Convolutional Neural Networks in Combination with Support Vector Machines for Complex Sequential Data Classification

Antreas Dionysiou[1], Michalis Agathocleous[1], Chris Christodoulou[1(✉)], and Vasilis Promponas[2]

[1] Department of Computer Science, University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus
{adiony01,magath06,cchrist}@cs.ucy.ac.cy
[2] Department of Biological Sciences, University of Cyprus, P.O. Box 20537, 1678 Nicosia, Cyprus
vprobon@ucy.ac.cy

Abstract. Trying to extract features from complex sequential data for classification and prediction problems is an extremely difficult task. Deep Machine Learning techniques, such as Convolutional Neural Networks (CNNs), have been exclusively designed to face this class of problems. Support Vector Machines (SVMs) are a powerful technique for general classification problems, regression, and outlier detection. In this paper we present the development and implementation of an innovative by design combination of CNNs with SVMs as a solution to the Protein Secondary Structure Prediction problem, with a novel two dimensional (2D) input representation method, where Multiple Sequence Alignment profile vectors are placed one under another. This 2D input is used to train the CNNs achieving preliminary results of 80.40% per residue accuracy (Q3), which are expected to increase with the use of larger training datasets and more sophisticated ensemble methods.

Keywords: Convolutional Neural Networks
Support Vector Machines · Deep learning · Machine learning
Bioinformatics · Protein Secondary Structure Prediction

1 Introduction

Learning, is a many-faceted phenomenon. The learning process includes the acquisition of new declarative knowledge, the development of cognitive skills through instructions and practice, the organizing of new knowledge into general, the effective representation of data and finally, the discovery of new theories and facts through practice and experimentation. Analysis of sequential data, feature extraction and prediction through Machine Learning (ML) algorithms/techniques, has been excessively studied. Nevertheless, the complexity

V. Kůrková et al. (Eds.): ICANN 2018, LNCS 11140, pp. 444–455, 2018.
https://doi.org/10.1007/978-3-030-01421-6_43

and divergence of the big data that exist nowadays keep this field of research open. When designing ML techniques for complex sequential data prediction, one must take into account, (a) how to capture both short- and long-range sequence correlations [1], and (b) how to focus on the most relevant information in large quantities of data [2].

A Convolutional Neural Network (CNN) is a class of deep, feedforward artificial neural networks (NN) that has successfully been applied to analyzing visual imagery [3,4]. CNNs were inspired by the human visual system, where individual cortical neurons respond to stimuli, only in a restricted region of the visual field, known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs have enjoyed a great success in large-scale image and video recognition [5]. This has become possible due to the large public image repositories, such as ImageNet [3], and high-performance computing systems, such as GPUs or large-scale distributed clusters [6]. Overall, CNNs are in general a good option for feature extraction, immense complexity sequence and pattern recognition problems [3–10].

Support Vector Machines (SVMs) were introduced by Cortes and Vapnik [11], initially for binary classification problems. SVMs are a powerful technique for linearly and non-linearly separable classification problems, regression, and outlier detection, with an intuitive model representation [11].

A challenging task for ML techniques is to make predictions on sequential data that encode high complexity of interdependencies and correlations. Application examples include problems from Bioinformatics such as Protein Secondary Structure Prediction (PSSP) [12–15]; even though the three dimensional (3D) structure of a protein molecule is determined largely by its amino acid sequence, yet, the understanding of the complex sequence-structure relationship is one of the greatest challenges in computational biology. A ML model designed for such data has to be in position to extract relevant features, and at the same time reveal any long/short range interdependencies in the sequence of data given. The major key point that needs to be considered when trying to solve the PSSP problem is the complex sequence correlations and interactions between the amino acid residues of a protein molecule. In order to maximize the prediction accuracy of a proposed NN technique for a specific amino acid in a protein molecule, the adjacent amino acids have to be considered by the proposed NN architecture.

In this paper we present a hybrid machine learning method based on the application of CNNs in combination with SVMs, for complex sequential data classification and prediction. The implemented model is then tested on the PSSP problem for 3-state secondary structure (SS) prediction.

