


OPEN

# Deeper Profiles and Cascaded Recurrent and Convolutional Neural Networks for state-of-the-art Protein Secondary Structure Prediction

Mirko Torrissi , Manaz Kaleel & Gianluca Pollastri

Protein Secondary Structure prediction has been a central topic of research in Bioinformatics for decades. In spite of this, even the most sophisticated ab initio SS predictors are not able to reach the theoretical limit of three-state prediction accuracy (88–90%), while only a few predict more than the 3 traditional Helix, Strand and Coil classes. In this study we present tests on different models trained both on single sequence and evolutionary profile-based inputs and develop a new state-of-the-art system with Porter 5. Porter 5 is composed of ensembles of cascaded Bidirectional Recurrent Neural Networks and Convolutional Neural Networks, incorporates new input encoding techniques and is trained on a large set of protein structures. Porter 5 achieves 84% accuracy (81% SOV) when tested on 3 classes and 73% accuracy (70% SOV) on 8 classes on a large independent set. In our tests Porter 5 is 2% more accurate than its previous version and outperforms or matches the most recent predictors of secondary structure we tested. When Porter 5 is retrained on SCOPe based sets that eliminate homology between training/testing samples we obtain similar results. Porter is available as a web server and standalone program at <http://distilldeep.ucd.ie/porter/> alongside all the datasets and alignments.

From DNA repair to enzyme catalysis, proteins are the chief actors within the cell. While we discovered more than 325 million protein sequences<sup>1</sup>, we still lack a feasible method to experimentally fully characterize them at a large scale. Nonetheless, nearly 150,000 protein structures are now freely available, growing by roughly 10,000 per year, making proteins a central research topic in Bioinformatics<sup>2</sup>. One of the most enduring open problems in Bioinformatics is Secondary Structure (SS) prediction<sup>3,4</sup>. It was inaugurated by Pauling and Corey in 1951, when they predicted the existence of the two most common SS conformations –  $\alpha$ -helix and  $\beta$ -sheet – before the first protein structure was fully determined<sup>5</sup>. What followed was the first generation of SS predictors, all based on exploiting statistical propensities of single AA towards specific SS conformations<sup>6–9</sup>. The second generation of SS predictors was developed relying on information contained within segments of multiple adjacent amino acids (AA)<sup>10</sup>, physicochemical properties<sup>11</sup>, and algorithmic developments such as Neural Networks (NN)<sup>12–14</sup>, graph theory<sup>15</sup>, and nearest-neighbor methods<sup>16</sup>. Finally, the encoding of richer input including evolutionary information characterises the third generation of SS predictors, where profile-based inputs extracted from alignments of multiple homologous sequences led to accuracies exceeding 70% for the first time<sup>17</sup>. Notably, each generation of SS predictors has taken great advantage of the constantly growing availability of computational resources and data to exploit deeper information through more advanced methods<sup>3,18</sup>. Moreover, since the 90 s, NN have become the de facto standard technique to predict SS<sup>12–14,19–26</sup>, and maintain a central role at the two most important academic assessments of protein structure predictors: CASP and CAMEO<sup>27,28</sup>.

Six decades of efforts towards more accurate protein SS predictions have passed<sup>14,29</sup>. Nonetheless, the theoretical limits of prediction – set at 88–90% accuracy per AA, mainly due to the intrinsic dynamic nature of protein structure<sup>30</sup> and ambiguity of SS class assignment – have not been reached yet, and the importance of accurate SS prediction as an intermediate step towards more complex protein features, such as tertiary, or quaternary

School of Computer Science, University College Dublin, Belfield, Dublin 4, Ireland. Correspondence and requests for materials should be addressed to G.P. (email: [gianluca.pollastri@ucd.ie](mailto:gianluca.pollastri@ucd.ie))

Method	profile-less	plain profiles	deep profiles
Baseline	45.19%	60.9%	61.33%
FFNN	69.85%	80.02%	80.45%
CBRCNN	71.33%	82.32%	83.1%

**Table 1.** Performances of single models of different NN architectures on the validation set.

structure, has not diminished<sup>4,31</sup>. We start this study assessing the potential and limits of SS prediction without evolutionary information, reaching roughly 70% accuracy, similar to that of early profile-based methods<sup>17</sup>. We then assess different NN architectures, focusing on classic window-based Feed Forward NN (FFNN) and cascaded Bidirectional Recurrent Neural Networks and Convolutional Neural Networks (CBRCNN) to gauge the relative strengths of each architecture. We investigate different pipelines to harness evolutionary information extracted with two of the most common tools – PSI-BLAST<sup>32</sup> and HHblits<sup>33</sup> – and benchmark different techniques to encode evolutionary information in the form of profiles. We develop a novel input encoding which is able to represent both evolutionary information and the identity of the query sequence. Finally, we implement the best methods into Porter 5, a state-of-the-art three- and eight-state SS predictor. Porter 5 is available as a light standalone program and a simple web server, alongside training and test datasets used for this study.

## Results

We trained profile-less and profile-based models with profiles encoded in a number of different ways. We identified the most successful predictors in 5-fold cross validation experiments on the training set. We ensembled some of these models in our final predictor Porter 5, which we tested on multiple independent sets alongside a number of the most recent SS predictors.

**Alignment-free predictions.** Evolutionary information, in the form of aligned sequences, was first used to significantly improve the prediction of SS in the early 90s<sup>17</sup>. The training sets used at the time contained only a few hundred proteins. For this study we were able to build a training dataset of almost 16,000 proteins (4 million AA). Given this massive growth in sample size, we tried to gauge whether it is now possible to produce reliable predictions without the use of alignments.

In 1993, an ensemble of 2 cascaded FFNN was adopted to reach a Q3 accuracy above 70% (see Methods: Measuring performances)<sup>17</sup>. We assessed window-based FFNN adopting an incremental training approach (described in Methods:FFNN) that allowed us to reach 69.7% Q3 accuracy with no profiles and just one hidden layer. We slightly improved the same FFNN up to 69.8% and 69.9% Q3 accuracy adding 1 or 2 hidden layers, respectively, our best results with a single FFNN without evolutionary information. It should be noted that a baseline predictor that classifies each residue into the most frequent secondary structure for its type results in a Q3 of 45.2%, 5.4% better than classifying all AA as the most common class (coils), see Table 1.

To summarize, adopting a considerably larger training set but without evolutionary information, we reached comparable results to the 1993 state-of-the-art<sup>17</sup>. Using CBRCNN (see Methods:CBRCNN) instead of FFNN we observed a further increase in accuracy, up to 71.3% on the same sets. While there might be advantages to alignment-less predictions, as they require considerably less computational time with respect to profile-based solutions (fractions of seconds per protein instead of minutes), their accuracy, at ~71%, is far from the state-of-the-art predictors including evolutionary information, estimated at ~82–83%<sup>29,34</sup>.

**Profile-based predictions.** In a second phase of this study we tested different ways to encode alignments in order to maximise input information to a predictor.

We generated alignments with both PSI-BLAST<sup>32</sup> and HHblits<sup>33</sup>. We did not limit the number of hits of PSI-BLAST or HHblits, resulting in alignments with an average of ~14,000 and ~1,300 proteins, respectively (see also Methods:Evolutionary Information). We encoded evolutionary information into 22 inputs using plain profiles, as described in Methods:Input Encoding.

We trained a three hidden layer FFNN constructed similarly to the best FFNN based on single-sequence inputs. We obtained 79.9% accuracy, a 10% improvement over the alignment-less case. Fine-tuning the FFNN hyperparameters did not substantially change the results, with only slight improvements for networks with larger hidden layers, confirming that there is more information embedded in a profile than in a single sequence.

It should also be noted that, when using profiles, we obtain a Q3 accuracy of 60.9% completely disregarding the context surrounding an AA (training a FFNN with window of size 1), see Table 1.

