Final Year Project

Deep Learning for protein subcellular localisation prediction

Alessandro Paolo Baccin

Student ID: 16724489

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science

Supervisor: Gianluca Pollastri

UCD School of Computer Science

University College Dublin

May 21, 2020

Table of Contents

List of Figures

List of Tables

Glossary

1HL one hidden layer convolutional neural network. 3, 21, 24, 29

2HL two hidden layers convolutional neural network. 2, 3, 21, 24–26, 28, 29, 32

3HL three hidden layers convolutional neural network. 2, 3, 21, 23, 24, 27–29, 32, 34

4HL four hidden layers convolutional neural network. 21, 24, 29, 32

bQ Balanced Accuracy. 10, 23–25, 27, 28, 32

CNN Convolutional Neural Network. 2, 11, 14, 22

eight-CM eight-class model. 2, 3, 23, 26, 28, 32, 34

F1-score F1-score per class. 10, 27

five-CM five-class model. 2, 3, 28, 32

FPR False Positive Rate. 9

GC Generalization Coefficient. 9, 10, 12, 15

HOM Homology. 2, 15

K Cohen's Kappa score. 2, 3, 10, 32–34

macro-F1-score Macro-averaged F1 scores. 10, 32

MCC Matthews Correlation Coefficient. 9, 10, 12

MSA Multiple Sequence Alignment. 2, 3, 6, 8, 9, 11, 13–16, 20, 28, 29, 32, 34

NLS Nuclear Location Signals. 8, 16

NN Neural Network. 11, 13, 14, 19, 21

PrE Expected accuracy. 10

Prec Precision. 10, 23, 25–27, 29

PrO Observed accuracy. 10

Q Accuracy. 9–12, 15, 32

SCL sub-cellular localization. 8

Sens Sensitivity, Recall. 9, 10, 23, 25–27, 29

seven-CM seven-class model. 2, 3, 6, 17, 20, 24–29, 32, 34

six-CM six-class model. 2, 3, 17, 27, 28

SOA State of the art. 2, 9–13, 15, 16, 32

Spec Specificity. 9

Chapter 1: **Interim report differences**

New chapter

- Models
- Evaluation using Cohen's Kappa Score

Updates

- Updated Abstract
- Added new evaluation metrics in Related Work and Ideas
- Updated Data Considerations to Data considerations and processing
- Updated Outline of Approach chapter
- Removal of "Project Work-plan" chapter
- Updated Summary and Conclusion

Abstract

The understanding of proteins is of vital importance for the creation of drugs, especially knowing that such drugs need to be created at a high pace since diseases like Malaria and HIV are still very widespread. This project aims to develop a generalized seven-class computational predictor with high independence from proteins' homology information. The models this project propose are developed using a Deep N-to-1 Neural Network provided amino-acid sequences and their evolutionary information(MSA) as input. Another objective of this project is to provide insights on how different Neural architectures, different types of classification and different types of input data-sets affect the overall quality of a model's predictions. The best model this project produced is a seven-class(referred as seven-CM in this report) classifier able to predict proteins into the classes: *Cytoplasm*, *Nucleus*, *Secreted*, *Cell membrane*, *Endoplasmic reticulum membrane*, *Plastid* and *Mitochondrion*.

GitHub repo of the tools used for this proejct.

Chapter 2: **Project Specification**

Core

Build training and testing sets, train a small number of simple, shallow networks to predict sub-cellular locations of proteins and test and analyse the results.

Advanced

Repeat the experiments on a large grid of shallow and deep models and perform a thorough comparative analysis of the performances.

Chapter 3: **Introduction**

Proteins are the building blocks of life. Although there is almost a complete knowledge regarding the amino acids they are composed of, there is not much information available regarding the location in which a protein resides. The protein sub-cellular localization(SCL) which can occur in the *Nucleus*, *Cytoplasm*, *Mitochondrion* or other cellular apparatuses, is of vital importance to understand what is the protein specific function, because its location correlates to the proteins it interacts and works with. The closer two proteins are, the more likely is that they will operate together. "Aberrant Protein locations can also affect the function a protein exhibit and contributes to the pathogenesis of many human diseases; such as metabolic, cardiovascular and neurodegenerative diseases, as well as cancer" ([1]).

In some cases, such as for proteins that reside in the nucleus, localization information can be extrapolated by checking for *Nuclear Location Signals* (NLSs) in the sequence, proteins which instead reside on secretory pathways, *mitochondria* and *chloroplasts* tell their location by having N-terminal cleavable peptides. Unfortunately, the approaches revolving around this kind of information cannot be applied to all kinds of proteins since many of them have no known motifs ([2],[3]) or no N-terminal peptides [4] defining their location. In these cases, the whole sequence should still carry enough clues to be able determine the location ([2], [3], [5]).

Extracting information with this degree of complexity from a whole sequence was not a suitable task for machine learning approaches since the amount of data backed up by experimental proofs was too small to be effectively used. Nowadays, in contrast, more and more proteins are being labeled and fed into constantly increasing unified data-sets that can be used by machine learning algorithms to create models with increasing accuracy and robustness [6]. SCL Predictors may vary in the discretisation they perform and on the tools they use. Classification methods can be divided into two broad groups: the homology, or knowledge, based methods extract the possible location of a protein by comparing it with proteins for which the location is known; and *ab initio* sequence based methods, which uses evolutionary information in the form of multiple sequence alignments(MSAs) [7], not depending on similarity to other sequences of known location [6].

The classifiers this project propose use the modern *ab initio* approach. The process of building these predictors starts from the creation of finely preprocessed and redundancy reduced UniProt [8] data-set. The data-sets are preprocessed and converted into the adequate formats by using an in-house tool created for the scope of this project (or other similar projects). Once the data-sets are ready, the classifiers are built using Convolutional Neural Networks, networks of different depth are employed for the creation of such models. The resulting predictors are then evaluated using a different number of metrics; these metrics have been select by taking into consideration the imbalances the data-sets present. Due to the uniqueness (mainly for the high number of labels used) of the predictors in this project, an evaluation against other publicly available predictor is not ideal.

Chapter 4: Related Work and Ideas

Numerous predictors (SCLpred [9], BaCelLo [6], SherLoc [10], DeepLoc [11] and more others) have been developed, each having a different take on many aspects of the model construction process, such differences can span from: data-sets used varying in size, notation and information(UniProt [8] various releases), incorporation of text features from PubMed databases [10], incorporation of GO notations from Gene Ontology [12] etc.); underlying predictive architecture (SVMs [6], Neural Networks [9], combinations of various techniques [11]); training methods (5/10 folds cross-validation [9] [13], amount of redundancy reduction); and the type of classification they can carry out (3 possible classes [14],5 [9],10 [11] etc.).

As stated in the Introductory chapter, models are built following two general principles: homology or knowledge-based approach and the *ab initio* approach. These methodologies are not mutually exclusive. Some models are generated combining both concepts, in SCLpredT [15] it is shown that using MSA data paired with homology info yields improved results when compared to a data-set providing only MSA data. Additional types of information can also be used, SherLoc [10] for example couples information inferred from the protein sequence with text-based features extracted from the PubMed database. In this report, various predictors will be discussed, with a specific focus on SCLpred [9] and SClpred-EMS [13], and it will be shown that the utilization of neural networks provide SOA performances consistently, as all the SCLpred iterations ([9], [15], [14], [13]) always match with the other top of the line predictors. SCLpred models share a lot in common, from the type of reductions the data-sets undergo before being usable, to the predictive models employed for the construction of the predictor.

Models Evaluation

In each of the following paragraphs, the predictors' performances and comparisons will be shown, before doing so, the measures employed for the comparisons are given(Spec, Sens, FPR, MCC, Q, GC).

$$Spec = \frac{TN}{TN+FP} \text{ or } (100 * \frac{TN}{TN+FP})$$

$$Sens = \frac{TP}{TP+FN} \text{ or } (100 * \frac{TP}{TP+FN})$$

$$FPR = 100 * \frac{FP}{FP+TN}$$

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

$$Q = \frac{\sum_i z_{ii}}{N}$$

$$GC = \sqrt{\frac{\sum_{ij}\frac{(z_{ij}-e_{ij})^2}{e_{ij}}}{N*(K-1)}}$$

Where, TP = Number of sequences predicted in class that are observed in that class; TN = Number of sequences not predicted in a class that are not observed in that class; FP = Number of sequences predicted in that class that are not observed in that class; FN = Number of classes not predicted in a particular class that are observed to be in that class; z_{ij} = the number of sequences of class i predicted to be in class j [9]; e_{ij} = the number of sequences of class i expected to be predicted in class j by chance [9]; N = number of sequences; K = number of classes.

Generalised Coefficient (GC) is a metric used in SCLpred [9], BaCelLo [6] and other predictor's evaluation, this metric minimizes the effect of class sizes [9]. This metric is not used for the evaluation of the models used in this project.

Evaluation of this project

For the evaluation of the models produced in this project the following metrics will be used: Prec, Sens, Q (both Sens and Q already described in the chapter above), bQ, F1-score, macro-F1-score and for a final, more contextualized evaluation the Cohen's Kappa Score (K) is adopted.

$$Prec = \frac{TP}{TP+FP}$$

$$bQ = \frac{Sens+Spec}{2}$$

$$F1-score = 2 * \frac{Prec*Sens}{Prec+Sens}$$

$$macro-F1-score = \frac{\sum_i f_i}{K}$$

$$K = \frac{PrO-PrE}{1-PrE}$$

$$PrE_i = \frac{Pred*Truth}{N}$$

$$PrE_t = \frac{\sum_i PrE_i}{N}$$

Where, f_i is the F1-score obtained for label i; PrO is the observed accuracy (Q); PrE_i is the expected accuracy per class; PrE_t is the overall expected accuracy.

Precision(Prec), recall(Sens), bQ and macro-F1-score are of particular interest within the scope of this project as for the employment of heavily imbalanced data-sets, these measures provide better insights on how good and relevant predictions actually are [16] (MCC can not be utilized as these are not binary classifiers). Prec allows to understand how genuine the predictions retrieved actually are, while Sens allow to understand how many of the actual sequences the models can retrieve. Balanced Accuracy(bQ) is a useful metrics when dealing with imbalanced data-sets; in the case were two classes, one containing 95 elements and the other one 5, are getting all predicted into the biggest class, the balanced accuracy won't be of 95% but of 50%. Another metric employed is the macro-F1-score, macro F1-score is preferred over the micro (which is listed in the paragraph above as Q, as they are actually the same metric [17]), by using Macro-F1 score we attribute equal importance to all the labels [18]; macro-F1-score is mainly added for completion's of the metrics adopted.

A direct comparison is not possible, since there are no available SOA predictors which perform a similar type of classification. An evaluation of the models developed in this project will still be carried out using the Cohen's Kappa Score [19]. Cohen's Kappa score is remarkable for handling imbalanced data-sets and multi-class problems [20]. K also provides more context on the quality of the predicted measures, while F1-score is simply described as the harmonic mean between Prec and Sens, K allows to rate a classifier's predictive power against a completely random classifier, more accurately, it takes into account the actual *Observed Accuracy*(PrO the accuracy produced by the model) and *Expected Accuracy*(PrE the accuracy that would be obtained from the completely random classifier). In Lands and Koch [20] a guiding scheme for evaluations using K was provided, such scheme is: $0.21 - 0.40$ as fair, $0.41 - 0.60$ as moderate, $0.61 - 0.80$ as substantial, and $0.81 - 1$ as almost perfect. In the Evaluation Chapter it will be shown how K describes the produced models.

On the following sections, an overview on some of the already present proteins localization classifiers is made.

SCLpred [9]

SCLpred predicts four classes of sub-cellular locations for animals and fungi and five classes for plants, using a different predictor for each reign (animal, fungal and vegetal). It does so without incurring into predefined transformation and without exploiting homology information; on the contrary, this localization system compresses the sequence into a hidden feature vector [9] by relying on an N-to-1 Neural Network (such Network can also be referred to as a Convolutional Neural Network - CNN). CNNs are versatile by nature and may be applied to other kinds of problems which revolve around the task of mapping protein sequences into arrays of attributes. Such Networks can also be employed for image analysis and applications that span various areas of knowledge. This predictor uses two different data-sets for training and testing plus an additional test set (the same used in *Casadio 2008*[21]). One of the training+test sets is the one used in BaCelLo [6] and the extra decoupled test set is extracted from the latest release (2010_06) of SwissProt (now UniProt) [8]. The data-set extracted from the Swiss-Prot 2010_06 release had to undergo many steps of general data reduction and many rounds of redundancy reduction for the protein sequences, as will be explained with more context in the "Data Considerations" chapter. Redundancy reduction is a vital step as such predictors try to achieve a high degree of generalization.

