nº Subsets						
	1	2	4	8	16	32	64
Rattus norvegicus	**99.33 ± 2.00**	**99.33 ± 2.00**	**99.33 ± 2.00**	**99.33 ± 2.00**	**99.33 ± 2.00**	**99.33 ± 2.00**	**99.33 ± 2.00**
Mus musculus	**99.23 ± 1.88**	**99.23 ± 1.88**	**99.23 ± 1.88**	**99.23 ± 1.88**	**99.23 ± 1.88**	**99.23 ± 1.88**	**99.23 ± 1.88**
Homo sapiens	**99.42 ± 0.36**	99.41 ± 0.36	99.38 ± 0.32	99.40 ± 0.35	99.39 ± 0.35	99.40 ± 0.35	99.41 ± 0.37
Drosophila melanogaster	99.24 ± 0.24	99.24 ± 0.20	99.25 ± 0.19	**99.26 ± 0.20**	99.25 ± 0.19	99.24 ± 0.18	99.24 ± 0.20
Arabidopsis thaliana	**99.79 ± 0.09**	99.78 ± 0.09	99.79 ± 0.11	99.78 ± 0.08	99.76 ± 0.08	99.77 ± 0.08	99.77 ± 0.08
Caenorhabditis elegans	**99.06 ± 0.28**	98.47 ± 0.37	98.43 ± 0.35	98.45 ± 0.38	98.46 ± 0.11	98.42 ± 0.31	**99.06 ± 0.28**
ALL-ORG	**98.96 ± 0.11**	98.84 ± 0.10	98.70 ± 0.14	98.66 ± 0.13	98.64 ± 0.14	98.60 ± 0.15	98.62 ± 0.19

Organism	***F-Measure***						
	nº Subsets						
	1	2	4	8	16	32	64
Rattus norvegicus	**97.14 ± 4.75**	**97.14 ± 4.75**	**97.14 ± 4.75**	**97.14 ± 4.75**	**97.14 ± 4.75**	**97.14 ± 4.75**	**97.14 ± 4.75**
Mus musculus	**98.43 ± 0.22**	**98.43 ± 0.22**	**98.43 ± 0.22**	**98.43 ± 0.22**	**98.43 ± 0.22**	**98.43 ± 0.22**	**98.43 ± 0.22**
Homo sapiens	99.13 ± 0.35	99.13 ± 0.35	99.13 ± 0.33	99.13 ± 0.34	99.13 ± 0.34	99.13 ± 0.35	**99.14 ± 0.11**
Drosophila melanogaster	98.99 ± 0.23	99.02 ± 0.23	99.02 ± 0.25	**99.03 ± 0.25**	99.02 ± 0.24	99.02 ± 0.24	99.02 ± 0.24
Arabidopsis thaliana	99.72 ± 0.10	99.72 ± 0.08	**99.73 ± 0.10**	99.72 ± 0.10	99.70 ± 0.11	99.71 ± 0.11	99.71 ± 0.11
Caenorhabditis elegans	**98.43 ± 0.22**	98.08 ± 0.17	98.05 ± 0.15	98.07 ± 0.18	98.07 ± 0.17	98.04 ± 0.11	**98.43 ± 0.22**
ALL-ORG	**98.45 ± 0.31**	98.30 ± 0.09	98.20 ± 0.09	98.19 ± 0.09	98.18 ± 0.17	98.09 ± 0.11	98.17 ± 0.13