CBRCNN performed significantly better on the same data and encoding, up to 82.3% for a single model, matching the performance of a fully tuned ensemble trained on a smaller set<sup>34</sup>.

**Deeper profiles.** We found beneficial to employ, at encoding time, a weighting scheme that aims to maximize the entropy of the profiles (see Methods:Input Encoding), i.e. that weighs more those sequences that are more informative (more different from the plain profile). This step improved the Q3 accuracy of our best FFNN and CBRCNN by 0.4% and 0.3%, respectively, while maintaining an encoding composed of 22 input numbers per AA.

We trained both FFNN and CBRCNN on several more encoding schemes (not reported), testing various options of PSI-BLAST and concatenating additional features such as protein length or the encoded sequence without a profile from alignments. We obtained the best results by adopting a simple, novel “clipping” technique (see Methods:Input Encoding) that is capable of presenting both the weighted profile from the aligned sequences and the identity of the AA in the protein itself, while keeping the encoding size unchanged. Combining this

Training	From scratch	Refining	Average	Union	Intersection	Concatenation
Q3 Accuracy	83.15%	83.41%	83.79%	83.41%	82.81%	83.77%

**Table 2.** Performances of single CBRCNN trained with different approaches relying on both PSI-BLAST and HHblits.

Training strategy	PSI-BLAST	HHblits	Concatenation	PSI-BLAST and HHblits	All the previous
Five-fold cross-validation	83.55%	83.55%	83.98%	84.18%	84.19%
Full set (Porter 5)	83.42%	83.39%	83.49%	84.13%	84.19%

**Table 3.** Assessment on the 2017\_test set of three-state ensembles trained on either five-fold cross-validation or full set.

clipping scheme and the alignment profile of maximal entropy, a single CBRCNN reached 83.1% Q3 accuracy, as reported in Table 1.

**HHblits.** As a final step to exploit evolutionary information, we adopted alignments generated by HHblits<sup>33</sup>, and compared it to PSI-BLAST<sup>32</sup>.

Although HHblits aligns considerably fewer sequences in our experimental settings (roughly a tenth of PSI-BLAST, that is ~1,300 proteins in our case), the set of hyperparameters selected for the CBRCNN trained on PSI-BLAST also worked close to optimally for training on HHblits inputs. In particular, after some tuning of the HHblits options (see Methods:Evolutionary Information), we observed a Q3 accuracy of 83.15% training CBRCNN on HHblits inputs, directly comparable with the 83.1% obtained on PSI-BLAST inputs. Refining on HHblits profiles models previously trained on PSI-BLAST gave a Q3 accuracy of 83.41%. Training a single CBRCNN on the average of PSI-BLAST and HHblits inputs improved the accuracy further, to 83.79%. We found less beneficial to train on inputs encoded from the union (83.41%) or the intersection (82.81%) of the two sets of alignments. Finally, we obtained 83.77% Q3 accuracy training on the concatenation of PSI-BLAST and HHblits profiles (44 inputs rather than 22). See Table 2 for a summary.

**Towards state-of-the-art predictor.** Finally, we built an ensemble of predictors based on the most successful individual models. All the experiments were run on five-fold cross-validation to gauge generalization performances of the ensemble<sup>35</sup>.

**Ensembling.** Bayesian model averaging is a classic ensembling approach which we exploited since the first version of Porter<sup>22</sup>: the outputs of individual models (outputs of a softmax function in our case) are simply averaged component by component. We ran preliminary testing by splitting our set into 1/5 for testing and 4/5 for training. An ensemble of the best 17 CBRCNN, with different hyperparameters but all trained on PSI-BLAST, achieved an accuracy of 84% (Table 2). Ensembles of decreasing sizes record modest reductions in performances, down to 83.82% with just 3 CBRCNN. Adding 3 structurally identical CBRCNN trained on HHblits inputs we observed a Q3 accuracy of 84.63%. We could not significantly improve on this by adding any further model trained on either HHblits or PSI-BLAST. Adding to the ensemble the single best performing CBRCNN trained on the concatenation of PSI-BLAST and HHblits inputs led to a further small increase in performances, up to 84.7% Q3.

We then tested this same ensemble of 7 models in 5-fold cross-validation, without any further tuning of hyperparameters or any change in the models selected. We obtained very similar results to our preliminary testing. The overall ensemble accuracy, averaged over the 5 folds, was 84.85%.

Finally, we trained from scratch the 7 best performing CBRCNN (selected by cross-validation, as described above) on the full training set rather than on individual training folds of the cross-validation. We then tested an ensemble of these 7 models on a completely independent set (see “2017\_test” in Methods:Datasets) containing over 3,000 proteins. We compared the accuracy of this ensemble against the ensemble of all 35 models resulting from the 5-fold cross-validation training (3 PSI-BLAST CBRCNN, 3 HHblits CBRCNN and 1 PSI-BLAST + HHblits CBRCNN for each of the 5 folds). As reported in Table 3, while there were some differences between the accuracies of individual components of these two solutions, the overall ensembles performed almost identically, hence the retrained ensemble of 7 models is preferable for the final predictor as it is computationally more compact than the ensemble of 35 models.

**Stacking and further results.** We tried many other architectural solutions during preliminary testing, including deep FFNN architectures, and structures akin to Residual Neural Networks<sup>36</sup>, in which the global inputs to the model (the profile of residue frequencies) is presented to downstream stages through shortcut connections alongside the predictions of previous stages<sup>37</sup>. While we observed small improvements when modestly increasing the number of hidden layers (up to 3–4, depending on the precise configuration), the results we obtained were generally poorer than those we observed with CBRCNN - typically around 1.5% worse than individual CBRCNN of similar size, and approximately 2% worse than those of a stack of 2 CBRCNN, which is what we used in our final predictors. While it is not entirely clear why, it appears that the recurrent stages in the CBRCNN are more efficient at capturing the sequential dynamics of our inputs than those of feed-forward networks alone, possibly because of their unrestricted input size. We did observe only marginal improvements in performances (roughly

Method	Q3	SOV'99	SOV_refine	Q8	SOV8'99	SOV8_refine
<b>Porter 5</b>	<b>84.19%</b>	<b>81.19%</b>	<b>76.72%</b>	<b>73.02%</b>	<b>69.91%</b>	<b>72.09%</b>
Porter 5 (HHblits and PSI-BLAST)	83.49%	80.17%	75.64%	71.94%	69.03%	71.45%
Porter 5 (PSI-BLAST only)	83.42%	80.41%	75.8%	72.11%	69.28%	71.56%
Porter 5 (HHblits only)	83.39%	80.19%	75.59%	71.8%	68.87%	71.16%
SSpro 5.1 with templates	82.62%	79%	74.58%	71.91%	68.68%	70.72%
PSIPRED 4.01	82.06%	77.83%	72.95%	N.A.	N.A.	N.A.
RaptorX-Property	82.04%	78.57%	73.66%	70.74%	67.59%	69.65%
Porter 4	82%	78.85%	73.89%	N.A.	N.A.	N.A.
DeepCNF	81%	76.96%	71.84%	69.76%	66.42%	68.5%
SSpro 5.1 ab initio	80.7%	76.85%	72%	68.85%	65.33%	67.54%

**Table 4.** Q3/Q8 accuracy and SOV score per AA on the full test set.

Method	Q3 per AA	SOV'99 per AA	SOV_refine per AA	Q3 per protein	SOV'99 per protein	SOV_refine per protein
<b>Porter 5</b>	<b>83.81%</b>	<b>80.41%</b>	<b>75.73%</b>	<b>84.32%</b>	<b>81.05%</b>	<b>76.45%</b>
Spider3	83.15%	79.43%	74.68%	83.42%	79.79%	75.07%
Porter 5 (HHblits only)	83.06%	79.49%	74.71%	83.68%	80.26%	75.58%
SSpro 5.1 with templates	82.58%	78.54%	74.02%	83.94%	80.29%	76.15%
PSIPRED 4.01	81.88%	77.36%	72.33%	82.48%	78.22%	73.31%
RaptorX-Property	81.86%	78.08%	72.99%	82.57%	78.99%	74.03%
Porter 4	81.66%	78.05%	72.89%	82.29%	78.61%	73.55%
SSpro 5.1 ab initio	81.17%	76.87%	72.03%	81.1%	76.92%	72.12%
DeepCNF	81.04%	76.74%	71.47%	81.16%	76.99%	71.7%

**Table 5.** Performances on the smaller 2017\_test set for which Spider3 generates predictions, sorted by Q3 accuracy.

+0.1%) when stacking more than 2 CBRCNN stages with shortcut connections, and decided against including these more complex models into our final testing and predictor.