Data Reduction considerations

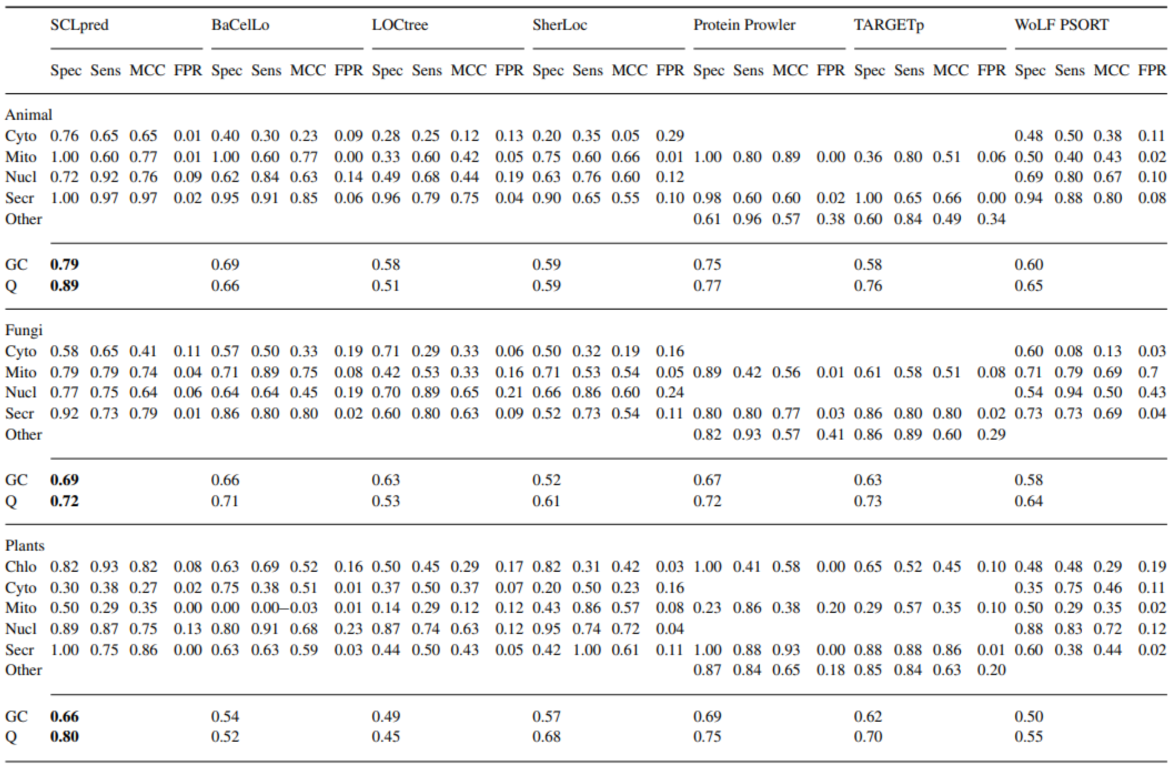
The data reduction approach used for SwissProt 2010_06 is very similar (if not entirely similar) to the approaches used in the more recent SCLpred predictors. To begin with, proteins that are not labeled with a sub-cellular localization and that do not present experimental identifiers are removed. Through this operation the biggest chunk of non-usable sequences is removed. After this, sequences are redundancy reduced against themselves using an all-against-all BLAST, which is the Basic Local Alignment Search Tool [22], an algorithm which allows the comparison of amino acid sequences with a whole database of sequences, identifying the ones with a certain degree of similarity (that can be specified by the end-user). MSAs information is extracted by running PSI-BLAST (Position Specific Iterated BLAST [22]) three times on the uniref90 [23] database, PSI-BLAST is able to extrapolate evolutionary information of sequences by confronting specific positions of the sequence with specific positions of other sequences. SCLpred is trained using k-fold cross validation, with k = 10, meaning that for each one of the 10 total folds, a tenth of the fold is reserved for testing, another tenth for validation and the remaining is reserved for training.

Predictive core: N1-NN Neural Network

This Neural Network is the pulsing heart that fuels all the members of the SCLpred family, in SCLpred [9] the "basic" (non-Deep) version of the Neural Network is used. This is composed of only a 2-layered convolutional layer followed by an average pooling and a fully connected 2-layered Neural Network. The convolutional layer's job is to map input motifs (each motif is the size of 41 residues, as the upper bound for most signal peptides and other signals is 40) into a sequence of feature vectors by learning a non-linear function. In SCLpred [9] network configuration, this translates to having a cascade of replicated 2-layered NNs, each learning the non-linear function from a sequence window. As each Network in the first layer produces a feature vector, there is the need of combining all this information into one unique feature vector that the fully connected NN can use to make its prediction. Such feature vector is obtained by using a Pooling unit which sums all the feature vectors and normalize the result using a normalization constant K. A more in-depth description will be given out in the Outline of Approach chapter. The actual classification is carried out by the final fully connected NN using the feature vector extracted.

SCLpred evaluation

Compared to BaCelLo [6], which already surpassed contemporary SOA predictors, SCLpred performs better for animals and fungi, also regarding the accuracy (Q), while for plants the results are the same in terms of Q, but SCLpred still returns a better Generalization Coefficient (GC). When compared instead to a pool of other SOA predictors, SCLpred still defends itself very well, ranging from second to third place for all the 3 kingdoms.



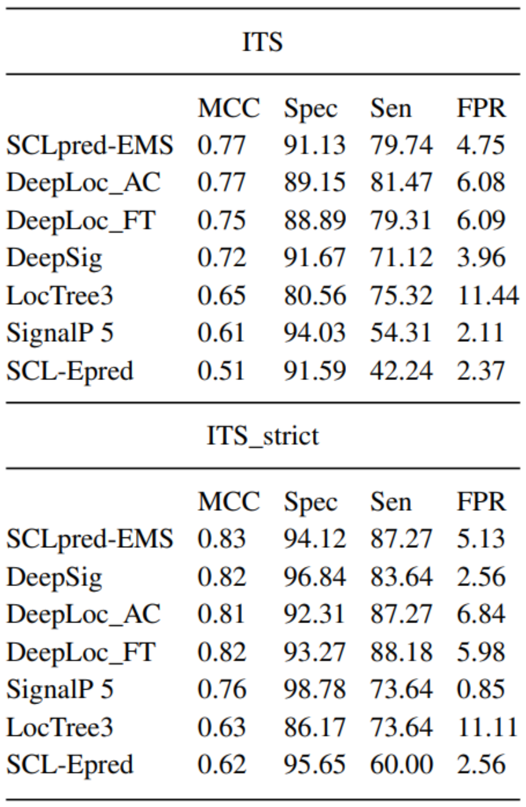
	SCLpred				BaCelLo				LOCtree				SherLoc				Protein Prowler				TARGETp				WoLF PSORT			
	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR	Spec	Sens	MCC	FPR
Animal																												
Cyto	0.76	0.65	0.65	0.01	0.40	0.30	0.23	0.09	0.28	0.25	0.12	0.13	0.20	0.35	0.05	0.29									0.48	0.50	0.38	0.11
Mito	1.00	0.60	0.77	0.01	1.00	0.60	0.77	0.00	0.33	0.60	0.42	0.05	0.75	0.60	0.66	0.01	1.00	0.80	0.89	0.00	0.36	0.80	0.51	0.06	0.50	0.40	0.43	0.02
Nucl	0.72	0.92	0.76	0.09	0.62	0.84	0.63	0.14	0.49	0.68	0.44	0.19	0.63	0.76	0.60	0.12									0.69	0.80	0.67	0.10
Secr	1.00	0.97	0.97	0.02	0.95	0.91	0.85	0.06	0.96	0.79	0.75	0.04	0.90	0.65	0.55	0.10	0.98	0.60	0.60	0.02	1.00	0.65	0.66	0.00	0.94	0.88	0.80	0.08
Other																	0.61	0.96	0.57	0.38	0.60	0.84	0.49	0.34				
GC	**0.79**				0.69				0.58				0.59				0.75				0.58				0.60			
Q	**0.89**				0.66				0.51				0.59				0.77				0.76				0.65			
Fungi																												
Cyto	0.58	0.65	0.41	0.11	0.57	0.50	0.33	0.19	0.71	0.29	0.33	0.06	0.50	0.32	0.19	0.16									0.60	0.08	0.13	0.03
Mito	0.79	0.79	0.74	0.04	0.71	0.89	0.75	0.08	0.42	0.53	0.33	0.16	0.71	0.53	0.54	0.05	0.89	0.42	0.56	0.01	0.61	0.58	0.51	0.08	0.71	0.79	0.69	0.7
Nucl	0.77	0.75	0.64	0.06	0.64	0.64	0.45	0.19	0.70	0.89	0.65	0.21	0.66	0.86	0.60	0.24									0.54	0.94	0.50	0.43
Secr	0.92	0.73	0.79	0.01	0.86	0.80	0.80	0.02	0.60	0.80	0.63	0.09	0.52	0.73	0.54	0.11	0.80	0.80	0.77	0.03	0.86	0.80	0.80	0.02	0.73	0.73	0.69	0.04
Other																	0.82	0.93	0.57	0.41	0.86	0.89	0.60	0.29				
GC	**0.69**				0.66				0.63				0.52				0.67				0.63				0.58			
Q	**0.72**				0.71				0.53				0.61				0.72				0.73				0.64			
Plants																												
Chlo	0.82	0.93	0.82	0.08	0.63	0.69	0.52	0.16	0.50	0.45	0.29	0.17	0.82	0.31	0.42	0.03	1.00	0.41	0.58	0.00	0.65	0.52	0.45	0.10	0.48	0.48	0.29	0.19
Cyto	0.30	0.38	0.27	0.02	0.75	0.38	0.51	0.01	0.37	0.50	0.37	0.07	0.20	0.50	0.23	0.16									0.35	0.75	0.46	0.11
Mito	0.50	0.29	0.35	0.00	0.00	0.00	–0.03	0.01	0.14	0.29	0.12	0.12	0.43	0.86	0.57	0.08	0.23	0.86	0.38	0.20	0.29	0.57	0.35	0.10	0.50	0.29	0.35	0.02
Nucl	0.89	0.87	0.75	0.13	0.80	0.91	0.68	0.23	0.87	0.74	0.63	0.12	0.95	0.74	0.72	0.04									0.88	0.83	0.72	0.12
Secr	1.00	0.75	0.86	0.00	0.63	0.63	0.59	0.03	0.44	0.50	0.43	0.05	0.42	1.00	0.61	0.11	1.00	0.88	0.93	0.00	0.88	0.88	0.86	0.01	0.60	0.38	0.44	0.02
Other																	0.87	0.84	0.65	0.18	0.85	0.84	0.63	0.20				
GC	**0.66**				0.54				0.49				0.57				0.69				0.62				0.50			
Q	**0.80**				0.52				0.45				0.68				0.75				0.70				0.55			

Figure 4.1: From [9]. SCLpred comparison against other SOA predictors.

SCLpred-EMS [13]

The Data reduction process is very similar to the one employed in SCLpred [9]. The Convolutional Network used is also the same but with an added convolutional layer (making it deep), therefore, employing two 2-layered convolutional layers. In this model, the second layer convolutes on the features extracted from the first layer, capturing even more abstract nuances from the sequence. The output feature vector from the second convolutional layer is again average pooled and fed into the fully connected network which performs the prediction. SCLpred-EMS uses k-fold cross-validation, but with k$= 5$. What SCLpred-EMS does more than SCLpred is to create an even more heavily redundancy reduced training and test sets, as the two test sets are created for comparison against other predictors. Another difference between SCLpred and SCLpred-EMS is the size of the input motifs, which is reduced to 21; this value was chosen based on experimental result, as for motifs' size above 21 performances started to slowly degrade. The resulting model tested on the two heavily redundancy reduced sets (namely ITS and ITS_strict) is the result of an ensemble average of the best 5 models obtained during 5-fold cross-validation (similar approach already used in SCLpred[9])

By running comparisons with other SOA predictors, namely DeepLoc-1.0 [11], DeepSig [24], LocTree3 [25] and SignalP 5.0 [26], and basing the evaluation on MCC, SCLpred-EMS is only matched by the accurate version of DeepLoc for the non-strict data-set. While it performs better when compared on the strict set, the higher MCC might be caused by the small size of the data-set and other factors [13].



ITS

	MCC	Spec	Sen	FPR
SCLpred-EMS	0.77	91.13	79.74	4.75
DeepLoc_AC	0.77	89.15	81.47	6.08
DeepLoc_FT	0.75	88.89	79.31	6.09
DeepSig	0.72	91.67	71.12	3.96
LocTree3	0.65	80.56	75.32	11.44
SignalP 5	0.61	94.03	54.31	2.11
SCL-Epred	0.51	91.59	42.24	2.37

ITS_strict

	MCC	Spec	Sen	FPR
SCLpred-EMS	0.83	94.12	87.27	5.13
DeepSig	0.82	96.84	83.64	2.56
DeepLoc_AC	0.81	92.31	87.27	6.84
DeepLoc_FT	0.82	93.27	88.18	5.98
SignalP 5	0.76	98.78	73.64	0.85
LocTree3	0.63	86.17	73.64	11.11
SCL-Epred	0.62	95.65	60.00	2.56

Figure 4.2: From [13]. SCLpred-EMS evaluation against other modern SOA predictors.

SCLpredT[15]

SCLpredT proposes to expand the data-set by including more information other than the MSAs and sequence itself. Thanks to this new approach SCLpredT predicts one more class of protein sequences than SCLpred does. This new class is referred to as the *Organelle* class. The input data-set was generated from the Gene Ontology [12] data-set that provides annotations for more than 13 million proteins, such increment in the available data is also what made possible the introduction of the new class. It also must be stated that the *Secreted* class previously present in SCLpred [9] is now substituted with the *Extracellular* class, as most secreted proteins performs their tasks outside the boundaries of the cell [15]. Again, a redundancy reduction process analogue to the one used in SCLpred [9] had to be employed.

One of SCLpredT goals was to discover whether better results could be achieved from the introduction of Homology information in the form of Localization Annotations of similar sequences, or from the usage of secondary structure information.

To answer such questions, three different data-sets were created so that results could be compared. The first data-set provided only MSA information, the second incorporated MSA information coupled with secondary structure information and the third data-set had MSA and Homology information. Of the 3 data-sets, the data-set comprehending Homology and MSA information is the one that showed relevant improvements, especially in the animals and fungi kingdom. Increased accuracy of predictions in the plant's kingdom was also noticed, but with less significance. A similarity threshold by which homology information provided significant improvements was identified, for animals and fungi, proteins which had 50% more similarity with the best template carrying the location label improved the accuracy significantly; for plants this threshold was at 60% [15]. The second data-set, including secondary structure information did not provide any benefit, in many cases it even performed worse than the only MSA set. This might be caused by a bias in the training from the structural information, which could have prevented the NN from learning more subtle clues and nuances about the sequences. This hypothesis is backed up by the significantly shorter training times using such set. To support the additional homology information, the NN had to be expanded with one more Network which learned a different non-linear function N_h^2 from the homology information.