(a) Cascade SVM Performance - Small Datasets

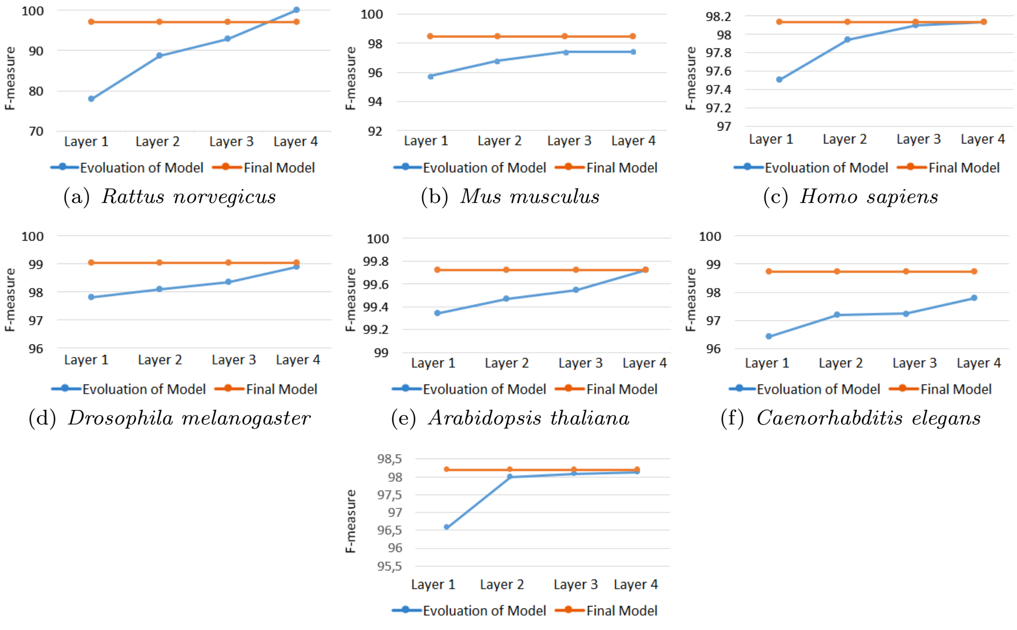
(b) Cascade SVM Performance - Large Datasets

Fig. 5. Training Time (s) *versus* Number of Subsets.

number of training vectors decreases, the representativeness in the dataset is lost. In this way, more support vectors are required for the classification model. This particular case the issue can be avoided, if the training set maintains its representativeness (by collecting more examples for the considered organism).

In Fig. 5 the computational time spent[12] is displayed for model training of each organism. Fig. 5 (a) illustrates the performance of the structure for organisms with few molecules (*Rattus norvegicus* and *Mus musculus*), while Fig. 5 (b) shows the performance of the structure of the other organisms.

[12] Experiments run on a Pentium Core i7, 32GB of RAM, using Windows 10 64 bits.



(a) *Rattus norvegicus*

(b) *Mus musculus*

(c) *Homo sapiens*

(d) *Drosophila melanogaster*

(e) *Arabidopsis thaliana*

(f) *Caenorhabditis elegans*

(g) *ALL-ORG*

Fig. 6. Improvement of the model in the cascade structure.

It is observed in Fig. 5 (a) that the cascade structure did not perform well, because the organisms have few molecules, and the cost of executing the cascade structure becomes more expensive than the benefit provided by it. This increases the time required to train the models.

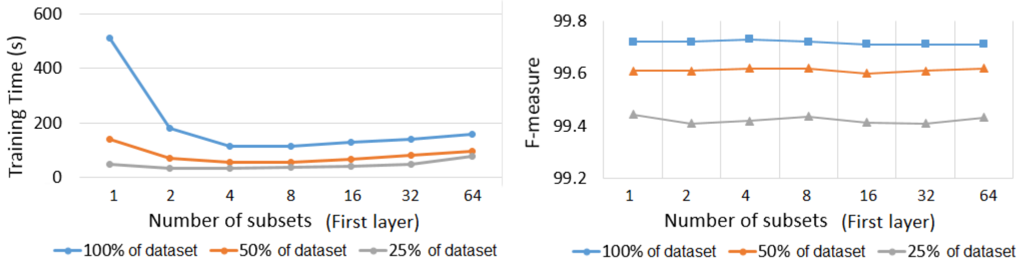
On the other hand, in Fig. 5 (b) we observe an opposite behaviour for datasets with many molecules. For these organisms, when the number of divisions (subsets) is increased, the computational time begins to reduce, because the problem is divided into smaller problems; which represents fewer records to be analyzed and consequently reduces the training time of the model. This is because the SVM reaches quadratic scales in relation to the size of the data set, as discussed previously.

The maximum reduction in training time is achieved when the Cascade SVM is with 8 subsets in the first layer, and consequently 8 SVMs being trained in parallel. In addition, this is due to the fact that the processor has only 8 physical cores and the modelling with 16, 32 or 64 *threads* presents competition for physical cores with continuous context exchanges. When the number of inputs is increased to 16, 32 and 64, the processing time begins to grow slightly, since the cost to manage the cascade algorithm begins to be greater than the efficiency provided by the CSVM structure.