**Eight-state prediction.** We applied the same pipeline described in Ensembling (section above) to the prediction of the full DSSP 8-class definition of SS<sup>38</sup>. It should be noted that this slightly increases the total number of tunable parameters of the CBRCNN with respect to three-state SS prediction. In particular, we applied Bayesian model averaging on an equal number of CBRCNN trained on either PSI-BLAST or HHblits inputs, and some trained on concatenated inputs (as in Table 2). We obtained 71.76%, 71.66% and 72.29% Q8 accuracy training single CBRCNN on 4/5 of the training set on PSI-BLAST, HHblits and concatenated inputs, respectively. An ensemble of 3 CBRCNN trained on PSI-BLAST inputs yields 72.47% Q8 accuracy on the same fold. When we add to the input of these networks the output of the ensemble of the 3 corresponding (PSI-BLAST) models trained on the 3-class problem we record a further improvement, to 72.79% Q8 (+0.3%), without dramatically increasing the encoding size (total of 25 inputs). We extended this approach to the 3 HHblits-trained models and to the one trained on concatenated PSI-BLAST and HHblits profiles.

The overall ensemble of 7 models trained on the full training set achieves 73.02% Q8 accuracy on the 2017\_test set described in Methods:Dataset (see also Table 4). An ensemble of the same 7 CBRCNN without the three-state predictions as inputs has an accuracy of 72.11%, confirming that including these predictions is beneficial.

**Assessment of multiple predictors on independent test set.** Porter 5 is an ensemble of 7 CBRCNN (see Ensembling, above): 3 trained on PSI-BLAST, 3 trained on HHblits and 1 trained on both (44 inputs rather than 22). Porter 5 relies on 7 more CBRCNN to predict eight-state SS (see Eight-state prediction, above). We tested Porter 5 against Porter 4<sup>34</sup>, Spider3<sup>26</sup>, SSpro 5.1<sup>24</sup>, PSIPRED 4.01<sup>20</sup>, RaptorX-Property<sup>39</sup> and DeepCNF<sup>25</sup> on the 2017\_test set we created, containing 3,154 proteins. Spider3 rejects proteins containing undetermined (X) amino acids (562 overall) and, when we use the parameters required by Spider3, either PSI-BLAST or HHblits do not return a valid result for 129 proteins. Because of this we report results on two sets: one where we exclude the proteins on which we could not obtain a valid response from Spider3 (2,463 entries composed by 497,142AA) (Table 5); the full set of 3,154 proteins (Table 4) on which Spider3 is not assessed.

Porter 5 is the most accurate 3-state and 8-state predictor in our tests on the 2017\_test set with 3-class accuracy of 83.8% on the smaller version of the set and 84.2% on the larger one, 0.7% better than Spider3, 1.2–1.6% better than SSpro 5.1 with templates, and at least 2% more accurate than all the other predictors.

Performances of the servers show very similar deviations and differences greater than approximately 0.12% in Table 4 and 0.14% in Table 5 are significant at  $p = 0.05$ . Porter 5 is also very fast given the small size of its models (on average 39k parameters for the 3-class networks, 58k for 8 classes). Once the alignments by PSI-BLAST and HHblits are present, Porter 5 runs 2 orders of magnitude faster than Spider3.

Method	Q3	SOV <sup>99</sup>	SOV_refine	Q8	SOV8 <sup>99</sup>	SOV8_refine
CAMEO	85.48%	82.08%	78.08%	74.99%	72.36%	74.81%
CASP13	82.99%	78.36%	73.39%	71.08%	66.95%	69.27%

**Table 6.** Assessment of Porter 5 on CASP13, i.e. 43 targets, and on the last 6 months of CAMEO, i.e. 463 proteins released from Dec 28, 2018 to Jun 22, 2019.

Method	Q3	SOV <sup>99</sup>	SOV_refine	Q8	SOV <sup>99</sup>	SOV_refine
NMR	81.61%	76.64%	70.55%	67.52%	62.41%	63.86%
X-ray	<b>84.65%</b>	<b>81.99%</b>	<b>77.81%</b>	74%	<b>71.24%</b>	<b>73.54%</b>

**Table 7.** Porter 5 on NMR vs X-ray crystallography proteins.

We also measured the SOV<sup>99</sup><sup>40</sup> and the SOV\_refine<sup>41</sup> (see Methods:Measuring performances) of every SS predictor on both versions of the 2017\_test set. Porter 5 is consistently the best-performing 3-state and 8-state SS predictor, with both SOV scores at least 1% and 2% better than any other SS predictor on small and large versions of the set, respectively. Porter 5 is also 1.2% better than any other predictor considering the 8-state SOV<sup>99</sup> and the SOV\_refine scores.

Finally, we measured Porter 5 performances on the CASP13<sup>28</sup> set, and on 6 months of proteins (December 28 2018 to June 22 2019) released by CAMEO<sup>27</sup>. The results are reported in Table 6 and roughly confirm the Porter 5 results we obtained on the 2017\_test set.

**Nuclear magnetic resonance.** We also analyzed the performance of Porter 5 separately on proteins resolved by Nuclear Magnetic Resonance (NMR) and by X-ray crystallography. NMR proteins are predicted at a significantly lower Q3 accuracy (81.6%,  $\sigma = 0.12\%$ ), possibly because of their different statistics (e.g. average length and composition) or less certain determination of SS. The X-ray only section of the 2017\_test set, which is roughly 90% of the total, is predicted at an average Q3 of 84.65% (Table 7).

**Porter 5 with SCOP based redundancy reduction protocol.** While redundancy reduction protocols similar to the one we adopted to build our sets are widely used<sup>17,21,22,26,34,39,42–44</sup>, this type of redundancy reduction does not fully eliminate the occurrence of proteins with similar 3D structures (hence similar SS) in the training and testing sets<sup>20,45</sup>. Normally, the only way to genuinely control for this and produce sets that are completely devoid of structurally homologous examples is to resort to classifications of protein structures such as SCOPe/ASTRAL<sup>46</sup> and use information gleaned from these to guide the construction of the data sets, e.g. by selecting only one representative per superfamily or family of proteins. The drawback of this procedure is that the resulting sets will be smaller, and it has been shown in different occasions (e.g.<sup>34</sup>) that, all other factors being the same (e.g. same algorithms, same redundancy reduction protocols) larger data sets lead to improvements in performances. In order to gauge the effect of stricter redundancy reduction criteria on the methods presented here, we retrained 2 separate versions of Porter 5 using the JPred4 sets<sup>45</sup>. In these sets, only one representative for each of the 1,358 SCOPe/ASTRAL v.2.04<sup>46</sup> superfamily domain sequences is selected for the training set, while a further 150 proteins from superfamilies not included in the training set are used as a blind test set. The first version we retrained uses the exact same protocol and ensemble as Porter 5, including recent versions of the UniRef database for the creation of MSA and a combination of alignments by PSI-BLAST and HHblits, but it is trained on the 1,348 JPred4 set. In the second version we also adopted the same alignments used by JPred4, based on release 2014\_7 of UniRef90 and obtained using PSI-BLAST, which are available from the JPred4 web site. It should be noted that in this case, given that we do not use HHblits alignments, all the models in the ensemble are trained solely on PSI-BLAST profiles. The first version, adopting recent alignments, achieves 84.62% correct prediction on the JPred4 blind set. The second version, which relies on the exact same training and testing data as JPred4, achieves 83.62% correct prediction. JPred4, which is based on a standard feed-forward neural network architecture, has a 82.29% Q3 per amino acid on the same sets. While the testing set is small (150 proteins), these results suggest that more sophisticated machine learning algorithms (and, indeed, more up to date alignment sets and treatment thereof) may be beneficial to predictive performances. These results also roughly match what we found on our larger data sets, although it should be noted that the class definition in these sets is slightly different in that DSSP class ‘G’ is assigned to Coil rather than to Helix. This different assignment has been shown in the past to lead to somewhat higher Q3 values, e.g. in<sup>47</sup>, which might explain how Porter achieves a similar Q3 when trained on a set which is an order of magnitude smaller than its original training set.