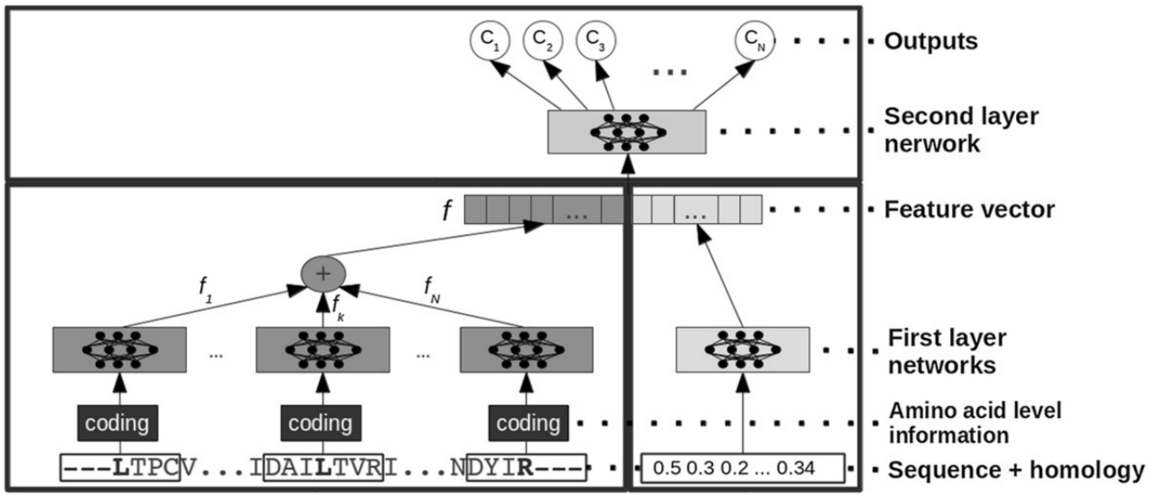
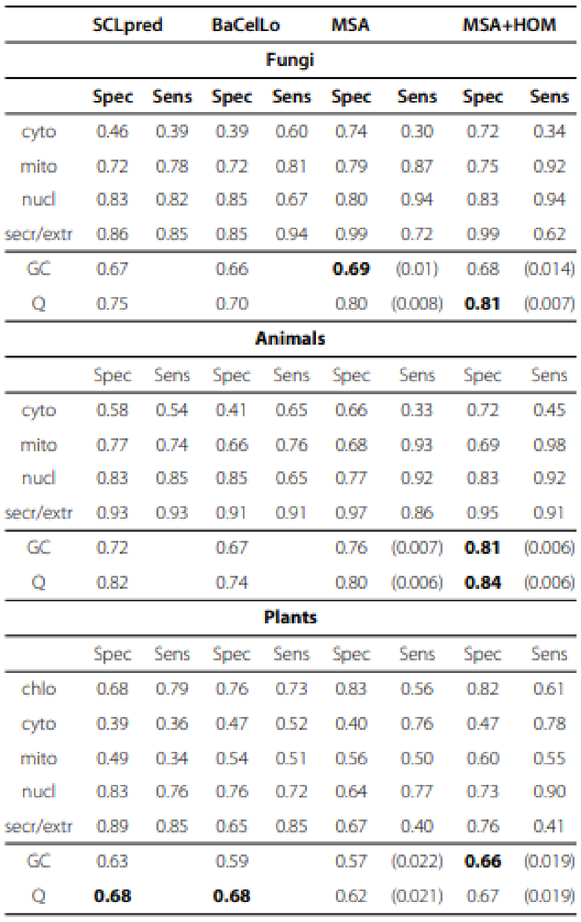


Figure 4.3: From [15]. A graphical representation of the CNN employed in SCLpredT highlighting the extra neural network employed for the extraction of features from the homology data.

Redundancy reduction and other data reduction tasks were also performed on the data-sets used for SCLpredT [15] with the same fashion as in SCLpred [9] and SCLpred-EMS [13]. MSAs were also extracted and embodied on the sequence following the same methodology.



	SCLpred		BaCelLo		MSA		MSA+HOM	
				Fungi				
	Spec	**Sens**	**Spec**	**Sens**	**Spec**	**Sens**	**Spec**	**Sens**
cyto	0.46	0.39	0.39	0.60	0.74	0.30	0.72	0.34
mito	0.72	0.78	0.72	0.81	0.79	0.87	0.75	0.92
nucl	0.83	0.82	0.85	0.67	0.80	0.94	0.83	0.94
secr/extr	0.86	0.85	0.85	0.94	0.99	0.72	0.99	0.62
GC	0.67		0.66		**0.69**	(0.01)	0.68	(0.014)
Q	0.75		0.70		0.80	(0.008)	**0.81**	(0.007)
				Animals				
	Spec	Sens	Spec	Sens	Spec	Sens	Spec	Sens
cyto	0.58	0.54	0.41	0.65	0.66	0.33	0.72	0.45
mito	0.77	0.74	0.66	0.76	0.68	0.93	0.69	0.98
nucl	0.83	0.85	0.85	0.65	0.77	0.92	0.83	0.92
secr/extr	0.93	0.93	0.91	0.91	0.97	0.86	0.95	0.91
GC	0.72		0.67		0.76	(0.007)	**0.81**	(0.006)
Q	0.82		0.74		0.80	(0.006)	**0.84**	(0.006)
				Plants				
	Spec	Sens	Spec	Sens	Spec	Sens	Spec	Sens
chlo	0.68	0.79	0.76	0.73	0.83	0.56	0.82	0.61
cyto	0.39	0.36	0.47	0.52	0.40	0.76	0.47	0.78
mito	0.49	0.34	0.54	0.51	0.56	0.50	0.60	0.55
nucl	0.83	0.76	0.76	0.72	0.64	0.77	0.73	0.90
secr/extr	0.89	0.85	0.65	0.85	0.67	0.40	0.76	0.41
GC	0.63		0.59		0.57	(0.022)	**0.66**	(0.019)
Q	**0.68**		**0.68**		0.62	(0.021)	0.67	(0.019)

Figure 4.4: From[15]. SCLpredT evaluation of normal MSA and MSA+HOM dataset against SCLpred [9] and BaCelLo [6].

In this comparison with the SCLpred [9] and BaCelLo [6] predictors, although omitting the *organelle* class, which is not predicted by either BaCelLo or SCLpred, a clear improvement in the GC and Q is obtained, even when using the data-set containing only MSA information.

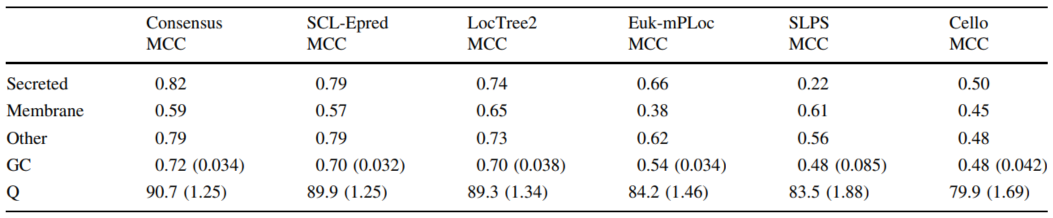
SCL-Epred [14]

The last member of the SCLpred family that will be discussed is SCL-Epred. This predictor performs a novel type of classification (which will be picked up and carried on by SCLpred-EMS [13]). A protein can be classified as a membrane protein, extracellular/secreted or *other*. This type of classification is particularly required for drug discovery [14], as drug targets are more likely to be *extracellular* than *intracellular* [27].

SCL-Epred is trained and tested using 10-fold cross-validation on a set generated from SwissProt Release 2011_02, its independent test was performed using instead a heavily redundancy reduced set obtained from SwissProt Relase 2012_07.

The input encoding and MSAs alignments are also generated similarly to how they are generated for the other SCLpred predictors. The N1-NN Neural Network employed in SCL-Epred is also the same employed for SCLpred and SCLpredT (for the MSA only data-set).

Compared on the 3-classes classification to other SOA predictors (LocTree2[28], Euk-mPLoc[29], SLPS[30], Cello[31]) and with Consensus [14], a meta predictor composed by: LocTree2, SignalP and SCL-Epred itself, SCL-Epred display egregious performances, confirming the validity of the SCLpred approach for designing predictors.



	Consensus MCC	SCL-Epred MCC	LocTree2 MCC	Euk-mPLoc MCC	SLPS MCC	Cello MCC
Secreted	0.82	0.79	0.74	0.66	0.22	0.50
Membrane	0.59	0.57	0.65	0.38	0.61	0.45
Other	0.79	0.79	0.73	0.62	0.56	0.48
GC	0.72 (0.034)	0.70 (0.032)	0.70 (0.038)	0.54 (0.034)	0.48 (0.085)	0.48 (0.042)
Q	90.7 (1.25)	89.9 (1.25)	89.3 (1.34)	84.2 (1.46)	83.5 (1.88)	79.9 (1.69)

Figure 4.5: From [14]. A comparison of the SCLEpred predictor on the 3 class classification against other SOA predictors.

BaCelLo [6]

The BaCelLo predictive system shares many similarities with SCLpred. First, the classification they carry out is the same (*Nucleus*, *Cytoplasm*, *Secreted*, *Mitochondrion*, *Chloroplast* - only for plants) and second, they both belong to the *de novo/ab initio* class of predictors. BaCelLo identifies that a generalized predictor cannot only rely on the presence of N-terminal peptides or C-terminal peptides and NLSs as indicators for a protein's location as such information might not available on all proteins, and even proteins which belong to those locations might not always carry such motifs within their sequence. BaCelLo combines intelligence from the sequences itself, MSAs and the additional capability to check for the presence of motifs like the ones already mentioned. BaCelLo predictive architecture can be abstracted as a waterfall of SVMs [6].

During the development of this predictor, another problem has been identified. The amount of protein data for each sub-cellular region is uneven, this is also why predictors are often, but not in all cases (DeepLoc [11]), limited to a small number of classes. This discrepancy in the amount of data backing up each class can create a bias when introduced into a training set. As a workaround for such bias, BaCelLo predictive architecture tends to favorise smaller classes when performing predictions. This capability is achieved by manipulating the distances in the hyper-plane where data-points are discretised by the SVMs [6].

Chapter 5: Data Considerations and processing

The main data-set used for this project is derived from a .tab file containing $190'000$ protein sequences. The .tab was obtained from UniProt, "The Universal Protein Resource is a comprehensive resource for protein sequence and annotation data" [8]. UniProt provides a large database of up-to-date protein sequences whose experimental and predicted data is cured by experts from all around the globe. The sheer amount and quality of labelled proteins made UniProt the perfect candidate. The data-set included proteins from the animal, fungi and vegetable kingdoms; using one unique predictor for all three kingdoms is one of the key differences between this project's models and most of the models already available. For comparisons, a data-set including proteins from only the animal and fungi kingdom was also created and used; this data-set was employed in the creation a six-class predictor (six-CM), this classifier carries out the same classification as seven-CM, excluding *Plastids*, which do not appear in animal and fungal cells. Because the initial .tab file did not include accurate information about the organisms proteins belonged to (it did not include the Organism ID, which is particularly useful to filter out proteins from different kingdoms) a different, smaller data-set was retrieved from the UniProt website.

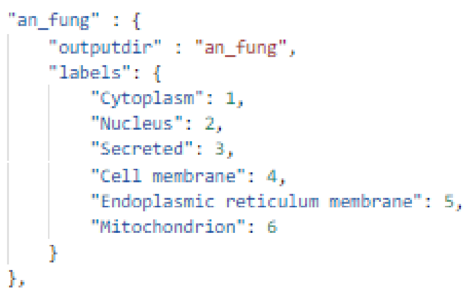
As mentioned above, the data-set the project starts with is a structured .tab file containing most of the information necessary, apart from the Organism ID. For each protein, the list of the attributes is the following: Entry, Entry name, Status, Protein names, Gene names, Organism (lengthy textual description of the organism), Length, Sequence, Sub-cellular location, Taxonomic Lineage.

Data preprocessing with Python

Almost all of the data preprocessing necessary was carried out with a custom Python tool, named fypreprocessing. The explanation of the "fypreprocessing" tool will be now used as a road-map to explain all the necessary reprocessing steps and what characteristics the data-set presented.