2 Methodology

2.1 The CNN Architecture

CNNs are biologically-inspired variants of Multi-Layer Perceptrons (MLPs). The CNN architecture consists of an input layer (inactive), multiple hidden layers and an output layer. Generally speaking, CNNs combine three architectural ideas to

ensure some degree of shift, scale, and distortion invariance: local receptive fields, shared weights, and spatial subsampling/pooling [7]. The hidden layers of a CNN typically consist of convolutional layers, pooling layers and fully connected layers. There are four main operations performed by a CNN: (a) convolution, (b) non linearity (Rectifier Linear Unit - ReLU), (c) pooling or sub sampling, and (d) classification. One of the major characteristics of CNNs is that they take advantage of the fact that the input would be like an "image", so they constrain the architecture in a more sensible way. Every layer of a CNN transforms one volume of activations to another through a differential function. The arrangement of a CNN's neurons, unlike a regular NN, is in 3 dimensions: width, height and depth. The Convolutional Layer (CL) is the core building block of a CNN that basically performs the feature extraction process. The key hyperparameter of a CL is the kernel. The kernel is basically a 2D array initialized with random values, and it is used to compute dot products between the entries of the filter and the input volume at any position. The stride is another important hyperparameter that defines the amount of sliding of the kernel across the width and height of the input volume. The result of the kernel sliding over the width and height of the input volume is the feature map, a 2D array holding the responses/activations of the kernel at any spatial position. Moreover, the CNNs' ability to handle complex sequential data relies in part to the sparse connections of neurons. More specifically, each neuron is connected to only a local region of the input volume (i.e., receptive field), and as a result CNNs are capable of encoding complex sequential data correlations in their structure. The Pooling Layer (PL) is another critical block, for building a CNN. Generally speaking, a common technique for constructing a CNN is to insert a pooling layer in-between successive CLs. The main purpose of a pooling layer is to (a) reduce the representation size, (b) reduce the amount of computation in the NN, and (c) control overfitting. The PL uses a filter of a certain dimension and resizes the input given spatially, by striding the filter across the input volume and performing usually the MAX operation. The last layer of a CNN is usually a fully-connected Softmax output layer. Nevertheless, this final step can be practically realized with any suitable classifier. In particular, a small advantage was reported when the softmax output layer of a CNN was replaced by a linear SVM [16].

In this work, the libraries used for CNN and SVM implementations are Deeplearning4j (https://deeplearning4j.org) and LibSVM [17] with Scikit-learn front-end (http://scikit-learn.org), respectively.

2.2 Data Representation

As mentioned above, CNNs are capable of analyzing image-like inputs. The major obstacle on trying to solve a complex sequential data classification problem with CNNs is the representation of the data, in such a way that the network is able not only to understand the shape of the input volume, but also to track the complex sequence correlations among the input volume. Transforming the sequential data shape so as to make it look like an "image", allows CNNs to capture the complex sequence-structure relationship, including to model the SS

interactions among adjacent or distant amino acid residues in the PSSP problem. Along these lines, we reorganised the input data shape so that the vectors of each sample in the sequential data are placed one under another, and in such a way create an "image-like" input that will be effectively read correctly and understood by the CNN. In particular, for PSSP we have created a new input volume by placing Multiple Sequence Alignment (MSA) [18] profile vectors of each amino acid one under another to construct a 2D representation of the MSA profiles of a certain number of neighbouring amino acid residues (Fig. 1). By sliding the kernel over the newly constructed input volume, CNNs are able to perform feature extraction for each record data, but also consider neighboring correlations and interactions, if any exist. Note that unlike other techniques, the attention given to any neighboring record correlations is equally weighted across all the input volume, for each sample given. This lets the CNN discover and capture any short, mid- and long range correlations among the input records and consider them all equally in terms of the output volume created. One of the major contributions of this paper is this innovative input data representation, especially designed for the complex sequential data of the PSSP problem.