We also assessed Jpred4 on the 2019\_test set (see Table 8 and description of the set in the following section) with classes recast to match the Jpred4 class assignment. In this case the more modern predictors including Porter 5 show Q3 3.7–4.7% higher than Jpred4 and similar improvements in SOV, suggesting that larger training sets, alongside larger alignment sets and more sophisticated algorithms, may be beneficial.

**Assessment of latest SS predictors.** In a separate test set we assessed some very recent predictors which have been trained on sets more recent than our 2017\_test set, i.e. MUFOLD-SS<sup>43</sup>, NetSurfP-2.0<sup>48</sup> and SPOT-1D<sup>44</sup>. Because of this, we generated a second independent test set (also see Methods:Datasets) starting from June 24 2019 PDB proteins, 25% redundancy reduced against the Porter 5, NetSurf-2.0 and SPOT-1D training sets (we



Method	Q3	SOV <sup>99</sup>	SOV <sub>refine</sub>	Q8	SOV8 <sup>99</sup>	SOV8 <sub>refine</sub>
SPOT-1D	82.13%	76.65%	71.37%	69.69%	65.52%	67.18%
<b>Porter 5</b>	<b>81.74%</b>	<b>76.67%</b>	<b>71.03%</b>	<b>68.25%</b>	<b>63.24%</b>	<b>64.87%</b>
NetSurfP-2.0	81.3%	75.64%	70.3%	67.93%	62.77%	64.66%
MUFOLD-SS	81.09%	75.28%	69.87%	68.21%	64.3%	66.33%
<i>Jpred4</i>	77.38%	72.29%	64.96%	N.A.	N.A.	N.A.

**Table 8.** Most recent predictors and *Jpred4* assessed on 2019\_test set of 618 proteins.

3-states	8-states	Training Set		2017_test Set		2019_test Set	
Helices	G	38%	135,498	39.22%	21,404	37.41%	2,300
	H		1,306,610		233,961		31,854
	I		714		177		33
Sheets	E	22.15%	800,297	20.9%	129,425	17.87%	15,411
	B		41,026		6,793		916
Coils	C	39.85%	764,391	39.88%	133,183	44.72%	21,605
	S		331,075		57,678		9,804
	T		417,815		68,973		9,452

**Table 9.** Overview of AA composition of Training, 2017\_test and 2019\_test.

could not access the MUFOLD-SS training set). As either MUFOLD-SS or SPOT-1D or both did not produce a valid prediction for 243 out of the 861 original proteins in the set, Table 8 shows the performances observed on the 618 proteins successfully predicted by all predictors of this group. The 3 predictors have similar performances, with Porter 5 slightly outperforming both MUFOLD-SS and NetSurfP-2.0, but being slightly outperformed by SPOT-1D. Differences of 0.28% in Q3 and 0.37% in Q8 in the table are significant at  $p = 0.05$ . It should be noted that SPOT-1D relies on the predictions of SPOT-Contact<sup>49</sup>, i.e. a Contact Map predictor<sup>4</sup>, which in turn requires Spider3<sup>26</sup>, CCMpred<sup>50</sup>, and DCA<sup>51</sup>. This results in a far more computationally intense pipeline which in essence derives the SS from a guess of a protein's 3D structure through its contact map<sup>4</sup>.

## Discussion

In this study we describe the development of a new, state-of-the-art SS predictor, Porter 5<sup>52</sup>. We trained window-based feed-forward neural networks with different hyperparameters and input encoding (see Table 1 to define our baselines and assess the quality of our large training set (see Table 9). We developed both a state-of-the-art model, and a novel encoding technique, i.e. “clipping”. We assembled the final predictor Porter 5 as a simple ensemble of models trained on different inputs, i.e. either PSI-BLAST<sup>32</sup> or HHblits<sup>33</sup> or a concatenation of both. We applied a very similar approach to the harder eight-state SS prediction problem to develop the eight-state version of Porter 5 which represents, analogously to the three-state Porter 5, the state-of-the-art for this task (see Table 4). Porter 4, the previous release of Porter, was trained on 7,522 proteins<sup>34</sup>. Thanks to the constant growth of the PDB<sup>2</sup>, we performed all the experiments on a training set twice as large as the one adopted for Porter 4, i.e. 15,753 proteins (see Methods:Datasets). The results we present in this study confirm the continuing positive contribution of a larger, well-distributed training set.

For this study, we exploited evolutionary information through different encodings and gauged their importance with respect to an encoding containing only the plain protein sequence. While we can now predict SS using plain protein sequences at an accuracy that would have represented the state-of-the-art including evolutionary information 25 years ago, we observed that evolutionary information is as important as ever, boosting prediction accuracies by 10% or more. In particular we used evolutionary information mined by both PSI-BLAST<sup>32</sup> and HHblits<sup>33</sup> and observed that while they lead to broadly similar predictive accuracies when used individually, their combination is clearly beneficial. To the best of our knowledge, Porter 5 is the first SS predictor to ensemble models trained on PSI-BLAST or HHblits, which (empirically) appears to be the most effective way to exploit both algorithms at the same time (see Table 2).

We have also studied a number of different models, confirming that recurrent neural network architectures are particularly effective at SS prediction, with a combination of bidirectional recurrent networks and dense convolutional layers being the best performing model. While a modest increase in the number of stages adopted worked well for us, we did not observe improvements in performances beyond 3–4 internal layers for feed-forward networks and 2 stages of BRNN-CNN stacks. This seems to suggest that, at least given the current sizes of training sets, recurrent neural network stages capture all the long-range information that can be exploited effectively.

Unlike many other modern “deep” predictors, Porter 5's models are individually tuned to have roughly correct individual expressive power rather than being oversized in the first place and kept to the right capacity by regularization techniques or dropout. Individual models within Porter 5 have 40,000–60,000 free parameters. This is significantly less than the average 500,000 parameters of DeepCNF<sup>25</sup>, i.e. the PSI-BLAST version of RaptorX-Property<sup>39</sup>, or the well over one-million of Spider3<sup>26</sup>, although it is still a 2–3 fold increase with respect to the 13,000–18,000 free parameters of Porter 4<sup>34</sup>. The relative small size of Porter 5 also means that, once alignments are available, individual predictions are extremely fast to run.

We assessed Porter 5 on a first independent test set (2017\_test), along with some of the SS predictors trained up to 2017: DeepCNP<sup>25</sup>, Porter 4<sup>34</sup>, PSIPRED 4.01<sup>20</sup>, RaptorX-Property<sup>39</sup>, Spider3<sup>26</sup> and both versions of SSpro 5.1<sup>24</sup>, i.e. profile-based and template-based. In all our tests Porter 5 outperformed the other methods, often by large margins with an accuracy of approximately 84% for 3-class SS prediction and 73% for 8-class prediction. It should also be noted that our assessment might be somewhat optimistic for some of the competing predictors since we did not perform any redundancy reduction of our final test set against their training sets<sup>35</sup>.