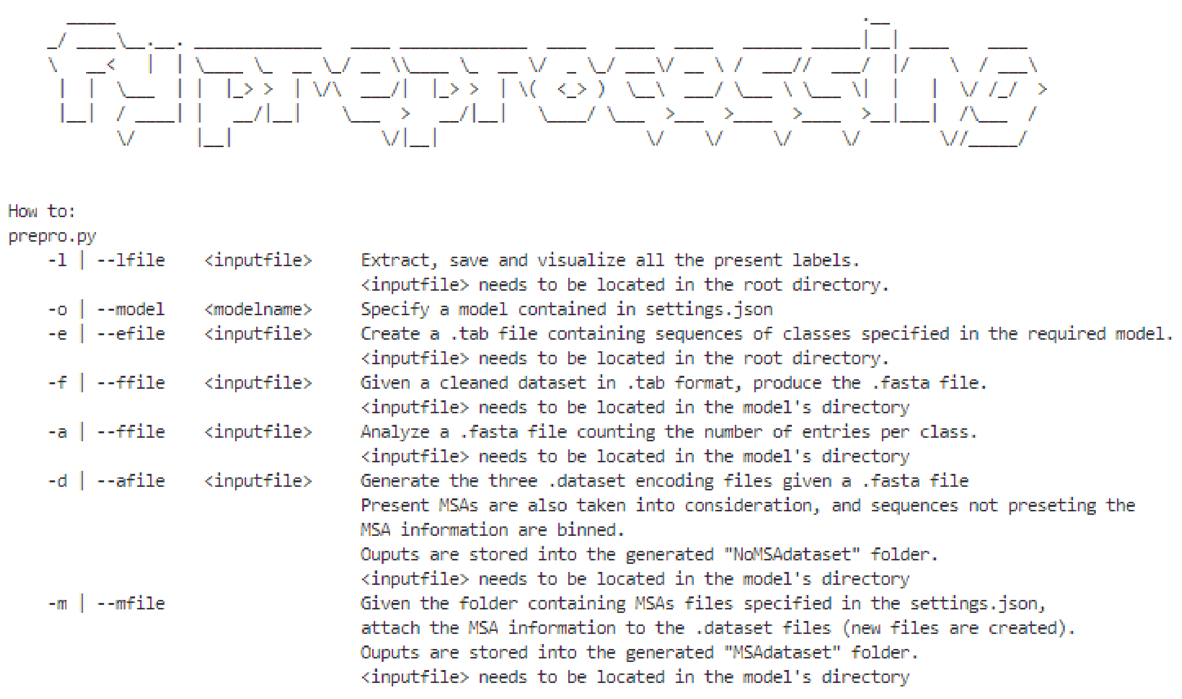
Data cleaning

The tool require the inclusion of a *settings.json* file to carry out the preprocessing steps, this file contains the meta-information necessary for each model.



```
"an_fung" : {
    "outputdir" : "an_fung",
    "labels": {
        "Cytoplasm": 1,
        "Nucleus": 2,
        "Secreted": 3,
        "Cell membrane": 4,
        "Endoplasmic reticulum membrane": 5,
        "Mitochondrion": 6
    }
},
```

Figure 5.1: Example of an entry appearing in *settings.json*, each entry specifies the directory in which all the generated files are to be saved, another required information is the classes to extract and their encoding numbers.



```
   ____                                                                          .__
 _/ ____\___.__.______________   ____________   ____   ____   ____   ______ ______|__| ____    ____
 \   __<   |  |\____ \_  __ \_/ __ \____ \_  __ \/  _ \_/ ___\/ __ \ /  ___//  ___/  |/    \  / ___\
  |  |  \___  ||  |_> >  | \/\  ___/|  |_> >  | \(  <_> )  \__\  ___/ \___ \ \___ \|  |   |  \/ /_/  >
  |__|  / ____||   __/|__|    \___  >   __/|__|    \____/ \___  >___  >____  >____  >__|___|  /\___  /
        \/     |__|               \/|__|                       \/    \/     \/     \/        \//_____/

How to:
prepro.py
    -l | --lfile    <inputfile>     Extract, save and visualize all the present labels.
                                    <inputfile> needs to be located in the root directory.
    -o | --model    <modelname>     Specify a model contained in settings.json
    -e | --efile    <inputfile>     Create a .tab file containing sequences of classes specified in the required model.
                                    <inputfile> needs to be located in the root directory.
    -f | --ffile    <inputfile>     Given a cleaned dataset in .tab format, produce the .fasta file.
                                    <inputfile> needs to be located in the model's directory
    -a | --ffile    <inputfile>     Analyze a .fasta file counting the number of entries per class.
                                    <inputfile> needs to be located in the model's directory
    -d | --afile    <inputfile>     Generate the three .dataset encoding files given a .fasta file
                                    Present MSAs are also taken into consideration, and sequences not preseting the
                                    MSA information are binned.
                                    Ouputs are stored into the generated "NoMSAdataset" folder.
                                    <inputfile> needs to be located in the model's directory
    -m | --mfile                    Given the folder containing MSAs files specified in the settings.json,
                                    attach the MSA information to the .dataset files (new files are created).
                                    Ouputs are stored into the generated "MSAdataset" folder.
                                    <inputfile> needs to be located in the model's directory
```

Figure 5.2: "fypreprocessing" –help command output.

The first command proposed is the –lfile (-l) command, which is used to display all the labels contained in the given .tab file. By using the –lfile command on the UniProt data-set it is showed that the top ten most numerous classes are, in decreasing order: *Cytoplasm* (7248 sequences), *Nucleus* (6059), *Secreted* (3838), *Cell membrane* (2806), *Mitochondrion* (1118), *Plastid* (1094), *Endoplasmic reticulum membrane* (1084), *Cell projection* (457), *Membrane* (439) and *Mitochondrion inner membrane* (415). The top seven labels plus the *other* label (grouping all the proteins that do not belong to any of the other 7 classes) are chosen, the sheer class size is an important factor when choosing which classes to work with, this because when applying redundancy reduction the entire data-sets suffers high numerosity reduction. It is interesting to notice that by listing the most frequent labels in the second smaller data-sets (animal and fungal cells only), the ranking of the labels is almost identical.

The models are defined in the *settings.json*, the .tab file is cleaned accordingly to model specified using the –model argument. The actual cleaning of the data-set is carried out using the –efile command; given a model this command will create a cleaned .tab file containing only the sequences of classes specified in the model meta-information. The cleaning process consist of: removal of sequences with missing information, removal of sequences exceeding the length of $10'000$ and sequences smaller than 30 sequences, another important cleaning step is removal of the sequences with no "eco:0000269" [11] identifier present, this identifier ensures that the localization information has been experimentally verified. The number of columns is also reduced, keeping only: Entry, Sequence, Length, Sub-cellular location [CC].

By cleaning and selecting only the seven classes of interest the data-set size goes from $190'000$ entries to $23'244$ (this number vary with the classes selected). The removal of sequences that are not experimentally verified is a particularly important step, this is necessary because some of the sequences (which do not present "eco:0000269") are tagged by using information retrieved from other predictors, so it is preferred to avoid predicting data using predicted data [13].

.fasta encoding

When dealing with such high amount of protein sequences, it is important to consider the presence of redundancy among them. Having heavily redundant data can introduce a bias in the model produced; this project on the other hand predict different types of proteins from the extrapolated attributes and patterns in the sequences, not by using homology between them. In data-sets like the one used by Sherloc2 [32], homology reduction is done by using an 80% cutoff [11], similarly to the data-set used in SCLpred, where homology reduction also utilizes an 80% threshold[13], these thresholds define the amount of similarity two sequences are allowed to have, so no sequence is more than 20% similar to any other sequence within the data-set. The data-sets used for this project, both the main and the animal-fungal are redundancy reduced using BLAST with the same threshold used in SCLpred(80%). Homology reduction is performed on a website that allows for cluster analysis of data: cd-hit suite [33]. In order for the data-set to be crunched by these two website a .fasta file needs to be produced from the raw set; "*FASTA format is a text-based format for representing either nucleotide sequences or peptide sequences, in which base pairs or amino acids are represented using single-letter codes. A sequence in FASTA format begins with a single-line description, followed by lines of sequence data. The description line is distinguished from the sequence data by a greater-than (">") symbol in the first column. It is recommended that all lines of text be shorter than 80 characters in length*" [34].

The .fasta encoding corresponds to the –dfile command of the Python tool. By analyzing the .fasta file returned by the cd-hit suite using the –afile command, it can be seen that the number of protein sequences decreased from roughly 23 thousands to $10'804$. The ranking of the labels remains untouched but it is interesting to notice how all the classes have been almost halved, with the exception of the "Cell Membrane" class which got reduced by $\approx 70\%$.

.dataset encoding

After homology reduction is performed on the data-set, the .fasta file has to be encoded into a custom file-type, a .dataset file, which can be fed into the Neural Network. The Neural Network requires a protein sequence to have its residues encoded into 22 real valued numbers. This step is carried out using the **fypreprocessing** tool –dfile command, this command also splits the data-set into three different files, a train file, a validation file and a test file. The sequences are distributed unevenly across the three files, with train.dataset containing $\approx 80\%$. of the entries, and test.dataset and validation.dataset containing the remaining $\approx 10\%$ each, for consistency, a static seed is used, so that the splits on the same data-sets are consistent. After this step, the data is ready to be fed into the NN.

	Seven-class model	Eight-class model (other)	Six-class model (no veg.)	Five-class model (EMS)
Original dataset	190'193	190'193	44'124	190'193
Cleaning & classes extraction	23'244	29'111	10'525	23'244
BLAST on cd-hit	10'804	12'764	6'079	10'804
No-MSA and MSA				
train.dataset	8'188	10'023	3'982	8'188
test.dataset	1'020	1'248	499	1'020
validation.dataset	1'046	1'267	515	1'046

Table 5.1: Table showing showing all the data-sets used and their sizes after each transformation.

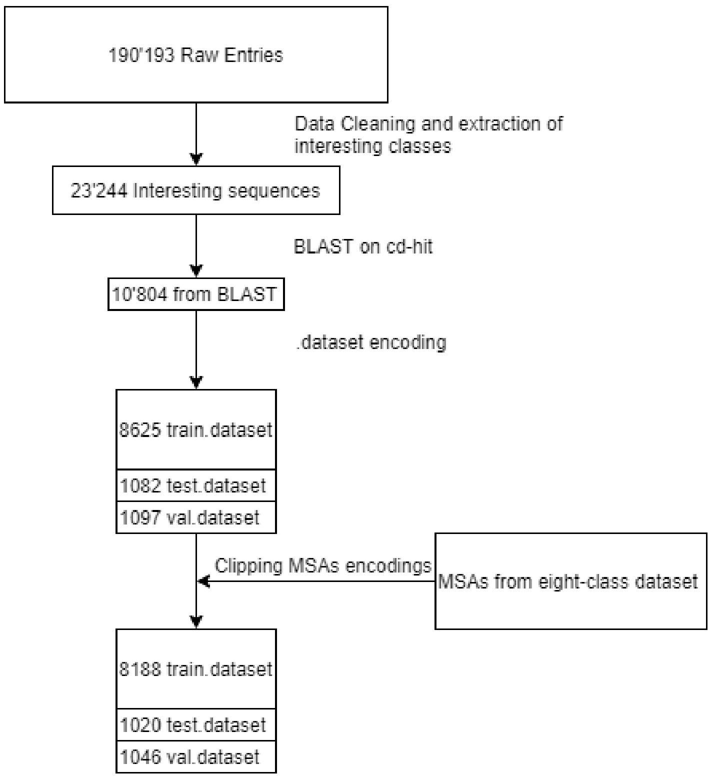


Figure 5.3: Diagram summarizing all the preprocessing steps applied for (seven-CM).

After the .dataset encoding is done, MSA information needs to be embodied in the sequence, this is why the 22 number encoding is used. The 22 numbers represent the frequency at which each residue in the alignment can occur in that column position, with the frequency of the original clipped to 1, all the other frequencies depend on the residues appearing in that position in related sequences. The amount of identity that two sequences need to have in order for this process to be compared is arbitrary. The algorithm used to generate the MSA information is the PSI-BLAST [22] algorithm. The generation of the alignments has been carried out by this project's supervisor, Prof. G. Pollastri using the .fasta file after the redundancy reduction setup. Such alignments are then clipped to the sequences present in all the three .dataset files. It is important to state that due to the fact that the alignments have been generated with the original data-set including the *other* class and with a redundancy reduction of 70%, there are discrepancies between MSA and non-MSA data-sets. Because of the amount of time the MSA generation requires (roughly one week), it was opted to mirror both data-sets by removing sequences from the Non-MSA data-sets (instead of generating new MSAs) which did not have a corresponding MSA equivalent, around 500 sequences were lost because of this.

Chapter 6: **Outline of Approach**

The approach taken for this project highly follows the approach taken in SCLpred-EMS [13] for the initial steps. Divergences are necessary due to the different scope the projects have. The starting configuration is a NN with two 2-layered convolutional layers followed by an average pooling and a 2-layered fully connected network will be used. Other configurations with one, three and four 2-layered convolutional layers are used, configurations with different sizes of hidden feature vectors were also attempted.

The Deep N1-NN used is composed of three layers, the first two layers are the convolutional layers which extract the features, these two convolutional layers are then followed by an average pooling unit which feeds the obtained feature vector into the fully connected neural network which perform the actual mapping of the feature vector to the output class.

The main architecture used (2HL, two 2-layered convolutional layers followed by an average pooling etc.) will be described in the chapter below. The other architectures used (1HL, 3HL, 4HL) have an identical structure, the only difference is in the number of hidden convolutional layers employed and the sizes of the hidden feature vectors.

Convolutional Layers

Here is where the first convolution of the whole amino acid sequence occurs, the result is an intermediate state vector representing a compression of the info carried by the sequence. This vector is obtained by applying a non-linear function I to various windows of residues: $is^i = I(ic_i)$, where $ic_i = (i - c, ..., i, ..., i + c)$ correspond to the motif window at position i. A motif size of 22 is used. In practice, the function I translates to a 2-layered NN which is replicated for each window of residues.

A similar process occurs in the second convolutional layer, where instead of the input motif the intermediate non-linear function H^k where k corresponds to the layer ($k = 1$ in this case), is applied to a windows of intermediate states at position j, $hc_j^1 = (j - \gamma, ..., j, ..., j + \gamma)$, the resulting feature vector is obtained as $hs_i^1 = H^1(hc_j^1)$. The squashing functions used in the convolution layer is hyperbolic tangent.

Each of the output vectors at position p (hs_p^1) obtained from the last convolutional layer are then fed into a pooling unit, averaging them element-wise and producing the vector v.

Output Layer

In this layer, the feature vector v is mapped into a single property, in this case, the sub-cellular localization [9]. The prediction is carried out by applying again the non-linear function O learned by the network to the vector v, making $class = O(v)$. The activation function used by the fully connected neural network is a $Softmax$ function.