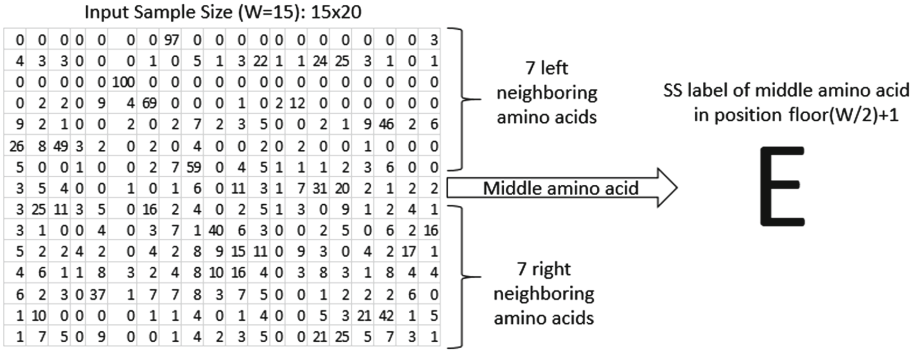

Fig. 1. Example of Data Representation Method: An example of data representation of an input sample using a window size of 15 amino acids. Each line represents the MSA profile vector for the specific amino acid. The SS label for the example input sample showed in this figure, is the SS label for the middle amino acid.

2.3 Application Domain and Data

High quality datasets for training and validation purposes are a prerequisite when trying to construct useful prediction models [2]. Therefore, we have chosen PSSP a well known bioinformatics problem, which is characterized by the complexity of the correlations between the data records due to the existence of combinations of short, mid and long range interactions.

The PSSP, which is based on the Primary Structure (PS) of a protein molecule is considered to be an important problem, since the SS can be seen as a low-resolution snapshot of a protein's 3D structure, and can thus shed light

on its functional properties and assist in many other applications like drug and enzyme design. As mentioned above, the understanding of the complex sequence-structure relationship is one of the greatest challenges for the PSSP problem. Since the currently known experimental methods for determining the 3D structure of a protein molecule are expensive, time consuming and frequently inefficient [12], different methods and algorithms for predicting the secondary structure of a protein molecule have been developed [8,12,14,15,19,20]. In particular, Recurrent Neural Networks (RNNs) were successful in the PSSP problem [20], as their architecture may capture both short- and long-range interactions needed for PSSP. CNNs though can detect and extract high complexity features from an input sequence and at the same time track any short-, mid- or long-range interactions depending on the window size. Thus we decided to use CNNs in combination with our novel data representation method for the PSSP problem.

A protein is typically composed by 20 different amino acid types which are chemically connected to form a polypeptide chain, folding into a 3D structure by forming any-range interactions. There are eight main SS states that each amino acid can be assigned to, when a protein 3D structure is available, which are typically grouped in three classes, namely: Helix (H), Extended (E) and Coil/Loop (C/L) with different geometrical and hydrogen-bonding properties. In this work, we use CB513 [19], a non-redundant dataset which has been heavily used as a benchmark for the PSSP problem that contains 513 proteins excluding eight proteins with names: 1coiA_1-29, 1mctI_1-28, 1tiiC_195-230, 2erlA_1-40, 1ceoA_202-254, 1mrtA_31-61, 1wfbB_1-37 and 6rlxC_-2-20 due to corrupted MSA profiles. The use of MSA profiles enhanced the performance of PSSP ML algorithms, since they incorporate information of homologous sequences, which may facilitate the detection of subtle, yet important, patterns along the sequences [14]. In particular, for representing each protein sequence position, we use a 20-dimensional vector, which corresponds to the frequencies of 20 different amino acid types as calculated from a PSI-BLAST [21] search against the NCBI-NR (NCBI: https://www.ncbi.nlm.nih.gov/) database. Note that we have also performed an experiment on a much larger dataset, namely PISCES [22] which shows promising results.