Finally, we assessed Porter 5 against some of the most recent predictors which have also been trained on very recent and large training sets, i.e. MUFOLD-SS<sup>43</sup>, NetSurfP-2.0<sup>48</sup> and SPOT-1D<sup>44</sup>. As these predictors' training sets overlapped with our original test set, we generated a second smaller independent test set (2019\_test) based on PDB sequences uploaded up to June 24th. In this case we observed results which are broadly similar between these newer predictors, with Porter 5 slightly outperforming both MUFOLD-SS and NetSurfP-2.0, but slightly outperformed by SPOT-1D which, however, is built on a more complex and computationally intensive (though highly effective) pipeline in which the SS is predicted through a protein's contact map.

## Methods

**Datasets.** The selection and preparation of datasets to adopt has a central role in any machine learning method<sup>35</sup>. We built our datasets from the Protein Data Bank (PDB)<sup>2</sup>, the public repository of all the freely and publicly known protein structures. We assembled our final datasets only with proteins sharing up to 25% sequence identity<sup>35</sup>. Specifically, we built our training set from the PDB released on Dec 11 2014, internally redundancy-reduced at a 25% identity threshold. We also built an independent test set (2017\_test) from the PDB released after Dec 11 2014 and up to Jun 14, 2017. We redundancy-reduced this set at a 25% identity threshold against the training set. Further, we internally redundancy reduced the resulting set at a 25% identity threshold. Finally, we removed all proteins with at least 10 consecutive undetermined AA from both sets. The training set contains 15,753 proteins (3,797,426 AA) and 2017\_test 3,154 proteins (651,594 AA), among the largest ever used to build a SS predictor. The SS states were assigned according to the Dictionary of Protein SS (DSSP)<sup>38</sup> and their distribution is highlighted in Table 9. In different tests the training set is used as a whole for training purposes or split into 5 randomly distributed folds in cross-validation for hyperparameter optimization<sup>35</sup>. The 2017\_test set is only used in the final part of this study to evaluate our final solutions and other solutions previously published. The training and the test sets are available at <http://distilldeep.ucd.ie/porter/>.

We also curated an additional independent test set (2019\_test) to fairly compare Porter 5 against some of the most recent SS predictors, i.e. MUFOLD-SS<sup>43</sup>, NetSurfP-2.0<sup>48</sup> and SPOT-1D<sup>44</sup>, which have been trained on sets overlapping with our 2017\_test set. We removed any protein shorter than 30 AA or containing more than 10% of undetermined AA from the PDB proteins deposited up to Jun 24 2019. We then redundancy-reduced this set against the training sets of SPOT-1D, NetSurfP-2.0 and our training set at 25% identity threshold. Finally, we reduced the internal redundancy of this set at a 25% sequence identity threshold and obtained 861 proteins. As MUFOLD-SS or SPOT-1D or both do not return a valid answer for 243 of these proteins, we report results on 618 proteins, comprising 91,375 amino acids (2019\_test).

**Evolutionary information.** A key aspect of any modern SS predictor is harnessing evolutionary information<sup>53</sup>. PSI-BLAST<sup>32</sup> and more recently HHblits<sup>33</sup> are methods widely used for the purpose – i.e. gathering known protein sequences which are likely to be evolutionarily related to the protein of interest<sup>54</sup>. We relied on both, finding the best results with the default settings and iterating them 3 times with an e-value of 0.001<sup>55</sup> without limiting the number of sequence hits. PSI-BLAST is run on the May, 2016 version of UniRef90<sup>1</sup>, containing almost forty-two millions clusters. HHblits is run on the February, 2016 version of UniProt20, containing over eight millions clusters. Our experiments show similar results when a model is trained with either PSI-BLAST or HHblits, but significant improvements when both are used (see Results:HHblits).

**Input encoding.** Among the several encoding schemes assessed, we focused on three approaches: alignment-free, plain profiles and weighted profiles.

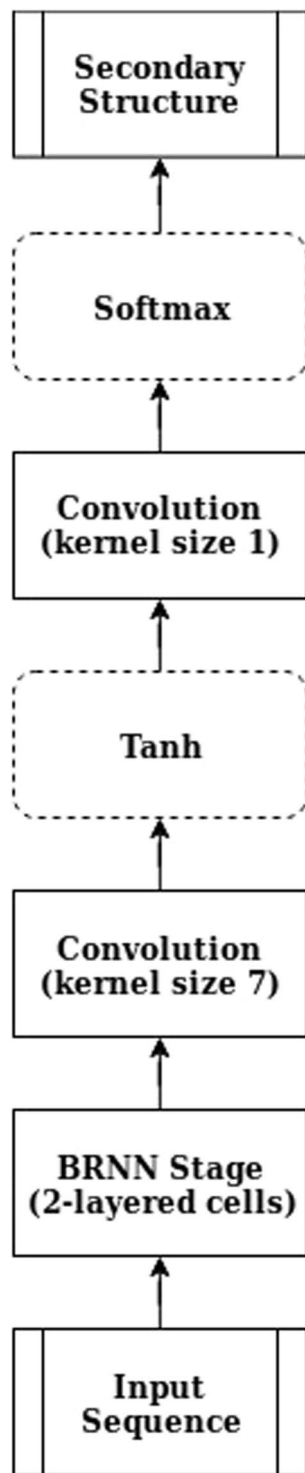
For the alignment-free case, when no evolutionary information is employed, we adopted a simple one-hot encoding of 20 positions - one for each standard AA - and a zero vector for non standard AA, i.e. “B”, “J”, “O”, “U”, “Z”, and “X”.

For the plain profiles case, our baseline for employing evolutionary information, we adopted arrays of 22 positions composed of 20 frequencies for standard AA, 1 for unknown or non-standard and the last position for gaps. The first 21 numbers are normalized to add up to 1 without considering gaps, while the 22nd number represents the total frequency of gaps in a column of the alignment.

For the weighted profiles case, we maximized the entropy deriving from the evolutionary information applying a weighting scheme to the plain profiles<sup>56,57</sup>. In particular, we calculated the weight of each sequence in the alignment as:

$$W_{seq} = \sum_{n=1}^{length} -\log f[aa_{seq}(n)]$$

where  $f[aa_{seq}(n)]$  is the relative frequency of the n-th AA of sequence  $seq$  within column  $n$  of the alignment. We then weighted every sequence  $seq$  in the alignment by  $W_{seq}$  and, finally, normalized as for the plain profiles case, i.e. the first 21 components add up to 1 and the 22nd is normalized independently. Differently from plain profiles, we did not consider external gaps when calculating the gap frequency.



**Figure 1.** Diagram of the BRCNN. The input sequence is processed by three stages, i.e. one BRNN and two CNN stages, in order to predict the SS. The final architecture of Porter 5 - the CBRCNN - is the (two) cascaded version of the above.

Clipping is the novel encoding method we introduce in this study. The simple idea is to set to 1 the position in the profile vector associated to the AA in the query sequence, regardless of its frequency in the alignment. This approach can be seen as a merging technique between one-hot encoding – adopted when evolutionary information is lacking – and any method to represent evolutionary information. It should also be noted that no information in the profile is lost when adopting clipping, as any one of the 21 numbers in the profile is equal to 1 minus the sum of the others.