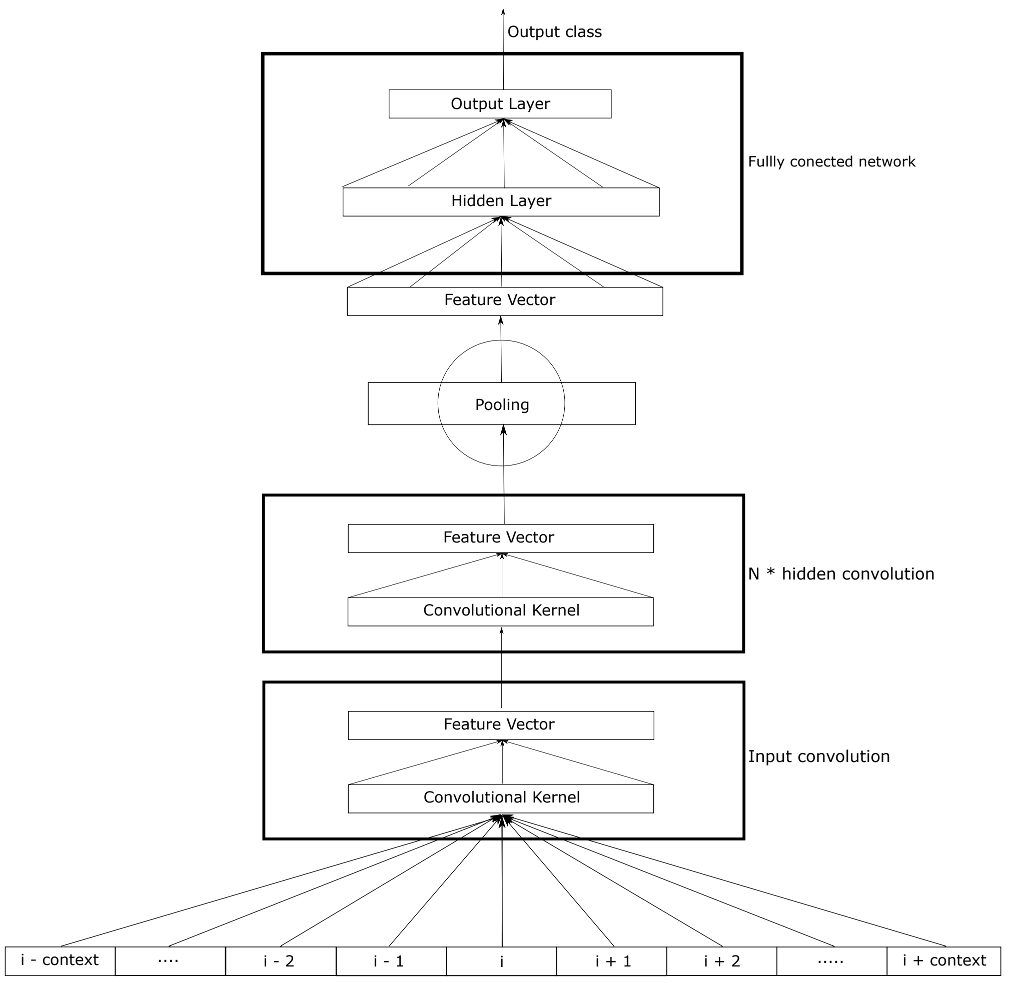


Figure 6.1: From [13]. A graphical depiction of the CNN employed that employed for this project.

General Considerations

The whole neural network is a feed forward network trained by gradient descent on the error. The number of hidden units in the first convolutional layer, the number of hidden units in the second convolutional layer and the number of hidden units in the feature to output network are kept to 10, although some of the models have also been tested hidden units sizes of 20 and 15. The value 10 proved to be optimal, and the modification did not show any significant improvements. The number of epochs used in the training phase started as 1000, this number was then decreased to 800 as the convergence to the *minima* occurred between epoch 400 to 600 depending on the model, 800 would give enough room to all models to reach the *minima* and have plenty of epochs to do some refinements. The learning rate used across all the models is 0.1, again this other hyper-parameter did show any significant improvement to the overall predictions, lower learning rates did only slow the convergence to the *minima*. The number of free parameter ranged from slightly above 150 to almost 13000 for the most numerous one, although in theory it is supposed that one hidden layer should be sufficient to solve most of the problems using a CNN, it is observed that in practical cases, a higher number of layers (this does scale infinitely) is beneficial to the classifier's performances. It will be shown in the next chapter in what cases increasing the number of hidden layers is actually beneficial to the predictions and how beneficial this is.

Chapter 7: **Models**

For this project, a total of 13 models were generated. These models differ in the architecture used, type of classification and given data-sets.

As already mentioned, the type of classification affects the quality of predictions; the early adoption of the *other* class is an example of this. From the early training sessions of the eight-CM, it became clear that such class was preventing the model from classifying a specific group of labels correctly, this was probably caused by the heterogeneity of sequences within it. Inside the class *other*, proteins appertaining to some of the following classes were present: *Cell projection*, *Membrane*, *Mitochondrion inner membrane*, *Golgi apparatus membrane*, *Cell junction* and *Golgi apparatus*. The problem with these classes (and more similar ones that have not been listed) is that most of them are a sub-classes of some of the other classes or belong to the same cellular system; *Golgi apparatus membrane* and *Golgi apparatus* are both part of the *Endomembrane system* together with the *Endoplasmic Reticulum membrane*, *Secreted* and *Cell membrane* classes [35]. Because of this, the error during training was not decreasing as the model would place such sequences into more suitable classes.

Seven-class model

This type of classification has been tested on different networks of different depth, with and without MSA. The best bQ for this classifier is obtained using a deep model of three hidden convolutional layers. The total number of free parameters for this architecture (3HL) was around 4200. The bQ obtained with this architecture is $\approx 59\%$. This architecture provided an improvement of $\approx 7\%$ on bQ compared to the models employing only one or two hidden convolutional layers. The bQ obtained with these two architectures was, respectively: 52.8% and 52%. It is interesting to notice how beneficial the employment of the three hidden convolutional for the smaller classes is. If compared to the architecture with two hidden convolutional layers, the Prec for the classes *Endoplastic reticulum membrane*, *Plastid* and *Mithochondrion* improve from: 0.56, 0.58, 0.33 to 0.62, 0.59 and 0.48. Sens also see an analogue improvement, improving from 0.29, 0.52 and 0.12 to 0.48, 0.54 and 0.44 (Table 7.2). By looking at the confusion matrix in 7.1, it can be noted how the majority of the miss-predicted sequences of the groups *Nucleus* and *Cytoplasm* are, for their majority, miss-predicted between the two classes. This occurs at a distinctly high rate, 96 out of the 347 proteins in the *Cytoplasm* class are predicted into *Nucleus*, and, at an actually higher rate, 119 out of the 285 *Nucleus* proteins are actually predicted into *Cytoplasm*. This erroneous trend appear in all of the other models at a similar rate. Another similar trend is the miss-classification of *Cell membrane* sequences into the *Endoplastic reticulum membrane* class. This trend does not appears in reverse order, but it is also shared across all the trained models; similarly to this, 16 out of the 48 *Mitochondrion* sequences are predicted into the *Cytoplasm* class. This seems to be also reoccurring across all the predictors (Table 7.2).

	Seven-class model			Eight-class model (other)			Six-class model (no veg.)			Five-class model (EMS)		
	Train	*Test*	*Validation*	*Train*	*Test*	*Validation*	*Train*	*Test*	*Validation*	*Train*	*Test*	*Validation*
Cytoplasm	2'680	347	325	2'684	299	313	1'504	198	189	2'680	347	325
Nucleus	2'494	285	324	2'432	335	316	1'396	149	169	2'494	285	324
Secreted	1'040	139	141	1'041	130	133	189	27	27	**2'124**	**290**	**284**
Cell membrane	681	86	92	650	78	93	385	62	54	**2'124**	**290**	**284**
Endoplasmic r. m.	403	65	51	388	48	62	249	33	38	**2'124**	**290**	**284**
Plastid	438	50	55	432	52	51				438	50	55
Mitochondrion	452	48	58	440	46	55	259	30	38	452	48	58
Other				1'956	260	244						

Table 7.1: Table showing the distribution of classes among the models produced.

		1 Hidden Conv. Layer				2 Hidden Conv. Layer				3 Hidden Conv. Layer			
	Supp	*pred*	*Prec*	*Sens*	*F1-Score*	*Prec*	*Prec*	*Sens*	*F1-Score*	*pred*	*Prec*	*Sens*	*F1-Score*
Cytoplasm	347	418	0.56	0.67	0.61	477	0.55	0.76	0.64	374	0.58	0.64	0.61
Nucleus	285	266	0.60	0.56	0.58	207	0.62	0.45	0.52	291	0.59	0.56	0.58
Secreted	139	155	0.79	0.88	0.83	148	0.82	0.88	0.85	150	0.83	0.88	0.85
Cell membrane	86	94	0.59	0.64	0.61	91	0.58	0.62	0.60	82	0.64	0.59	0.61
Endoplasmic r. m.	65	33	0.55	0.28	0.37	34	0.56	0.29	0.38	40	0.62	0.48	0.54
Plastid	50	31	0.81	0.50	0.62	45	0.58	0.52	0.55	36	0.59	0.54	0.56
Mitochondrion	48	23	0.35	0.17	0.23	18	0.33	0.12	0.18	47	0.48	0.44	0.46
Accuracy (Q)		0.61				0.60				0.62			
Balanced Accuracy (bQ)		0.53				0.52				0.59			
Macro-Averaged F1-Score		0.55				0.53				0.60			

Table 7.2: Table showing metrics for each of the models employed for the seven-CM classification.

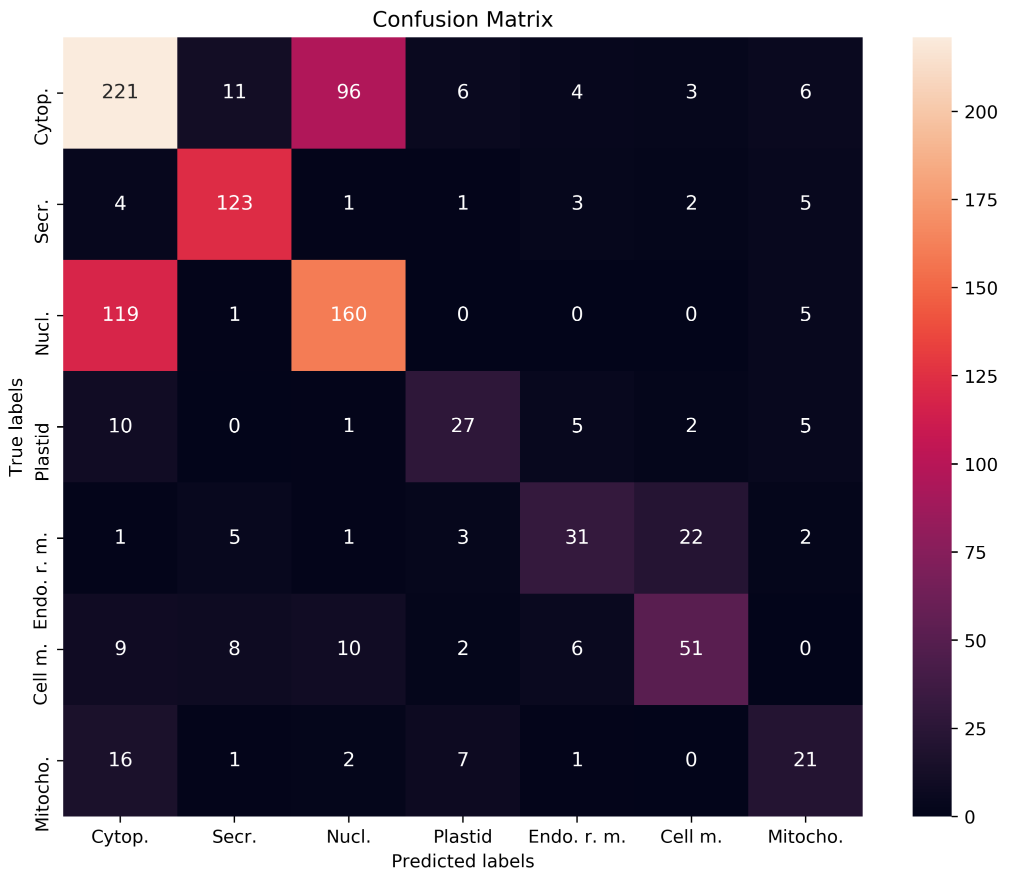


Figure 7.1: Confusion matrix for the Seven-class model (seven-CM) trained on the 3-hidden layers 3HL architecture with "standard" window size of 10.

As the three hidden layers architecture was successful, a four-layered architecture was also tested. The results for this architecture proved to be much worse than the standards set by the 1HL, 2HL and 3HL models. The 4HL architecture yield a bQ of 20.9% with the complete inability to classify any of the classes correctly. This is probably caused by the gradient being "diluted" too much in the back-propagation across the four layers (Table 7.3). Other architectures with different sizes of hidden feature vectors (a 3 hidden layers model with feature vector's sizes of 20 and a 2 hidden layers model with feature vector's sizes of 20 and 15) were tested; results did not show any notable improvements (nor degradation).