2.4 Support Vector Machines (SVMs)

The main idea behind SVMs is that the input vectors are non-linearly mapped to a higher dimensional feature space using an appropriate kernel function with the hope that a linearly inseparable problem in the input space becomes linearly separable in the new feature space, i.e., a linear decision surface can constructed [23]. An important advantage of SVMs is that the search for the decision surface that maximizes the margin among the target class instances ensures high generalization ability of the learning machine [24]. Their robust performance with respect to sparse and noisy data makes them a good choice in a number of applications from text categorization to protein function prediction [25]. Moreover, SVMs were shown to be the best technique for filtering on the PSSP problem [13]. Given this, we decided to test the filtering capabilities of SVMs

on the CNNs' SS prediction results, to see whether the accuracy is improved, and correct the predicted SS of a protein molecule gathered from an ensemble of CNNs.

3 Results and Discussion

3.1 Optimising the Parameters

The CNN implementation using the innovative input data representation described in Sect. 2.2 has been used and tested on the PSSP problem. To train the CNN, we have used the already mentioned CB513 dataset. More specifically, the model's input was a combination of a certain number of neighboring amino acids MSA profile record vectors, one under another, forming a 2D array. The target output label was the SS class for the middle point amino acid that had been examined.

A single CNN has been trained each time. We have decided to track the optimal hyperparameter values using a specific fold after dividing CB513 dataset into ten (10) folds. The main reason for optimizing the hyperparameters on a specific fold is the small size of CB513 dataset. Accuracy results using different hyperparameter values on the other folds are not expected to vary considerably. During this phase, multiple experiments were performed in order to tune up our model and finally achieve the highest results using the CNN. These were Q3 of 75.155% and Segment OVerlap (SOV [30]) of 0.713. CNNs with different numbers of CLs, PLs, kernel sizes, strides, number of parallel filters in each CL, and Gradient Descent (GD) optimization algorithms (Fig. 2) have been tested for optimising the parameter values. The optimization algorithms used are: Gradient Descent (GD), Gradient Descent with momentum (GD with momentum), Adaptive Gradient Algorithm (AdaGrad) [26], RMSprop [27], AdaDelta [28], Adaptive Moment Estimation (Adam) [29]. The two most critical hyperparameters that showed a big impact on the results are: (a) the optimization method used and (b) the number of neighboring amino acids to be considered in each sample (window size). More specifically, the parameter W is the number of total amino acids to be considered by the CNN when trying to predict the SS of the floor(W/2) + 1 amino acid. Then, according to the W parameter we reconstruct the input sample so as to become a 2D array with shape $W \times 20$. The results are shown in Fig. 3. Unlike Wang's et al. [8] method, where they use 42 input features for each residue in an one dimensional input vector format, we use $20 \times W$ (20 input features for each amino acid $\times$ window size) input features for each residue in a two dimensional input vector format where each line represents the MSA profile of an amino acid at any specific position. Generally speaking Wang's et al. [8] 42 input features used include our 20 input features (MSA profile for each amino acid) plus extra 22 input features for each amino acid. In this way, our method reduces the dimensionality of the problem without losing too much important information. Moving forward, we had to tune up the parameters that determine the network's architecture.

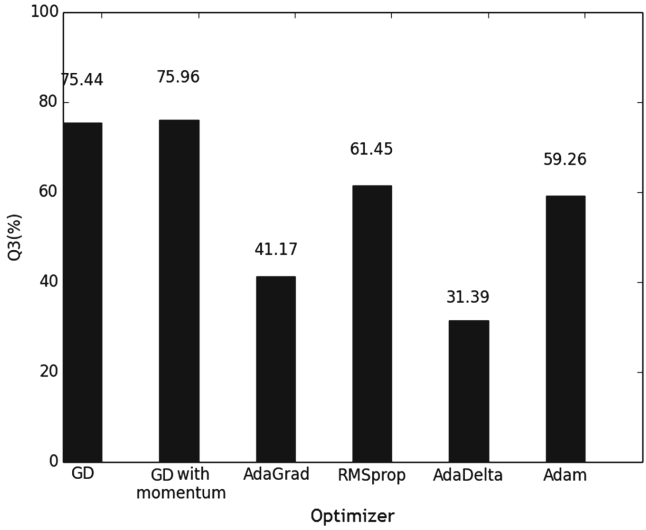

Fig. 2. Optimizers: CNNs Q3 accuracy results using different Gradient Descent (GD) optimization algorithms.