It is observed that for the *ALL-ORG*, with more than 100,000 sequences, as for the other organisms with approximately 20,000 sequences presented in Fig. 5 (b), the expected behavior of the Cascade structure was maintained, there is a reduction in computational time spent to train the models gradually, so that it stabilizes approximately when using 8 SVM's in the first layer.

To better understand the behavior of Cascade SVM, we extracted the models layer-by-layer of the experiments with 8 subsets, being 8 models of the first layer, 4 models of the second, 2 of the third and 1 of the last layer. After that, each model was validated with the same validation dataset, and the results were averaged from layer to layer. Fig. 6 presents these results.

As can be observed in Fig. 6, the image is divided by organisms and presents the evolution of the model layer-by-layer (blue line) until it converges to a global optimum (orange line). In these experiments, the *F-measure* metric was used to evaluate the model. It turns out that for some organisms, the line of evolution exceeded or did not reach the global optimum line. However, it



(a) Training Time (s) - *Arabidopsis thaliana*

(b) F-measure - *Arabidopsis thaliana*

Fig. 7. Relation between the number of divisions of the dataset and the time spent in processing, considering different sizes of datasets for the *Arabidopsis thaliana* organism.

is observed that, as the layers of the structure advance, the respective models evolve, providing a refinement of the model in search of the global optimum.

As discussed previously, for the datasets of *Rattus norvegicus* and *Mus musculus*, the Cascade SVM strategy is not recommended since, with the number of sequences analyzed, the conventional SVM performed better; unlike the other analyzed organisms, in which the cascade strategy proved effective. From this behavior, the following question was raised: what is the amount of sequences necessary for the Cascade SVM strategy to perform as expected? According to the experiments carried out, our answer to this question is that it is relative, because firstly that depends on the representativeness of the sequences being analyzed, and secondly on the quantity.

To better analyze the previous question, experiments were carried out with the *Arabidopsis thaliana* organism, because it has the largest number of sequences among the organisms studied. The experiment consists of reducing the number of organism sequences by 50% and then 25% of the original number, and subjecting them to the Cascade SVM with 1 (conventional), 2, 4, 8, 16, 32 and 64 entries in the first layer. Fig. 7 (a) graphically displays the training time for each experiment and Fig. 7 (b) shows the *F-measure* metric.

It is observed that, as the number of sequences analyzed reduced, the graph changes its behavior. When we are using 100% of the dataset (blue line), approximately 20,000 sequences, the reduction of training time is evident, showing that the Cascade SVM has the expected behavior. In the experiments where 50% (orange line) of the dataset was used, the reduction was no longer as significant. On the contrary, there is a slight increase in time from 8 divisions of the database. When only 25% (green line) of the dataset was used (5,000 sequences) it is noted that there was no difference in using the Cascade SVM or the Conventional SVM, the time was almost linear, with a slight growth from 8 divisions. This shows us that, at least for the *Arabidopsis thaliana* organism, applying Cascade SVM to a dataset with less than 5,000 sequences is not recommended.

6. CONCLUSIONS AND FUTURE WORK

In this study, we sought to investigate the behavior of the Cascade SVM strategy for the prediction problem of protein translation initiation sites. The results indicate that the impacts provided by the cascade structure are mainly related to the reduction of training time and the number of support vectors of the model. The cascade method when it reaches the maximum number of physical cores of the computational structure used (8 cores in our computational experiments) with 8 subsets in the first layer yielded results with the shortest training time, reaching a reduction of 88%, as observed for the organism *Caenorhabditis elegans* and 90% for *ALL-ORG*. In addition, there was a 9% reduction in the total number of support vectors of the model.

If we compare the evaluation metrics of the conventional approach with the cascade approach, it is noticeable that their values have been roughly the same, regardless of the number of variations tested. Thus, we concluded that the classifier trained with the cascade method, in relation to the evaluation metrics, is equivalent to the conventional classifier; but it can be simpler because it can reach fewer support vectors. The reduction in training time occurred similarly for all organisms studied, even when the management cost of the cascade structure began to be more expensive than the reduction provided by the cascade structure, which happens when the number of parallel SVM exceeds the number of physical cores of the structure.