		2 HL hidden f. v. of size 20 and 15				3 HL hidden f. v. of size 20				4 Hidden Conv. Layer			
	Supp	*pred*	*Prec*	*Sens*	*F1-score*	*pred*	*Prec*	*Sens*	*F1-score*	*pred*	*Prec*	*Sens*	*F1-score*
Cytoplasm	347	411	**0.55**	**0.65**	**0.60**	354	**0.58**	**0.60**	**0.59**	909	**0.36**	**0.95**	**0.52**
Nucleus	285	272	**0.59**	**0.56**	**0.58**	298	**0.58**	**0.61**	**0.59**	0.	**0.**	**0.**	**0.**
Secreted	139	131	**0.83**	**0.90**	**0.86**	149	**0.84**	**0.90**	**0.87**	111	**0.65**	**0.52**	**0.58**
Cell membrane	86	69	**0.61**	**0.49**	**0.54**	78	**0.56**	**0.51**	**0.54**	0.	**0.**	**0.**	**0.**
Endoplasmic r. m.	65	61	**0.54**	**0.51**	**0.52**	49	**0.57**	**0.43**	**0.49**	0.	**0.**	**0.**	**0.**
Plastid	50	22	**0.82**	**0.36**	**0.50**	41	**0.73**	**0.60**	**0.66**	0.	**0.**	**0.**	**0.**
Mitochondrion	48	34	**0.32**	**0.23**	**0.27**	51	**0.35**	**0.38**	**0.36**	0.	**0.**	**0.**	**0.**
Accuracy (Q)		**0.60**				**0.61**				**0.39**			
Balanced Accuracy (bQ)		**0.53**				**0.57**				**0.21**			
Macro-Averaged F1-Score		**0.55**				**0.59**				**0.16**			

Table 7.3: Table showing metrics for each of the remaining models employed for the seven-CM classification.

Eight-class model

This was the first model attempted for this project. The bQ obtained with this model is much worse than the bQ obtained with any of the other models. This classifier is completely unable to classify *Endoplastic reticulum membrane* sequences, apart from the reason also cited in this chapter's introduction, another cause of this might be the small class size (with only 48 sequences in the test set). This model was trained on a 2HL architecture, for a different number of epochs and learning rates. None of the different approaches improved the obtained results and without sequence alignments the models performed worse on all the remaining classes. The bQ achieved with this model is 40%, Surprisingly, high Sens values on the *Nucleus* (0.73) and *Secreted* (0.79) classes are obtained; the high Sens for the *Nucleus* is quite a feat, no other proposed model achieves such Sens rate. Prec for the *Secreted* label (0.75) is also promising, but this seems to be in line with the trend of the other predictors. High measures for both Prec and Sens seems "easy" to achieve. As stated in the introduction of this chapter, this model failed particularly in the classification of the *Other* class. To better visualize this, it is particularly useful to take a look at the confusion matrix for this model in figure 7.2. By looking at the matrix, it can be seen how the row and column for class *Other* are distributed across most of the other classes, most of the classes are predicted as *Other* and a lot of the actual *Other* sequences are distributed across the labels; the most numerous sequences that are wrongfully predicted to be not in *Other* are proteins from *Cytoplasm* and *Nucleus*. Another interesting aspect is that most of the classes's distribution is that the majority of the "Endoplastic reticulum membrane" sequences are erroneously predicted into the "Other" label.

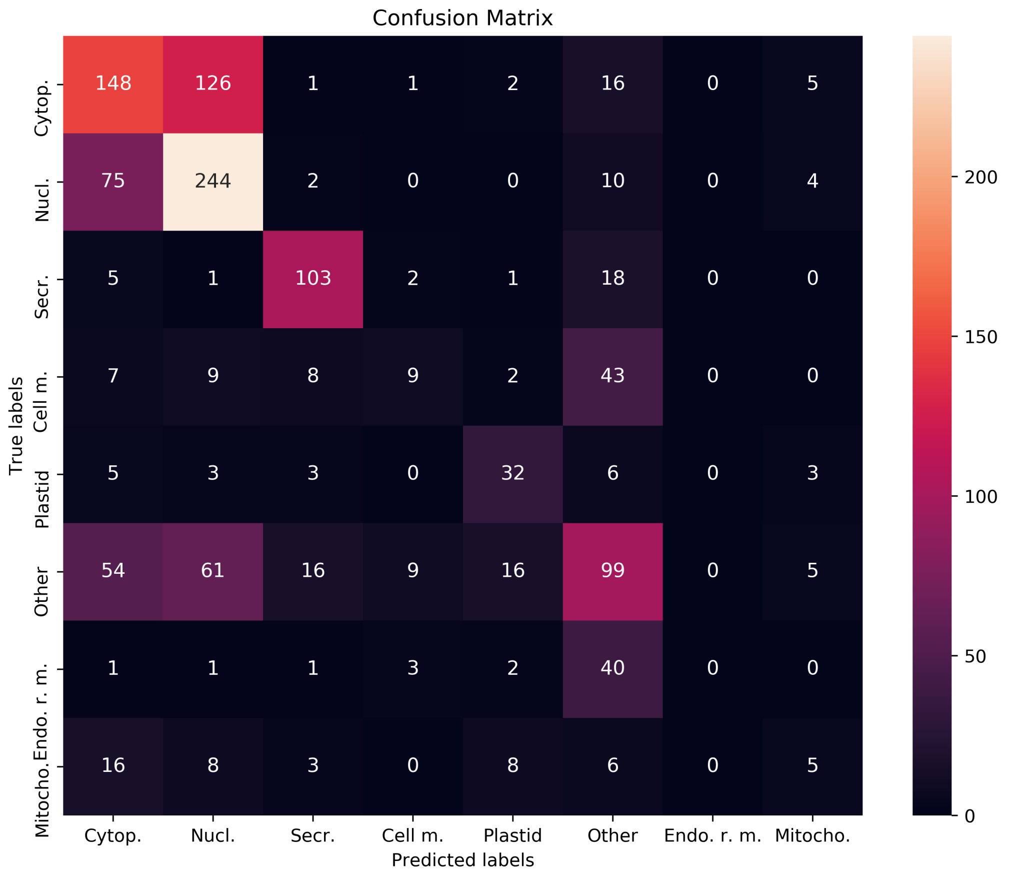


Figure 7.2: Confusion matrix for the Eight-class (eight-CM) model.

Six-class model (animals and fungi)

The scope of this model is to provide insights on whether having an only animal and fungal data-set would improve the obtained results. The data-set employed is smaller than the data-set employed in all the other models, and because of this, the obtained results need to be put into context. The distribution of proteins among all the classes is still very similar to the distribution in the other data-sets used. This model was trained on a 2HL. The Prec and Sens for the classes that seemed to be problematic on the seven-CM (*Endoplastic reticulum membrane* and *Mitochondrion*, *Plastid* is not present) have quite improved. However, for the two classes just cited the values obtained for Prec are 0.56 and 0.54 while the values obtained for Sens are 0.58 and 0.5. The same type of error already cited is also happening within this classifier, with the miss-classification between *Nucleus* and *Cytoplasm* as well as the misplacement of 11 out of the 31 Mitochondrial sequences (Table 7.3).

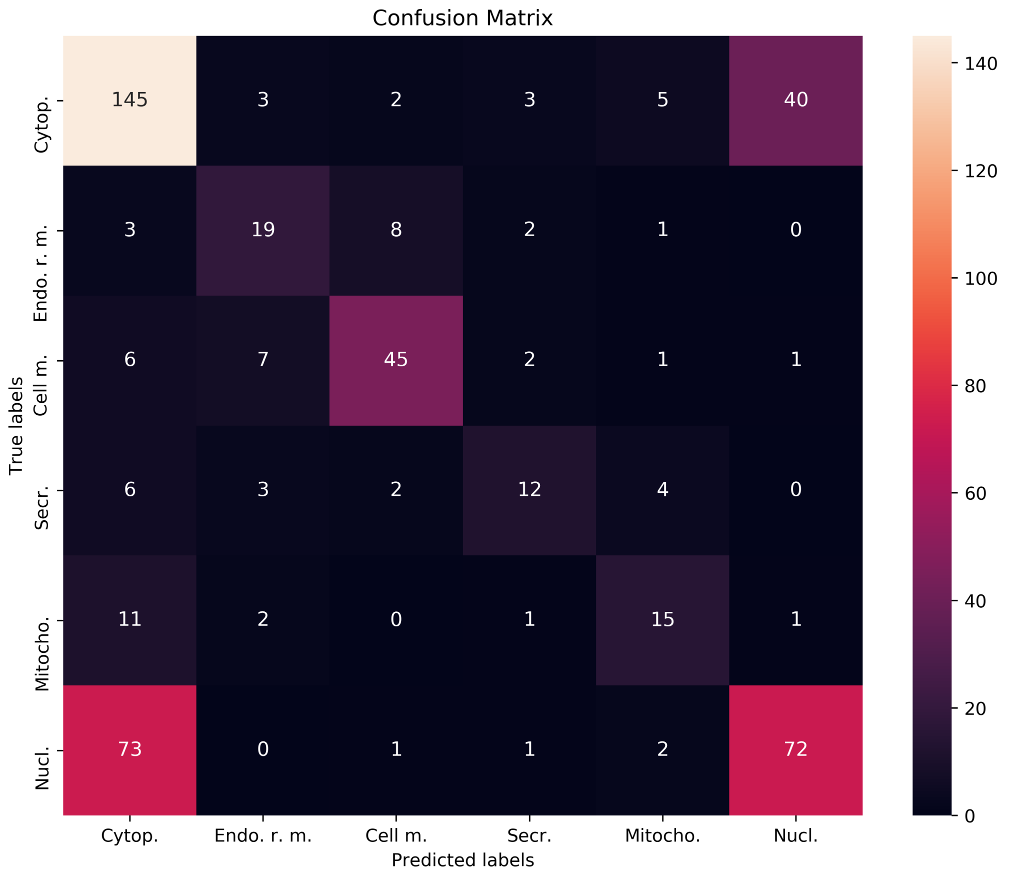


Figure 7.3: Confusion matrix for the Six-class (six-CM) model.

Five-class model (EMS)

This model provide a classification more similar to the one provided in the SCLpred-EMS predictor [13]. All the sequences of the *EMS* system are being grouped into one single class. The *EMS* class for this model contains the protein sequences from the *Secreted*, *Cell membrane* and *Endoplastic reticulum membrane* classes. By having these proteins grouped together, in their actual biological class, the model generated presents a bQ in par with the best seven-CM obtained (using the 3HL architecture). The Prec and Sens of the predictions on the EMS class is 0.87 (for both), while the Sens for the problematic *Mitochondrion* is actually 0.1: worse when compared to the same value obtained in the seven-CM 3HL model. This model was also tested on the 3HL architecture, as this architecture showed improved results with seven-CM. For this model the 3HL architecture did not give any improvements - it actually decreased the balanced accuracy score of 1 percentage point. By looking at Table 7.4, displaying the Confusion Matrix for this model, we can see how much more accurate predictions are for the *EMS* class, with a Prec, Sens and F1-score of 0.87. Compared to the best seven-CM, there si an improvement from 0.07 to 0.10 on this three measures if we compared then to the average of the measures obtained for the three separate classes in seven-CM 3HL. Apart from this improvement, the same trend of miss-classification between *Nucleus* and *Cytoplasm* is observed, as well as the erroneous classification of *Mitochondrion* into *Cytoplasm*, which actually happens at a greater magnitude, with half of the Mitochondrial sequences being classified as *Cytoplasmic*. This is also reflected in the recall (Sens) of this class, which is 0.12

		5-class model 2HL				5-class model 3HL				6-class model (hum + fung) 2HL					8-class model 2HL				
	Supp	*pred*	*Prec*	*Sens*	*F1*	*pred*	*Prec*	*Sens*	*F1*	*Supp*	*pred*	*Prec*	*Sens*	*F1*	*Supp*	*pred*	*Prec*	*Sens*	*F1*
Cytoplasm	347	197	**0.56**	**0.57**	**0.56**	214	**0.57**	**0.62**	**0.59**	198	145	**0.59**	**0.73**	**0.66**	299	148	**0.48**	**0.49**	**0.49**
Nucleus	285	176	**0.57**	**0.62**	**0.59**	166	**0.56**	**0.58**	**0.57**	149	72	**0.63**	**0.48**	**0.55**	335	244	**0.54**	**0.73**	**0.62**
Secreted	*290*	*253*	***0.87***	***0.87***	***0.87***	*258*	***0.91***	***0.89***	***0.90***	27	12	**0.57**	**0.44**	**0.50**	130	103	**0.75**	**0.79**	**0.77**
Cell membrane	*290*	*253*	***0.87***	***0.87***	***0.87***	*258*	***0.91***	***0.89***	***0.90***	62	45	**0.78**	**0.73**	**0.75**	78	9	**0.38**	**0.12**	**0.18**
Endoplasmic r. m.	*290*	*253*	***0.87***	***0.87***	***0.87***	*258*	***0.91***	***0.89***	***0.90***	33	19	**0.56**	**0.58**	**0.57**	48	0	**0.**	**0.**	**0.**
Plastid	50	29	**0.85**	**0.58**	**0.69**	33	**0.70**	**0.66**	**0.68**						52	32	**0.51**	**0.62**	**0.56**
Mitochondrion	48	15	**0.47**	**0.31**	**0.38**	7	**0.35**	**0.15**	**0.21**	30	15	**0.54**	**0.50**	**0.52**	46	5	**0.23**	**0.11**	**0.15**
Other															260	99	**0.42**	**0.38**	**0.40**
Accuracy (Q)		**0.66**				**0.66**				**0.62**					**0.51**				
Balanced Accuracy (bQ)		**0.59**				**0.58**				**0.58**					**0.40**				
Macro-Averaged F1-Score		**0.62**				**0.59**				**0.59**					**0.39**				

Table 7.4: Table summarizing all the metrics for the five-CM, six-CM and eight-CM.

lower than the recall obtained in seven-CM 3HL. Also, this is even more accentuated in the 3HL model used for this type of classification, where the amount of correctly labeled "Mitochondrion" sequences dropped from fifteen to seven.