Input	3-state				8-state			
	PSI-BLAST or HHblits			Concatenated	PSI-BLAST or HHblits			Concatenated
NF/B	25	30	30	25	30	35	36	30
NHF/B	40	40	45	40	45	55	60	45
NHY	50	50	55	50	50	45	48	50
CoF/B	3	3	3	3	3	3	3	3
Cseg	10	10	10	10	10	10	10	10
Cwin	7	7	7	7	7	7	7	7

**Table 10.** The hyperparameters of the models employed for Porter 5. A total of 7 models are ensembled in both 3- and 8-state component. The PSI-BLAST or HHblits only models share the same hyperparameters.

We also build a version of Porter 5 which predicts the eight-state SS classes by the DSSP program<sup>38</sup>. In this case the output of the three-state Porter 5 is concatenated to the input – i.e. 25 inputs rather than 22, as in the three-state Porter 5.

**Feedforward neural networks.** We defined our baselines implementing window-based FFNN of up to 7 hidden-layers. The symmetric input-window allows segments of AA composed by an odd number of AA, centered on the current  $n$ -th position. More in detail, the input at  $n$ -th time step is defined as  $I(n) = \nu_{n-l}, \nu_{n-l+1}, \nu_{n-l+2}, \dots, \nu_{n+l}$  where  $\nu_n$  is the  $n$ -th encoded input and  $l$  is the number of right- and left-adjacent AA considered as additional contextual information, i.e. the input-window contains  $l*2 + 1$  AA at any position.

We trained one-hidden-layer FFNN increasing the number of hidden units – to verify whether we had sufficient data to approximate the mapping function (from AA to SS)<sup>58</sup> – and then trained deeper solutions – i.e. increasing the number of hidden layers. To reduce the computational costs of the hyperparameter search, we adopted an incremental training technique. More in detail, we continued the trainings until completion, then substituted the top layer, i.e. the softmax layer, with untrained hidden-layer + softmax layers and trained these alone, leaving all the weights upstream of them untouched. Finally, we briefly refined the whole FFNN, training every hidden-layer – i.e. end to end, and iterated the process.

**Cascaded bidirectional recurrent and convolutional neural networks.** The CBRCNN, assessed in this study and at the core of Porter 5, is an additional refinement of the two-stage bidirectional recurrent neural network (BRNN) initially implemented for the first release of Porter<sup>22</sup> and successively exploited to predict several more protein structure annotations – e.g. relative solvent accessibility, torsion angles and contact density<sup>59–61</sup>. The CBRCNN preserves two cascaded stages, both containing a BRNN layer with two-layered recurrent cells, and introduces convolutional layers downstream of the BRNN to process windows of both forward and backward chain memories. Differently from the window-based FFNN, the CBRCNN fetches one input/AA at time but then elaborates the entire protein into two Markovian chains (of the BRNN), before processing windows of them through the convolutional layers (see Fig. 1).

In particular, a BRNN (with independent weights and one hidden layer) is followed by a 1D convolutional layer with kernel size greater than one – i.e. able to look at different time steps of the two preceding chain states –, then by a further convolutional layer of kernel size one with softmax outputs. Equivalently, the two convolutional stages can be thought of as a single map implemented by a two-layered network. The output of this overall network is then fed to a similar network for the second stage. The main differences between the first and the second stage are the network size and input: all the layer sizes in the second stage are half the size of those in the first stage, and the output of the first stage network is averaged in different segments to feed the second stage. In other words, the second stage CBRCNN learns to associate every target with a given number of segments, which are built averaging the output of the first stage CBRCNN.

We fixed the number of time steps seen by the first convolutional layer (i.e. the kernel size) to 7 – i.e. 3 adjacent steps per side plus the one at a given position –, and the number of segments of the second stage and their size to 15 and 21, respectively. Therefore, the second stage processes 15 windows, each containing the average of the first stage predictions over 21 time steps, for a total of 315 adjacent steps processed per prediction.

The number of hyperparameters to set in a BRNN, and the more sophisticated internal dynamics, makes this architecture a more complex neural network to train and tune with respect to a FFNN. Step by step, the memory size for the recurrent networks (NF/B), the hidden layer sizes for the recurrent networks (NHF/B) and for the layer preceding the softmax (NHY), in addition to the number of time steps seen by the convolutional layer (CoF/B) and the number and size of the segments feeding the second stage (Cseg and Cwin), have to be determined. The values for these hyperparameters in the models used within Porter 5 are reported in Table 10.

**Ensembling.** In all cases we ensembled models by simply taking the average of their class (softmax) outputs. In preliminary tests we briefly assessed more complex strategies, e.g. Bayesian Model Combination<sup>62,63</sup> in which model-specific weights are learned, but did not find evidence that they performed significantly better than the simple average.

**Measuring performances.** The two most commonly used measures to assess SS predictors, accuracy and SOV, have been employed in this study. Accuracy is simply the fraction of AA whose predicted SS class is the same as the observed class, as determined by DSSP<sup>38</sup>. For the 3-class problem (helix, sheet, and coil) we call this Q3 accuracy. For the 8-class problem ( $\alpha$ -helix,  $3_{10}$ -helix,  $\pi$ -helix,  $\beta$ -sheet, extended strand, hydrogen bonded turn, bend, and other) we call this Q8 accuracy. The 3 classes in the 3-class problem are obtained by merging DSSP-assigned  $\alpha$ -helix,  $3_{10}$ -helix and  $\pi$ -helix into class helix,  $\beta$ -sheet and extended strand into sheet, and the rest into coil.

We also measured the Segment Overlap (SOV) between the predicted SS and the true one. This latter measure is meant to evaluate the prediction from a more biological viewpoint considering segments rather than single AA as the relevant prediction units. We measured both SOV<sup>99</sup><sup>40</sup> and SOV\_refine<sup>41</sup>.

**Optimization.** We implemented momentum<sup>64</sup> and a dynamic adaptive learning rate to optimize the training process, along with standard stochastic gradient descent (SGD)<sup>65</sup>. We set momentum to 0.9 and divided by two the learning rate any time that the cross entropy error on training set had not decreased for 100 epochs. The training set is shuffled at the end of each epoch, while the size of a mini-batch is set to ~10 proteins, that is, the network weights are updated during training after estimating the gradient on ~10 proteins at a time.

## Data Availability

Porter 5 is available as a web server and light standalone program at <http://distilldeep.ucd.ie/porter/> alongside with all the datasets and alignments.