When investigating which of the considered organisms the Cascade SVM is ideal, we are faced with the results expressed in Fig. 5 (a), which shows that the performance of the cascade strategy for organisms with fewer sequences was not as good. For these organisms, conventional SVM is the most recommended. In this same line of thought / Related to this, we also observed that the Cascade SVM fails to provide a reduction in training time in the cases of organisms whose sequence numbers were less than 5,000 (for the *Arabidopsis thaliana* organism). For those cases, there was no significant gain in performance when using the Cascade SVM or the conventional SVM. Therefore it is recommended to use the conventional SVM, as it avoids implementation issues.

Although the data volume of the organisms studied does not seem large enough to use the hierarchical approach. It can be seen in Fig. 5 (b) that the behavior of the CSVM was the same for the large datasets, both for the organisms and for the dataset *ALL-ORG*, which in turn is approximately five times greater than the organism with the highest number of sequences.

In all experiments there was a reduction of approximately 90% of the computational time spent for training the models, for the data set *ALL-ORG* which spent approximately 6 hours (conventional SVM), reduced to approximately 25 minutes. This demonstrates that one can explore the potential of the CSM to deal with larger bases and of greater dimensionality, being a good strategy to reduce the computational time of the SVM.

Through understanding the behavior of the cascade strategy, we opened possibilities for the application of this strategy in transductive learning, which we believe will be of great value, since semi-supervised learning demands a significant computational consumption in its execution.

As future works to further this understanding, we suggest the investigation of Cascade SVM in other contexts of Bioinformatics, such as protein function prediction, in which the model generation time is usually high.

REFERENCES

Baek, J., Kim, J., Hyun, J., and Kim, E. New efficient speed-up scheme for cascade implementation of svm classifier. In *2015 International Joint Conference on Neural Networks (IJCNN)*. 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, pp. 1–6, 2015.

Boser, B. E., Guyon, I. M., and Vapnik, V. N. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. ACM, New York, NY, USA, pp. 144–152, 1992.

Chang, C.-C. and Lin, C.-J. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)* 2 (3): 27:1–27:27, May, 2011.

Decoste, D. and Schölkopf, B. Training invariant support vector machines. *Mach. Learn.* 46 (1-3): 161–190, Mar., 2002.

Garg, A. and Gupta, D. Virulentpred: a svm based prediction method for virulent proteins in bacterial pathogens. *BMC Bioinformatics* vol. 9, pp. 62–62, Jan, 2008.

Graf, H. P., Cosatto, E., Bottou, L., Dourdanovic, I., and Vapnik, V. Parallel support vector machines: The cascade svm. In *Advances in neural information processing systems*. Neural Information Processing Systems, NIPS 2004, Vancouver, British Columbia, Canada, pp. 521–528, 2004.

Guimarães, W. W., Pinto, C. L. N., Nobre, C. N., and Zárate, L. E. The relevance of upstream and downstream regions of mrna in the prediction of translation initiation site of the protein. In *17th IEEE International Conference on Bioinformatics and Bioengineering, BIBE 2017, Washington, DC, USA, October 23-25, 2017*. 2017 IEEE 17th International Conference on Bioinformatics and Bioengineering (BIBE), Washington, DC, USA, pp. 112–118, 2017.

Hatzigeorgiou, A. G. Translation initiation start prediction in human cdnas with high accuracy. *Bioinformatics* vol. 18, pp. 343–350, 2002.

Joachims, T. Advances in kernel methods. In *Advances in kernel methods: support vector learning*, B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.). MIT Press, Cambridge, MA, USA, Making Large-scale Support Vector Machine Learning Practical, pp. 169–184, 1999.

Kohavi, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'95. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 1137–1143, 1995.

Kozak, M. Compilation and analysis of sequences upstream from the translational start site in eukaryotic mrnas. *Nucleic Acids Res* 12 (2): 857–872, Jan, 1984.

Li, H. and Jiang, T. A class of edit kernels for svms to predict translation initiation sites in eukaryotic mrnas. In *Proceedings of the Eighth Annual International Conference on Research in Computational Molecular Biology*. RECOMB '04. ACM, New York, NY, USA, pp. 262–271, 2004.