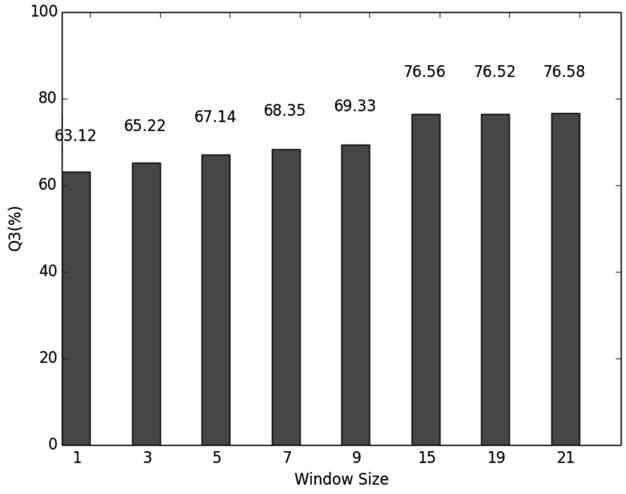

Fig. 3. Window Size: CNNs Q3 accuracy results with different window (W) sizes.

To get a general idea about the CNN performance we have trained it using the CB513 dataset. After tuning up the network architecture, the following optimal CNN parameter values resulted: (a) Number of convolutional layers: 3, (b) Number of Pooling Layers: 0, (c) Kernel/Filter size: 2×2, (d) Stride: 1, (e) Number of Parallel Filters per Layer: 5, (f) Neurons Activation Function: Leaky ReLU, and (g) Optimization method: Gradient Descent with momentum = 0.85.

The number of neighboring amino acids (W) that leads to some among the highest Q3 results and at the same time limiting the complexity of information been used (i.e., minimizing the window) was 15. Moreover, no significant change on Q3 accuracy results was noticed using larger window (W) sizes (Fig. 3). Based on the results, we realized that (i) smaller W values do not provide enough information to the network regarding the adjacent interactions between amino acids, and (ii) larger W values contain way too much (unnecessary in some way) information for the network to be handled and decoded properly.

We did not use pooling layers for our CNN architecture due to the fact that subsampling the features gathered from CNN is not relevant in the PSSP problem. Getting only the maximum value of a spatial domain does not work in PSSP as every value extracted from CLs may represent interactions of amino acids in a certain region. These are the most important factors that lead to low Q3 and SOV results using PLs.

3.2 10-Fold Cross-Validation on CB513

In order to validate the robustness of the model as well as to prove its efficiency to the exposure of various training and testing data, we had to complete the evaluation of the PSSP problem on the CB513 dataset, using a 10-fold cross-validation test. All the experiments made are with the optimal parameters of the model as described in Sect. 3.1. As shown in Table 1, the Q3 and SOV accuracy results of CNN with 10-fold cross-validation are 75.15% and 0.713 respectively.

Table 1. Summary of the results for all methods.

Method	$Q_3(\%)$	$Q_H(\%)$	$Q_E(\%)$	$Q_L(\%)$	*SOV*	SOV_H	SOV_E	SOV_L
CNN	75.155	69.474	67.339	84.566	0.713	0.696	0.669	0.734
CNN Ensembles	78.914	72.748	68.854	85.385	0.744	0.738	0.722	0.737
CNN Ens. + ER Filt.	78.692	70.147	66.921	87.053	**0.756**	0.669	0.713	0.731
CNN Ens. + SVM Filt.	**80.40**	80.911	70.578	85.165	0.736	0.724	0.716	0.743

3.3 Ensembles and External Rules Filtering

After tracking the optimal parameters for the CNN, we have performed six (6) experiments for each fold. Then, in an attempt to maximize the quality of the results gathered as well as to increase the Q3 and SOV accuracy, we proceeded with using the winner-take-all ensembles technique [31,32] on every single fold separately. This technique obtains the predictions of a number of same ML model experiments, and applies the winner takes all method on each amino acid residue SS class predicted. The dramatically improved results are shown in Table 1.