## References

1. The UniProt Consortium. UniProt: the universal protein knowledgebase. *Nucleic Acids Res.* **45**, D158–D169, <https://doi.org/10.1093/nar/gkw1099> (2016).
2. Berman, H. M. *et al.* The Protein Data Bank. *Nucleic Acids Res.* **28**, 235–242 (2000).
3. Rost, B. Review: Protein Secondary Structure Prediction Continues to Rise. *J. Struct. Biol.* **134**, 204–218, <https://doi.org/10.1006/jtbi.2001.4336> (2001).
4. Torrisi, M. & Pollastri, G. Protein Structure Annotations. In Shaik, N. A., Hakeem, K. R., Banaganapalli, B. & Elango, R. (eds) *Essentials of Bioinformatics, Volume I: Understanding Bioinformatics: Genes to Proteins*, 201–234 (Springer International Publishing, Cham, 2019), [https://doi.org/10.1007/978-3-030-02634-9\\_10](https://doi.org/10.1007/978-3-030-02634-9_10).
5. Pauling, L. & Corey, R. B. Configurations of Polypeptide Chains With Favored Orientations Around Single Bonds. *Proc. Natl. Acad. Sci. United States Am.* **37**, 729–740 (1951).
6. Szent-Gyorgyi, A. G. & Cohen, C. Role of Proline in Polypeptide Chain Configuration of Proteins. *Science* **126**, 697–698, <https://doi.org/10.1126/science.126.3276.697> (1957).
7. Davies, D. R. A correlation between amino acid composition and protein structure. *J. Mol. Biol.* **9**, 605–609, [https://doi.org/10.1016/S0022-2836\(64\)80232-1](https://doi.org/10.1016/S0022-2836(64)80232-1) (1964).
8. Lim, V. I. Structural principles of the globular organization of protein chains. A stereochemical theory of globular protein secondary structure. *J. Mol. Biol.* **88**, 857–872, [https://doi.org/10.1016/0022-2836\(74\)90404-5](https://doi.org/10.1016/0022-2836(74)90404-5) (1974).
9. Kabsch, W. & Sander, C. How good are predictions of protein secondary structure? *FEBS Lett.* **155**, 179–182, [https://doi.org/10.1016/0014-5793\(82\)80597-8](https://doi.org/10.1016/0014-5793(82)80597-8) (1983).
10. Garnier, J., Osguthorpe, D. J. & Robson, B. Analysis of the accuracy and implications of simple methods for predicting the secondary structure of globular proteins. *J. Mol. Biol.* **120**, 97–120, [https://doi.org/10.1016/0022-2836\(78\)90297-8](https://doi.org/10.1016/0022-2836(78)90297-8) (1978).
11. Pitsyn, O. B. & Finkelstein, A. V. Theory of protein secondary structure and algorithm of its prediction. *Biopolymers* **22**, 15–25, <https://doi.org/10.1002/bip.360220105> (1983).
12. Qian, N. & Sejnowski, T. J. Predicting the secondary structure of globular proteins using neural network models. *J. Mol. Biol.* **202**, 865–884, [https://doi.org/10.1016/0022-2836\(88\)90564-5](https://doi.org/10.1016/0022-2836(88)90564-5) (1988).
13. Kneller, D. G., Cohen, F. E. & Langridge, R. Improvements in protein secondary structure prediction by an enhanced neural network. *J. Mol. Biol.* **214**, 171–182, [https://doi.org/10.1016/0022-2836\(90\)90154-E](https://doi.org/10.1016/0022-2836(90)90154-E) (1990).
14. Holley, L. H. & Karplus, M. Protein secondary structure prediction with a neural network. *Proc. Natl. Acad. Sci. United States Am.* **86**, 152–156 (1989).
15. Mitchell, E. M., Artymiuk, P. J., Rice, D. W. & Willett, P. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. *J. Mol. Biol.* **212**, 151–166, [https://doi.org/10.1016/0022-2836\(90\)90312-A](https://doi.org/10.1016/0022-2836(90)90312-A) (1990).
16. Yi, T. M. & Lander, E. S. Protein secondary structure prediction using nearest-neighbor methods. *J. Mol. Biol.* **232**, 1117–1129, <https://doi.org/10.1006/jmbi.1993.1464> (1993).
17. Rost, B. & Sander, C. Prediction of Protein Secondary Structure at Better than 70% Accuracy. *J. Mol. Biol.* **232**, 584–599, <https://doi.org/10.1006/jmbi.1993.1413> (1993).
18. Przybylski, D. & Rost, B. Alignments grow, secondary structure prediction improves. *Proteins* **46**, 197–205 (2002).
19. Rost, B., Sander, C. & Schneider, R. Redefining the goals of protein secondary structure prediction. *J. Mol. Biol.* **235**, 13–26 (1994).
20. Jones, D. T. Protein secondary structure prediction based on position-specific scoring matrices I edited by Von Heijne, G. *J. Mol. Biol.* **292**, 195–202, <https://doi.org/10.1006/jmbi.1999.3091> (1999).
21. Baldi, P., Brunak, S., Frasconi, P., Soda, G. & Pollastri, G. Exploiting the past and the future in protein secondary structure prediction. *Bioinformatics* **15**, 937–946, <https://doi.org/10.1093/bioinformatics/15.11.937> (1999).
22. Pollastri, G. & McLysaght, A. Porter: a new, accurate server for protein secondary structure prediction. *Bioinformatics* **21**, 1719–1720, <https://doi.org/10.1093/bioinformatics/bti203> (2005).
23. Buchan, D. W. A. *et al.* Protein annotation and modelling servers at University College London. *Nucleic Acids Res.* **38**, W563–W568, <https://doi.org/10.1093/nar/gkq427> (2010).
24. Magnan, C. N. & Baldi, P. SSpro/ACCpro 5: almost perfect prediction of protein secondary structure and relative solvent accessibility using profiles, machine learning and structural similarity. *Bioinformatics* **30**, 2592–2597, <https://doi.org/10.1093/bioinformatics/btu352> (2014).
25. Wang, S., Peng, J., Ma, J. & Xu, J. Protein Secondary Structure Prediction Using Deep Convolutional Neural Fields. *Sci. Reports* **6**, <https://doi.org/10.1038/srep18962> (2016).
26. 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. *Bioinforma. (Oxford, England)* **33**, 2842–2849, <https://doi.org/10.1093/bioinformatics/btx218> (2017).
27. Haas, J. *et al.* Continuous Automated Model EvaluatiOn (CAMEO) complementing the critical assessment of structure prediction in CASP12. *Proteins: Struct. Funct. Bioinforma.* **86**, 387–398, <https://doi.org/10.1002/prot.25431>.