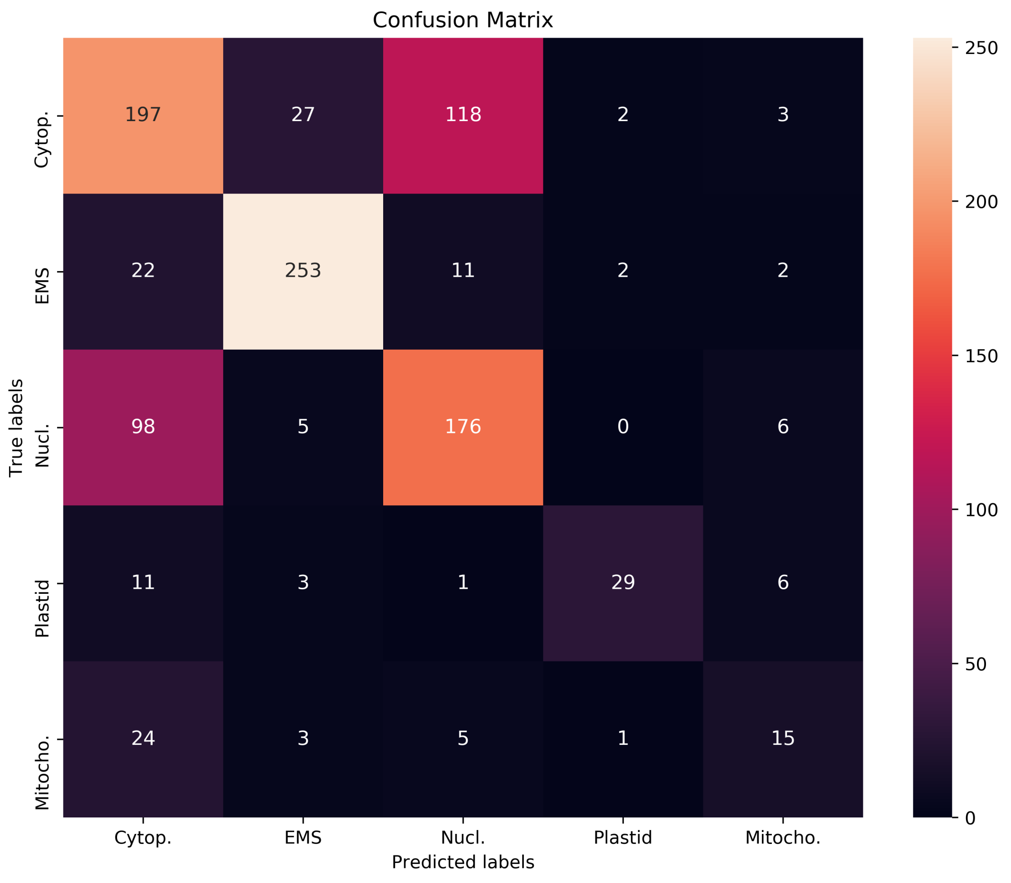


Figure 7.4: Confusion matrix for the Five-class (five-CM) model trained on the 2HL architecture.

MSA vs NoMSA

MSA information enriches the information carried in sequence. Clipping the alignments to the input data-sets is proved to consistently improve the performances of a model. As it can be seen from Table 7.5, on the same models trained without MSA information, bQ decreases between 5 to 6 percentage points, it also interesting to notice how MSA is crucial for the 3HL architecture

		1 Hidden Layer				2 Hidden Layers				3 Hidden Layers			
	Support	*predicted*	*Prec*	*Sens*	*F1-score*	*predicted*	*Prec*	*Sens*	*F1-score*	*predicted*	*Prec*	*Sens*	F1-score
Cytoplasm	347	464	0.49	0.66	0.56	325	0.51	0.48	0.49	905	0.36	0.94	0.52
Nucleus	285	210	0.56	0.41	0.48	392	0.51	0.70	0.59	0	0.	0.	0.
Secreted	139	157	0.75	0.84	0.79	142	0.82	0.83	0.83	115	0.63	0.53	0.57
Cell membrane	86	115	0.50	0.67	0.58	62	0.63	0.45	0.53	0	0.	0.	0.
Endoplasmic r. m.	65	37	0.65	0.37	0.47	56	0.61	0.52	0.56	0	0.	0.	0.
Plastid	50	24	0.50	0.24	0.32	32	0.34	0.22	0.27	0	0.	0.	0.
Mitochondrion	48	13	0.15	0.04	0.07	11	0.18	0.04	0.07	0	0.	0.	0.
Accuracy (Q)		0.55				0.56				0.39			
Accuracy (Q) with MSA		0.61				0.60				0.62			
Balanced Accuracy (bQ)		0.46				0.46				0.21			
Balanced Accuracy (bQ) with MSA		0.53				0.52				0.59			
Macro-Averaged F1-Score		0.47				0.48				0.16			
Macro-Averaged F1-Score with MSA		0.55				0.52				0.60			

Table 7.5: Table summarizing all the metrics for for seven-CM trained on the 1HL, 2HL and 3HL architectures with no MSA information.

to yield the best results, without alignments such architecture incurred the same problem the 4HL architecture incurred. Another interesting characteristic of MSA is that it improved the quality of prediction in all classes, although its effect is greater on the classes of smaller size. The three classes *Endoplastic reticulum membrane*, *Plastid* and *Mitochondrion* are the classes that saw the most improvements. For the *Mitochondrion* class, on both a 1HL and 2HL, precision and sensitivity (or recall) increased from 0.04 to 0.17 for the 1HL, for the 2HL this value increased from 0.04 to 0.12. Precision also saw a great increase, by going from 0.15 to 0.35 for the 1HL and by going from 0.18 to 0.33 for the 2HL model. With the 1HL architecture, the most numerous classes see an improvement ranging from four to fifteen percentage points (where the biggest improvement is seen in the recall for the *Nucleus* class) in both the Prec and Sens measures, on the 2HL architecture the improvements range from 4 to 28 percentage points (also for the Prec and Sens measures). It is also interesting to notice how the recall for the *Nucleus* class actually decreased from 0.70 to 0.45 in the 2HL model.

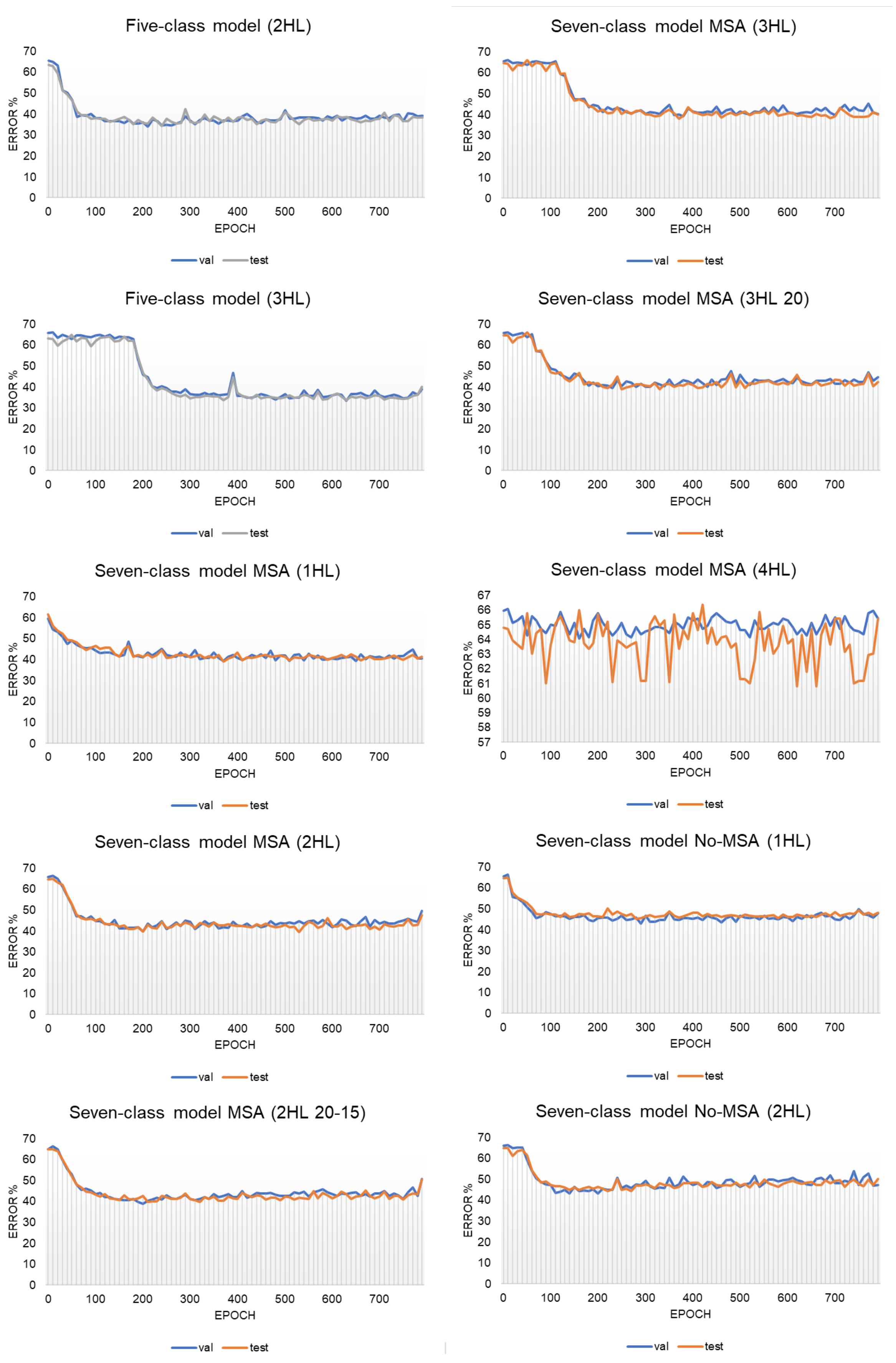


Figure 7.5: Error of validation and testing sets during the training on each model (Part 1).

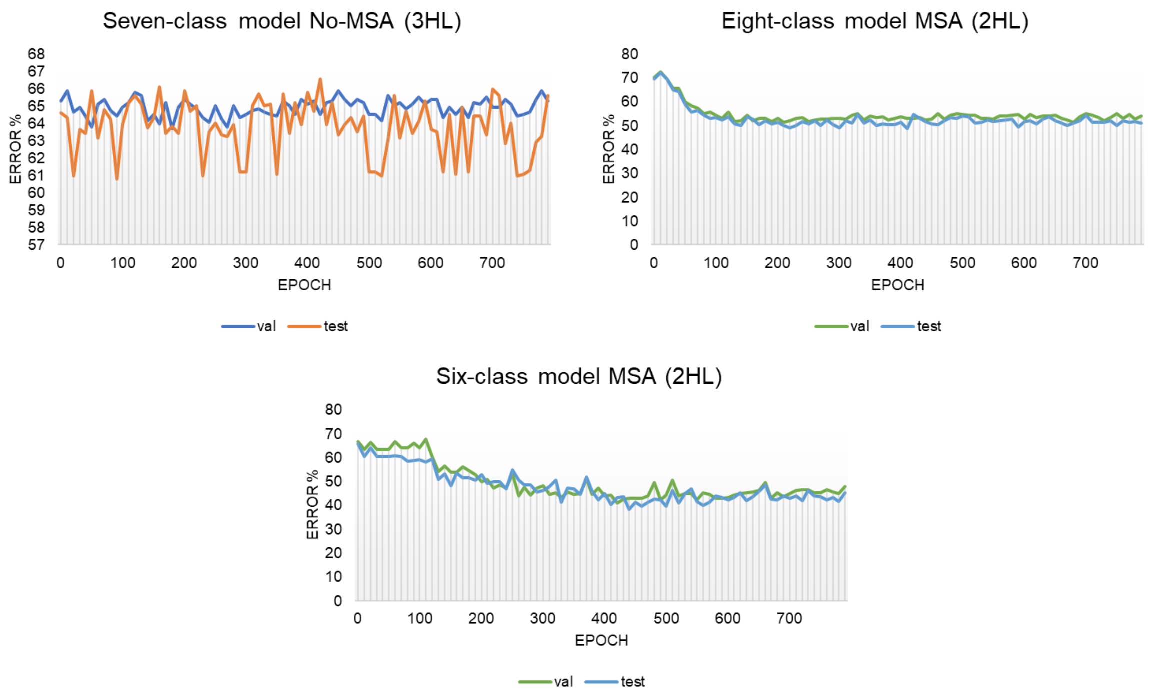


Figure 7.6: Error of validation and testing sets during the training on each model (Part 2).

Chapter 8: Evaluation using Cohen's Kappa Score

Cohen's Kappa score is a useful metric that handles multi-class classification problems and class imbalances. The important added value given by this metric is that it provides a good context of evaluation. K can be interpreted as how much of an improvement a classifier provides if compared to a randomized classifier; this helps coping with the fact that there are no available SOA classifiers producing a similar type of classification. If Table 8.1 is compared to Tables 7.2, 7.3, 7.5, it can be noted that although the magnitude of the numbers changes, K scores are for the most part between ≈ 0.4 and ≈ 0.5 while macro-F1-score, Q and bQ are mostly between ≈ 0.4 and ≈ 0.65; K exacerbates the "bad" models even more, namely the 3HL and 4HL for the seven-CM. Apart from the failing models, all the other models are sitting well within the "moderate" class defined by Lands and Koch[20]. eight-CM barely scratches to "moderate" class sitting in the "fair" class. Thest models according to the K score are still 3HL seven-CM and both the 2HL and 3HL for the five-CM. Apart from the changes in classification types, and by the provision of MSAs or lack thereof, the evaluation according to K does not show that a change in the number of hidden layers produced much higher results. in fact, increasing the number of layers only leads to a maximum improvement of 0.03. In contrast, the introduction of MSA information seems to provide much better improvements for ≈ 0.1.