Filtering the SS prediction using external empirical rules is usually the last step made, as a final attempt to improve the quality of the results. This is accomplished by removing conformations that are physicochemically unlikely to

happen [15]. Applying the external rules filtering on the CNN's SS prediction, interestingly, does not improve the Q3 score, but it improves the SOV. The results are shown in Table 1.

3.4 Filtering Using Support Vector Machines (SVMs)

CNNs showed very good results on the PSSP (Figs. 2, 3 and Table 1). Nevertheless, as mentioned above, we tried to use SVMs to perform the filtering task. More specifically, after gathering the predictions from the CNN we have trained a SVM using a window of SS states predicted by the CNN. After performing several experiments using different kernels, misclassification penalty parameters (C) [11], Gamma values (G) [11] and window sizes (WIN), we have decided for the optimal SVM parameters that lead to the highest Q3 and SOV accuracy on the PSSP problem and which are: (a) Kernel: Radial Basis Function, (b) C = 1, (c) G = 0.001 and (d) WIN = 7. The results are shown in Tables 2 and 3.

3.5 Summary of the Results

The results shown in Table 1 summarize the Q3 accuracy and SOV results gathered, with all the methods discussed in this paper, using 10-fold cross-validation. It is shown that the CNN can achieve relatively high Q3 and SOV results (75.155% and 0.713 respectively) by its own. Nevertheless, the CNN using ensembles improved the Q3 accuracy results by approximately 3% and SOV score by 0.031. Moving on, filtering the results using External Rules mentioned above, decreases the overall Q3 accuracy results to 78.692%, but dramatically increases the SOV score from 0.744 to 0.756. This was expected as filtering with External Rules has previously been reported to improve SOV scores, but at the same time decrease the overall Q3 accuracy [12]. Finally, using the combination of CNN ensembles and SVM as a filtering technique, achieves the highest Q3 accuracy results (80.40%). The Q3 values for different folds vary from 78.96% to 83.91% and the SOV from 0.71 to 0.78 (Table 2). This indicates that the results for different folds are of comparable quality. Moreover, the accuracies for the three classes, H, E, L, are calculated separately (see Q_H, Q_E, Q_L and SOV_H, SOV_E, SOV_L in Table 2) for getting deeper insight on the quality of the classifier, and mispredictions are quantified in a confusion matrix, graphically represented in Fig. 4. As we can see from Table 2, Q3 accuracy results gathered using CNN Ensembles and SVM filtering are just over **80%**, which is considered to be a high enough percentage when it comes to PSSP, and which also makes this combination of NN techniques a good option when it comes to complex sequential data classification and prediction problems. Heffernan's et al. [20] method achieves 84.16% Q3 accuracy using Bidirectional Recurrent Neural Networks without using a window, but these results are not directly comparable with our results, as they make use of a much larger dataset that contains 5789 proteins, compared to CB513 which contains 513 proteins.

As a conclusion to all the results presented in this paper, we can see that the CNNs can effectively detect and extract features from complex sequential data,

Table 2. CNN Ensembles and SVM Filtering: Q3 and SOV Results for each Fold.

Fold	$Q_3(\%)$	$Q_H(\%)$	$Q_E(\%)$	$Q_L(\%)$	*SOV*	SOV_H	SOV_E	SOV_L
0	79.69	79.77	70.05	84.75	0.74	0.73	0.71	0.75
1	79.74	78.69	68.06	86.77	0.73	0.73	0.71	0.74
2	78.96	78.64	68.27	84.94	0.72	0.71	0.71	0.73
3	79.55	79.09	67.89	86.12	0.71	0.72	0.70	0.73
4	79.26	78.55	70.00	84.79	0.73	0.72	0.73	0.72
5	79.70	80.27	70.18	84.31	0.73	0.71	0.72	0.73
6	79.64	79.85	68.87	85.26	0.73	0.73	0.71	0.74
7	83.70	87.68	76.86	83.91	0.76	0.73	0.71	0.77
8	83.91	87.53	76.33	84.62	0.78	0.75	0.74	0.79
9	79.85	79.04	69.27	86.18	0.73	0.71	0.72	0.73
Avg.	**80.40**	**80.91**	**70.57**	**85.16**	**0.736**	**0.724**	**0.716**	**0.743**