28. Schaarschmidt, J., Monastyrskyy, B., Kryshtafovych, A. & Bonvin, A. M. J. J. Assessment of contact predictions in CASP12: co-evolution and deep learning coming of age. *Proteins: Struct. Funct. Bioinforma.* <https://doi.org/10.1002/prot.25407> (2017).
29. Yang, Y. *et al.* Sixty-five years of the long march in protein secondary structure prediction: the final stretch? *Briefings Bioinforma.* <https://doi.org/10.1093/bib/bbw129> (2016).
30. Martin, J. *et al.* Protein secondary structure assignment revisited: a detailed analysis of different assignment methods. *BMC Struct. Biol.* **5**, 17, <https://doi.org/10.1186/1472-6807-5-17> (2005).
31. Rost, B. & Sander, C. Conservation and prediction of solvent accessibility in protein families. *Proteins: Struct. Funct. Bioinforma.* **20**, 216–226, <https://doi.org/10.1002/prot.340200303> (1994).
32. Altschul, S. F. *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res.* **25**, 3389–3402, <https://doi.org/10.1093/nar/25.17.3389> (1997).
33. Remmert, M., Biegert, A., Hauser, A. & Söding, J. HHblits: lightning-fast iterative protein sequence searching by HMM-HMM alignment. *Nat. Methods* **9**, 173–175, <https://doi.org/10.1038/nmeth.1818> (2012).
34. Mirabello, C. & Pollastri, G. Porter, PaleAle 4.0: high-accuracy prediction of protein secondary structure and relative solvent accessibility. *Bioinformatics* **29**, 2056–2058, <https://doi.org/10.1093/bioinformatics/btt344> (2013).
35. Walsh, I., Pollastri, G. & Tosatto, S. C. E. Correct machine learning on protein sequences: a peer-reviewing perspective. *Briefings Bioinforma.* **17**, 831–840, <https://doi.org/10.1093/bib/bbv082> (2016).
36. He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. *arXiv:1512.03385* [cs] ArXiv: 1512.03385 (2015).
37. Ripley, B. D. *Pattern recognition and neural networks* (Cambridge University press, 1996).
38. Kabsch, W. & Sander, C. Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22**, 2577–2637, <https://doi.org/10.1002/bip.360221211> (1983).
39. Wang, S., Li, W., Liu, S. & Xu, J. RaptorX-Property: a web server for protein structure property prediction. *Nucleic Acids Res.* **44**, W430–W435, <https://doi.org/10.1093/nar/gkw306> (2016).
40. Zemla, A., Venclovas, C., Fidelis, K. & Rost, B. A modified definition of Sov, a segment-based measure for protein secondary structure prediction assessment. *Proteins: Struct. Funct. Bioinforma.* **34**, 220–223, [10.1002/\(SICI\)1097-0134\(19990201\)34:2<220::AID-PROT7>3.0.CO;2-K](https://doi.org/10.1002/(SICI)1097-0134(19990201)34:2<220::AID-PROT7>3.0.CO;2-K) (1999).
41. Liu, T. & Wang, Z. SOV\_refine: A further refined definition of segment overlap score and its significance for protein structure similarity. *Source Code for Biol. Medicine* **13**, 1, <https://doi.org/10.1186/s13029-018-0068-7> (2018).
42. Klausen, M. S. *et al.* NetSurfP-2.0: improved prediction of protein structural features by integrated deep learning. *bioRxiv* 311209, <https://doi.org/10.1101/311209> (2018).
43. Fang, C., Shang, Y. & Xu, D. MUFOLD-SS: New deep inception-inside-inception networks for protein secondary structure prediction. *Proteins: Struct. Funct. Bioinforma.* **86**, 592–598, <https://doi.org/10.1002/prot.25487> (2018).
44. Hanson, J., Paliwal, K., Litfin, T., Yang, Y. & Zhou, Y. Improving prediction of protein secondary structure, backbone angles, solvent accessibility and contact numbers by using predicted contact maps and an ensemble of recurrent and residual convolutional neural networks. *Bioinformatics*, <https://doi.org/10.1093/bioinformatics/bty1006> (2018).
45. Drozdetskiy, A., Cole, C., Procter, J. & Barton, G. J. JPred4: a protein secondary structure prediction server. *Nucleic Acids Res.* **43**, W389–W394, <https://doi.org/10.1093/nar/gkv332> (2015).
46. Fox, N. K., Brenner, S. E. & Chandonia, J.-M. SCOPe: Structural Classification of Proteins—extended, integrating SCOP and ASTRAL data and classification of new structures. *Nucleic Acids Res.* **42**, D304–D309, <https://doi.org/10.1093/nar/gkt1240> (2014).
47. Cuff, J. A. & Barton, G. J. Evaluation and improvement of multiple sequence methods for protein secondary structure prediction. *Proteins: Struct. Funct. Bioinforma.* **34**, 508–519, [10.1002/\(SICI\)1097-0134\(19990301\)34:4<508::AID-PROT10>3.0.CO;2-4](https://doi.org/10.1002/(SICI)1097-0134(19990301)34:4<508::AID-PROT10>3.0.CO;2-4) (1999).
48. Klausen, M. S. *et al.* NetSurfP-2.0: Improved prediction of protein structural features by integrated deep learning. *Proteins: Struct. Funct. Bioinforma.* **87**, 520–527, <https://doi.org/10.1002/prot.25674> (2019).
49. Hanson, J., Paliwal, K., Litfin, T., Yang, Y. & Zhou, Y. Accurate prediction of protein contact maps by coupling residual two-dimensional bidirectional long short-term memory with convolutional neural networks. *Bioinformatics* **34**, 4039–4045, <https://doi.org/10.1093/bioinformatics/bty481> (2018).
50. Seemayer, S., Gruber, M. & Söding, J. CCMpred—fast and precise prediction of protein residue–residue contacts from correlated mutations. *Bioinformatics* **30**, 3128–3130, <https://doi.org/10.1093/bioinformatics/btu500> (2014).
51. Morcos, F. *et al.* Direct-coupling analysis of residue coevolution captures native contacts across many protein families. *Proc. Natl. Acad. Sci.* **108**, E1293–E1301, <https://doi.org/10.1073/pnas.1111471108> (2011).
52. Torrisi, M., Kaleel, M. & Pollastri, G. Porter 5: fast, state-of-the-art ab initio prediction of protein secondary structure in 3 and 8 classes. *bioRxiv* 289033, <https://doi.org/10.1101/289033> (2018).
53. Rost, B. & Sander, C. Combining evolutionary information and neural networks to predict protein secondary structure. *Proteins: Struct. Funct. Bioinforma.* **19**, 55–72, <https://doi.org/10.1002/prot.340190108> (1994).
54. Jones, D. T. & Swindells, M. B. Getting the most from PSI-BLAST. *Trends Biochem. Sci.* **27**, 161–164, [https://doi.org/10.1016/S0968-0004\(01\)02039-4](https://doi.org/10.1016/S0968-0004(01)02039-4) (2002).
55. Schäffer, A. A. *et al.* Improving the accuracy of PSI-BLAST protein database searches with composition-based statistics and other refinements. *Nucleic Acids Res.* **29**, 2994–3005 (2001).
56. Krogh, A. & Mitchison, G. Maximum entropy weighting of aligned sequences of proteins or DNA. *Proceedings. Int. Conf. on Intell. Syst. for Mol. Biol.* **3**, 215–221 (1995).
57. Pollastri, G., Przybylski, D., Rost, B. & Baldi, P. Improving the prediction of protein secondary structure in three and eight classes using recurrent neural networks and profiles. *Proteins: Struct. Funct. Bioinforma.* **47**, 228–235, <https://doi.org/10.1002/prot.10082> (2002).
58. Hornik, K. Approximation capabilities of multilayer feedforward networks. *Neural Networks* **4**, 251–257, [https://doi.org/10.1016/0893-6080\(91\)90009-T](https://doi.org/10.1016/0893-6080(91)90009-T) (1991).
59. Baú, D. *et al.* Distill: a suite of web servers for the prediction of one-, two- and three-dimensional structural features of proteins. *BMC Bioinforma.* **7**, 402, <https://doi.org/10.1186/1471-2105-7-402> (2006).
60. Mooney, C. & Pollastri, G. Beyond the Twilight Zone: Automated prediction of structural properties of proteins by recursive neural networks and remote homology information. *Proteins: Struct. Funct. Bioinforma.* **77**, 181–190, <https://doi.org/10.1002/prot.22429> (2009).
61. Pollastri, G., Martin, A. J., Mooney, C. & Vullo, A. Accurate prediction of protein secondary structure and solvent accessibility by consensus combiners of sequence and structure information. *BMC Bioinforma.* **8**, 201, <https://doi.org/10.1186/1471-2105-8-201> (2007).
62. Monteith, K., Carroll, J. L., Seppi, K. & Martinez, T. Turning Bayesian model averaging into Bayesian model combination. In *The 2011 International Joint Conference on Neural Networks*, 2657–2663, <https://doi.org/10.1109/IJCNN.2011.6033566> (2011).
63. Zhou, Z.-H., Wu, J. & Tang, W. Ensembling neural networks: Many could be better than all. *Artif. Intell.* **137**, 239–263, [https://doi.org/10.1016/S0004-3702\(02\)00190-X](https://doi.org/10.1016/S0004-3702(02)00190-X) (2002).
64. Polyak, B. T. Some methods of speeding up the convergence of iteration methods. *USSR Comput. Math. Math. Phys.* **4**, 1–17, [https://doi.org/10.1016/0041-5553\(64\)90137-5](https://doi.org/10.1016/0041-5553(64)90137-5) (1964).
65. Robbins, H. & Monro, S. A Stochastic Approximation Method. *The Annals Math. Stat.* **22**, 400–407, <https://doi.org/10.1214/aoms/1177729586> (1951).

## Acknowledgements

The work of M.T., and M.K. is supported by the Irish Research Council [GOIPG/2015/3717, and GOIPG/2014/603]. The UCD computing cluster provided computational resources for this work.

## Author Contributions

M.T. ran and designed most of the experiments, analysed the results and produced the first draft of the article, M.K. contributed to the analysis and to the design of the input encodings. G.P. contributed to the design of the study and the individual experiments, implemented much of the code, contributed to the analysis of the data and to the writing and revision of the manuscript. All authors reviewed the manuscript and agree with its contents.

## Additional Information

**Competing Interests:** The authors declare no competing interests.

**Publisher's note:** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2019