	Depth	*N. of free parameters*	*K*
Eight-class model	2HL	2'978	**0.38**
Seven-class model	1HL	1'747	**0.48**
	2HL	2'967	**0.47**
	2HL, hidden f. v. of 20-15	6'772	**0.48**
	3HL	4'187	**0.51**
	3HL, hidden f. v. of 20	12'947	**0.49**
	4HL	5'407	**0.11**
Seven-class model (no MSA)	1HL	1747	**0.4**
	2HL	2967	**0.41**
	3HL	4187	**0.11**
Six-class model	2HL	2956	**0.46**
Five-class model	2HL	2945	**0.52**
	3HL	4165	**0.53**

Table 8.1: Table summarizing the evaluation of each model using K.

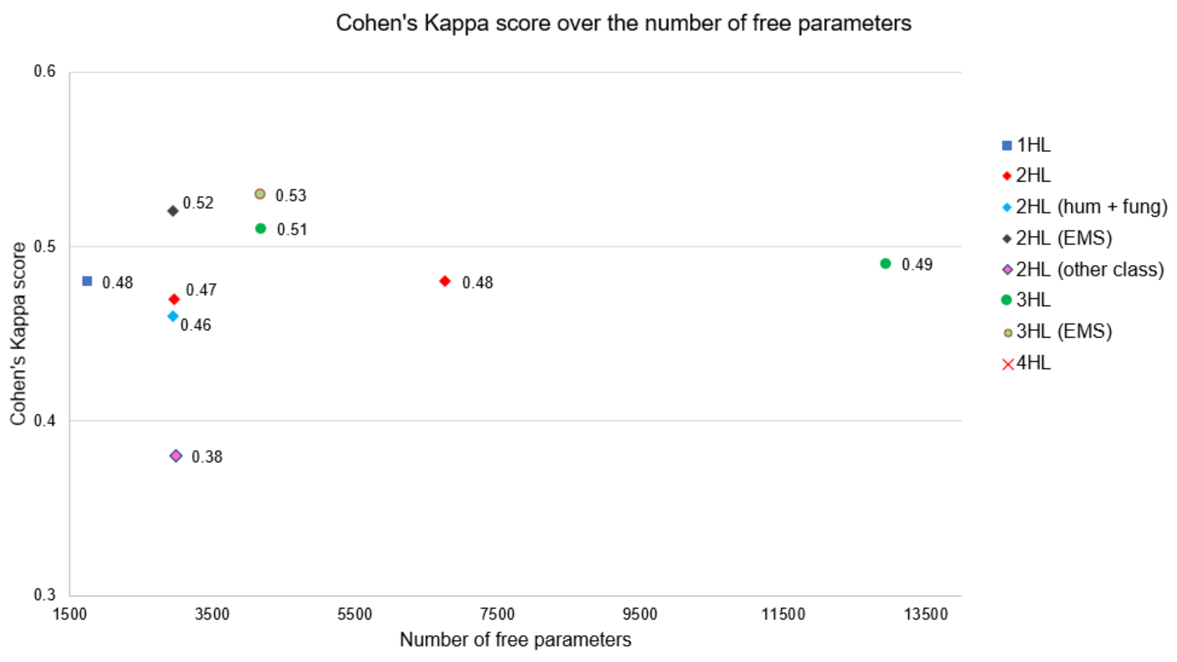


Figure 8.1: Scatter-plot summarizing all the models generated based on their K score.

Chapter 9: Summary and Conclusions

The best model produced with a high number of classes is the 3HL seven-CM, achieving K of 0.51. The worst model obtained, not considering the failed ones, is eight-CM with a "Fair" K score of 0.38. The adoption of the Kappa score allows for more confidence in saying that the models were actually providing an improvement over a randomized classification. It was shown how some classes are constantly predicted into other classes regardless of the inclusion or not of MSA information. Because of the time-demanding nature of the sequences alignment generation process, it was not feasible to attempt to generate MSAs with different degrees of predominance over the sequence motif (if MSA information is too detailed it might obscure the actual motif of the protein) and test the results they would have yielded. Similarly, k-fold validation and model ensembling were avoided for it's being too time-consuming, but it would have been of particular interest as to not only increase the sheer quality of predictions but to also increase their robustness.

Both the "Core" and "Advanced" objectives of the project have been fulfilled. The produced results provide an in-depth analysis of how different class sizes, number of free parameters and inclusion of MSA information affect the overall classifier's predictive power.

Chapter 10: **Acknowledgements**

I would like to thank the project supervisor G. Pollastri and the UCD IT services for the provision of computational facilities.

Bibliography

1. Hung, M.-C. & Link, W. Protein localization in disease and therapy. *Journal of Cell Science* **124,** 3381–3392. ISSN: 0021-9533. eprint: https://jcs.biologists.org/content/124/20/3381.full.pdf. https://jcs.biologists.org/content/124/20/3381 (2011).
2. Emanuelsson, O. Predicting protein subcellular localisation from amino acid sequence information. *Briefings in Bioinformatics* **3,** 361–376. ISSN: 1467-5463. eprint: http://oup.prod.sis.lan/bib/article-pdf/3/4/361/9731799/361.pdf. https://doi.org/10.1093/bib/3.4.361 (Dec. 2002).
3. Nair, R. & Rost, B. Better prediction of sub-cellular localization by combining evolutionary and structural information. *Proteins: Structure, Function, and Bioinformatics* **53,** 917–930. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.10507. https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.10507 (2003).
4. Bendtsen, J. D., Jensen, L. J., Blom, N., von Heijne, G. & Brunak, S. Feature-based prediction of non-classical and leaderless protein secretion. *Protein Engineering, Design and Selection* **17,** 349–356. ISSN: 1741-0126. eprint: http://oup.prod.sis.lan/peds/article-pdf/17/4/349/4694354/gzh037.pdf. https://doi.org/10.1093/protein/gzh037 (Apr. 2004).
5. Nakashima, H. & Nishikawa, K. Discrimination of Intracellular and Extracellular Proteins Using Amino Acid Composition and Residue-pair Frequencies. *Journal of Molecular Biology* **238,** 54–61. ISSN: 0022-2836. http://www.sciencedirect.com/science/article/pii/S0022283684712678 (1994).
6. Pierleoni, A., Martelli, P. L., Fariselli, P. & Casadio, R. BaCelLo: a balanced subcellular localization predictor. *Bioinformatics* **22,** e408–e416. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/22/14/e408/615167/btl222.pdf. https://doi.org/10.1093/bioinformatics/btl222 (July 2006).
7. Rost, B. & Sander, C. Improved prediction of protein secondary structure by use of sequence profiles and neural networks. *Proceedings of the National Academy of Sciences* **90,** 7558–7562. ISSN: 0027-8424. eprint: https://www.pnas.org/content/90/16/7558.full.pdf. https://www.pnas.org/content/90/16/7558 (1993).
8. Of Bioinformatics, U. C. B. I. I. R. S. I. *UniProt Consortium* Oct. 2019. https://www.uniprot.org/help/about.
9. Mooney, C., Wang, Y. & Pollastri, G. SCLpred: protein subcellular localization prediction by N-to-1 neural networks. *Bioinformatics* **27,** 2812–2819. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/27/20/2812/522508/btr494.pdf. https://doi.org/10.1093/bioinformatics/btr494 (Aug. 2011).
10. Shatkay, H. *et al.* SherLoc: high-accuracy prediction of protein subcellular localization by integrating text and protein sequence data. *Bioinformatics* **23,** 1410–1417. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/23/11/1410/16857534/btm115.pdf. https://doi.org/10.1093/bioinformatics/btm115 (Mar. 2007).
11. Almagro Armenteros, J. J., Sønderby, C. K., Sønderby, S. K., Nielsen, H. & Winther, O. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics* **33,** 3387–3395. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/33/21/3387/25166063/btx431.pdf. https://doi.org/10.1093/bioinformatics/btx431 (July 2017).

12. Consortium, G. O. The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Research* **32,** D258–D261. ISSN: 0305-1048. eprint: http://oup.prod.sis.lan/nar/article-pdf/32/suppl_1/D258/7621365/gkh036.pdf. https://doi.org/10.1093/nar/gkh036 (Jan. 2004).
13. Gianluca Pollastri, C. M. e. a. SCLpred-EMS: Subcellular localization prediction of endomembrane system and secretory pathway proteins by Deep N-to-1 Convolultional Neural Networks (2019).
14. Mooney, C., Cessieux, A., Shields, D. C. & Pollastri, G. SCL-Epred: a generalised de novo eukaryotic protein subcellular localisation predictor. *Amino Acids* **45,** 291–299. ISSN: 1438-2199. https://doi.org/10.1007/s00726-013-1491-3 (Aug. 2013).
15. Adelfio, A., Volpato, V. & Pollastri, G. SCLpredT: Ab initio and homology-based prediction of subcellular localization by N-to-1 neural networks. *SpringerPlus* **2,** 502. ISSN: 2193-1801. https://doi.org/10.1186/2193-1801-2-502 (Oct. 2013).
16. 2006. https://pdfs.semanticscholar.org/95df/dc02010b9c390878729f459893c2a5c0898f.pdf.
17. Shmueli, B. *Multi-Class Metrics Made Simple, Part II: the F1-score* Dec. 2019. https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1.
18. Ramaswamy, H. G. *Optimizing the Multiclass F-measure via Biconcave Programming* https://www.cse.iitk.ac.in/users/purushot/papers/macrof1.pdf.
19. *Performance Measures: Cohen's Kappa statistic* Apr. 2020. https://thedatascientist.com/performance-measures-cohens-kappa-statistic/.
20. Landis, J. R. & Koch, G. G. The measurement of observer agreement for categorical data. *Biometrics* **33 1,** 159–74 (1977).
21. Casadio, R., Martelli, P. L. & Pierleoni, A. The prediction of protein subcellular localization from sequence: a shortcut to functional genome annotation. *Briefings in Functional Genomics* **7,** 63–73. ISSN: 2041-2649. eprint: http://oup.prod.sis.lan/bfg/article-pdf/7/1/63/641156/eln003.pdf. https://doi.org/10.1093/bfgp/eln003 (Feb. 2008).
22. Altschul, S. F. *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research* **25,** 3389–3402. ISSN: 0305-1048. eprint: http://oup.prod.sis.lan/nar/article-pdf/25/17/3389/3639509/25-17-3389.pdf. https://doi.org/10.1093/nar/25.17.3389 (Sept. 1997).
23. Suzek, B. E., Huang, H., McGarvey, P., Mazumder, R. & Wu, C. H. UniRef: comprehensive and non-redundant UniProt reference clusters. *Bioinformatics* **23,** 1282–1288. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/23/10/1282/499932/btm098.pdf. https://doi.org/10.1093/bioinformatics/btm098 (Mar. 2007).
24. Savojardo, C., Martelli, P. L., Fariselli, P. & Casadio, R. DeepSig: deep learning improves signal peptide detection in proteins. *Bioinformatics* **34,** 1690–1696. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/34/10/1690/25118318/btx818.pdf. https://doi.org/10.1093/bioinformatics/btx818 (Dec. 2017).
25. Goldberg, T. *et al.* LocTree3 prediction of localization. *Nucleic Acids Research* **42,** W350–W355. ISSN: 0305-1048. eprint: http://oup.prod.sis.lan/nar/article-pdf/42/W1/W350/17423232/gku396.pdf. https://doi.org/10.1093/nar/gku396 (May 2014).
26. Almagro Armenteros, J. J. *et al.* SignalP 5.0 improves signal peptide predictions using deep neural networks. *Nature Biotechnology* **37,** 420–423. ISSN: 1546-1696. https://doi.org/10.1038/s41587-019-0036-z (2019).

27. Bakheet, T. M. & Doig, A. J. Properties and identification of human protein drug targets. *Bioinformatics* **25,** 451–457. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/25/4/451/16892396/btp002.pdf. https://doi.org/10.1093/bioinformatics/btp002 (Jan. 2009).

28. Goldberg, T., Hamp, T. & Rost, B. LocTree2 predicts localization for all domains of life. *Bioinformatics* **28,** i458–i465. ISSN: 1367-4803. eprint: http://oup.prod.sis.lan/bioinformatics/article-pdf/28/18/i458/16907725/bts390.pdf. https://doi.org/10.1093/bioinformatics/bts390 (Sept. 2012).

29. Chou, K.-C. & Shen, H.-B. Euk-mPLoc: A Fusion Classifier for Large-Scale Eukaryotic Protein Subcellular Location Prediction by Incorporating Multiple Sites. *Journal of Proteome Research* **6,** 1728–1734. ISSN: 1535-3893. https://doi.org/10.1021/pr060635i (May 2007).

30. Jia, P., Qian, Z., Zeng, Z., Cai, Y. & Li, Y. Prediction of subcellular protein localization based on functional domain composition. *Biochemical and Biophysical Research Communications* **357,** 366–370. ISSN: 0006-291X. http://www.sciencedirect.com/science/article/pii/S0006291X07006067 (2007).

31. Yu, C.-S., Chen, Y.-C., Lu, C.-H. & Hwang, J.-K. Prediction of protein subcellular localization. *Proteins: Structure, Function, and Bioinformatics* **64,** 643–651. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/prot.21018. https://onlinelibrary.wiley.com/doi/abs/10.1002/prot.21018 (2006).

32. Briesemeister, S. *et al.* SherLoc2: A High-Accuracy Hybrid Method for Predicting Subcellular Localization of Proteins. *Journal of proteome research* **8,** 5363–6 (Sept. 2009).

33. http://weizhongli-lab.org/cd-hit/.

34. *The Yang Zhang Lab* https://zhanglab.ccmb.med.umich.edu/.

35. Ozansoy, M. & Denizhan, Y. The Endomembrane System: A Representation of the Extracellular Medium? *Biosemiotics* **2,** 255–267 (Dec. 2009).