Table 3. CNN Ensembles and SVM Filtering: Statistical Analysis

	Q3	SOV
Sample standard deviation (s)	1.8140	0.0141
Variance (Sample standard) (s^2)	3.2906	0.0002
Mean (Average)	80.4	0.736
Standard error of the mean ($SE_{\bar{\chi}}$)	0.5736	0.0044

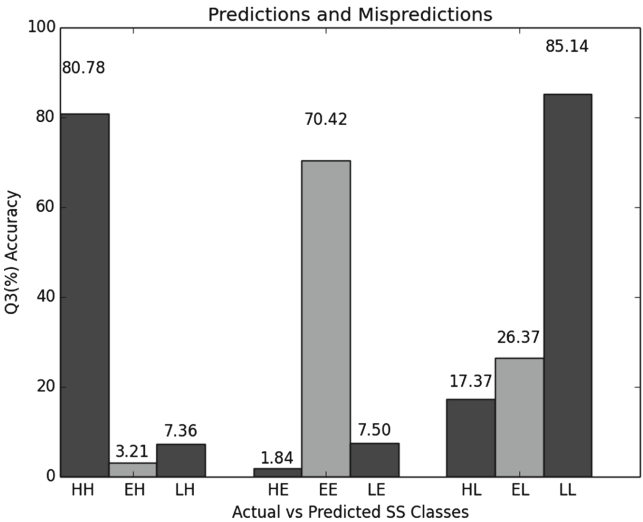

Fig. 4. Confusion Matrix: Predictions and mispredictions of the secondary structure classes H, E and C/L after applying ensembles on each fold using CB513 dataset. Q3 accuracy scores are shown for each class.

by utilizing our proposed "image" like data representation method used to train the CNNs for the PSSP problem. This is due to the fact that our CNN architecture was exclusively designed to face such problems. In addition, SVMs seem to be a good technique to be used for filtering the CNN output. The combination though, of these two ML algorithms seem to be a great option for complex feature extraction and prediction on sequential data, as we take advantage of the benefits of both techniques. Finally, by observing the results from the confusion matrix of Fig. 4, we can conclude that the combination of CNNs with SVMs filtering is a robust and high quality methodology and architecture, as it maximizes the correct predictions for each SS class. Results are expected to be improved by collecting more experiments for each fold, using larger datasets (e.g., PISCES) and deploying more sophisticated ensemble techniques.

References

1. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013)
2. Blum, A.L., Langley, P.: Selection of relevant features and examples in machine learning. Artif. Intell. **97**(1–2), 245–271 (1997)
3. Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet classication with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) Advances in Neural Information Processing Systems 25: Proceedings of the 26th International Conference on Neural Information Processing Systems, pp. 1097–1105. Curran Associates, Lake Tahoe, Nevada, Red Hook, NY (2012)
4. Rawat, W., Wang, Z.: Deep convolutional neural networks for image classification: a comprehensive review. Neural Comput. **29**(9), 2352–2449 (2017)
5. Srinivas, S., Sarvadevabhatla, R.K., Mopuri, K.R., Prabhu, N., Kruthiventi, S.S., Babu, R.V.: A taxonomy of deep convolutional neural nets for computer vision. Front. Robot. AI **2**, 36 (2016)
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
7. LeCun, Y., Bengio, Y.: Convolutional networks for images, speech, and time series. In: Arbib, M.A. (ed.) The Handbook of Brain Theory and Neural Networks, pp. 255–258. MIT Press, Cambridge (1998)
8. Wang, S., Peng, J., Ma, J., Xu, J.: Protein secondary structure prediction using deep convolutional neural fields. Sci. Rep. **6**, 18962 (2016)
9. Bluche, T., Ney, H., Kermorvant, C.: Feature extraction with convolutional neural networks for handwritten word recognition. In: Proceedings of the 12th IEEE International Conference on Document Analysis and Recognition, pp. 285–289 (2013)
10. Graves, A., Mohamed, A.R., Hinton, G.: Speech recognition with deep recurrent neural networks. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, pp. 6645–6649 (2013)
11. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
12. Baldi, P., Brunak, S., Frasconi, P., Soda, G., Pollastri, G.: Exploiting the past and the future in protein secondary structure prediction. Bioinformatics **15**(11), 937–946 (1999)