Liu, H. and Wong, L. Data mining tools for biological sequences. *Journal of bioinformatics and computational biology* 1 (01): 139–167, 2003.

Liu, X. Y., Wu, J., and Zhou, Z. H. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 39 (2): 539–550, April, 2009.

Luukkonen, B. G., Tan, W., and Schwartz, S. Efficiency of reinitiation of translation on human immunodeficiency virus type 1 mrnas is determined by the length of the upstream open reading frame and by intercistronic distance. *J Virol* 69 (7): 4086–4094, Jul, 1995. 7769666[pmid].

Mazo, C., Alegre, E., and Trujillo, M. Classification of cardiovascular tissues using lbp based descriptors and a cascade svm. *Computer methods and programs in biomedicine* vol. 147, pp. 1–10, 2017.

Morais, R. F. A. B. D., Miranda, P. B. C., and Silva, R. M. A. A meta-learning method to select under-sampling algorithms for imbalanced data sets. In *2016 5th Brazilian Conference on Intelligent Systems (BRACIS)*. 2016 5th Brazilian Conference on Intelligent Systems (BRACIS), Recife, Brazil, pp. 385–390, 2016.

Nobre, C. N., Ortega, J., and de PÃ¡dua Braga, A. High efficiency on prediction of translation initiation site (tis) of refseq sequences. In *Advances in Bioinformatics and Computational Biology*, M.-F. Sagot and M. Walter (Eds.). Lecture Notes in Computer Science, vol. 4643. Springer Berlin Heidelberg, Angra dos Reis, Brazil, pp. 138–148, 2007.

Papadonikolakis, M. and Bouganis, C. S. Novel cascade fpga accelerator for support vector machines classification. *IEEE Transactions on Neural Networks and Learning Systems* 23 (7): 1040–1052, July, 2012.

Pedersen, A. G. and Nielsen, H. Neural network prediction of translation initiation sites in eukaryotes: Perspectives for est and genome analysis. In *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, AAAI Press, pp. 226–233, 1997.

Pinto, C. L. N., Nobre, C. N., and Zárate, L. E. Transductive learning as an alternative to translation initiation site identification. *BMC Bioinformatics* 18 (1): 81, 2017.

Platt, J. C. Advances in kernel methods. In *Advances in kernel methods*, B. Schölkopf, C. J. C. Burges, and A. J. Smola (Eds.). MIT Press, Cambridge, MA, USA, Fast Training of Support Vector Machines Using Sequential Minimal Optimization, pp. 185–208, 1999.

Pruitt, K. D. and Maglott, D. R. Refseq and locuslink: Ncbi gene-centered resources. *Nucleic Acids Res* 29 (1): 137–140, Jan, 2001.

Silva, L. M., de Souza Teixeira, F. C., Ortega, J. M., Zárate, L. E., and Nobre, C. N. Improvement in the prediction of the translation initiation site through balancing methods, inclusion of acquired knowledge and addition of features to sequences of mrna. *BMC Genomics* 12 (Suppl 4): S9–S9, Dec, 2011.

Stormo, G. D., Schneider, T. D., and Gold, L. M. Characterization of translational initiation sites in e. coli. *Nucleic Acids Research* 10 (9): 2971–2996, 1982.

Sun, Z. and Fox, G. Study on parallel svm based on mapreduce. In *In International Conference on Parallel and Distributed Processing Techniques and Applications*. Citeseer, Las Vegas, USA, pp. 16–19, 2012.

Vapnik, V. N. *The Nature of Statistical Learning Theory*. Springer-Verlag New York, Inc., New York, NY, USA, 1995.

Wen, Y.-M. and Lu, B.-L. pp. 480–486. In F.-L. Yin, J. Wang, and C. Guo (Eds.), *A Cascade Method for Reducing Training Time and the Number of Support Vectors*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 480–486, 2004.

Zhang, S., Hu, H., Jiang, T., Zhang, L., and Zeng, J. Titer: predicting translation initiation sites by deep learning. *Bioinformatics* 33 (14): i234–i242, 2017.

Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., and Müller, K.-R. Engineering support vector machine kernels that recognize translation initiation sites. *Bioinformatics* 16 (9): 799–807, Sept., 2000.

Received Month Year; revised Month Year; accepted Month Year