13. Kountouris, P., Agathocleous, M., Promponas, V.J., Christodoulou, G., Hadjicostas, S., Vassiliades, V., Christodoulou, C.: A comparative study on filtering protein secondary structure prediction. IEEE/ACM Trans. Comput. Biol. Bioinform. **9**(3), 731–739 (2012)
14. Rost, B., Sander, C.: Combining evolutionary information and neural networks to predict protein secondary structure. Proteins: Struct. Funct. Bioinform. **19**(1), 55–72 (1994)
15. Salamov, A.A., Solovyev, V.V.: Prediction of protein secondary structure by combining nearest-neighbor algorithms and multiple sequence alignments. J. Mol. Biol. **247**(1), 11–15 (1995)
16. Tang, Y.: Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239 (2013)
17. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27 (2011)
18. Wallace, I.M., Blackshields, G., Higgins, D.: Multiple sequence alignment. Curr. Opin. Struct. Biol. **15**(3), 261–266 (2005)
19. Cuff, J.A., Barton, G.J.: Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. Proteins: Struct. Funct. Bioinform. **34**(4), 508–519 (1999)
20. Heffernan, R., Yang, Y., Paliwal, K., Zhou, Y.: Capturing non-local interactions by long short-term memory bidirectional recurrent neural networks for improving prediction of protein secondary structure, backbone angles, contact numbers and solvent accessibility. Bioinformatics **33**(18), 2842–2849 (2017)
21. Schaffer, A.A., et al.: Nucl. Acids Res. **25**, 3389–3402 (1997)
22. Wang, G., Dunbrack Jr., R.L.: PISCES: a protein sequence culling server. Bioinformatics **19**(12), 1589–1591 (2003)
23. Vapnik, V.N.: An overview of statistical learning theory. IEEE Trans. Neural Netw. **10**(5), 988–999 (1999)
24. Meyer, D., Wien, F.T.: Support vector machines. R News **1**(3), 23–26 (2001)
25. Furey, T.S., Cristianini, N., Duffy, N., Bednarski, D.W., Schummer, M., Haussler, D.: Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics **16**(10), 906–914 (2000)
26. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. J. Mach. Learn. Res. **12**, 2121–2159 (2011)
27. Tieleman, T., Hinton, G.: Lecture 6.5 - RMSProp, Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Netw. Mach. **4**(2), 26–31 (2012)
28. Zeiler, M. D.: ADADELTA: An Adaptive Learning Rate Method. arXiv preprint arXiv:1212.5701 (2012)
29. Kingma, D. P., Ba, J. L.: Adam: a method for stochastic optimization. In: Suthers, D., Verbert, K., Duval, E., Ochoa, X. (Eds.) Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015), Leuven, Belgium, pp. 1–13. ACM, New York, NY, USA (2015)
30. Rost, B., Sander, C., Schneider, R.: Redefining the goals of protein secondary structure prediction. J. Mol. Biol. **235**(1), 13–26 (1994)
31. Granitto, P.M., Verdes, P.F., Ceccatto, H.A.: Neural network ensembles: evaluation of aggregation algorithms. Artif. Intell. **163**(2), 139–162 (2005)
32. Fukai, T., Tanaka, S.: A simple neural network exhibiting selective activation of neuronal ensembles: from winner-take-all to winners-share-all. Neural Comput. **9**(1), 77–97 (1997)