What cleaves? Is proteasomal cleavage prediction reaching a ceiling?

Ingo Ziegler, [1] Bolei Ma,[1] Ercong Nie,[1]
Bernd Bischl,[2,3] David Rügamer,[2] Benjamin Schubert,[4,5] Emilio Dorigatti[2,4]
[1] Center for Information and Language Processing, LMU Munich,
[2] Department of Statistics, LMU Munich,
[3] Munich Center For Machine Learning,
[4] Institute of Computational Biology, Helmholtz Zentrum München,
[5] Department of Mathematics, TUM Munich
`{ziegler.ingo, bolei.ma}@campus.lmu.de`, `nie@cis.lmu.de`,
`{bernd.bischl, david.ruegamer, emilio.dorigatti}@stat.uni-muenchen.de`
`benjamin.schubert@helmholtz-muenchen.de`

Abstract

Epitope vaccines are a promising direction to enable precision treatment for cancer, autoimmune diseases, and allergies. Effectively designing such vaccines requires accurate prediction of proteasomal cleavage in order to ensure that the epitopes in the vaccine are presented to T cells by the major histocompatibility complex (MHC). While direct identification of proteasomal cleavage *in vitro* is cumbersome and low throughput, it is possible to implicitly infer cleavage events from the termini of MHC-presented epitopes, which can be detected in large amounts thanks to recent advances in high-throughput MHC ligandomics. Inferring cleavage events in such a way provides an inherently noisy signal which can be tackled with new developments in the field of deep learning that supposedly make it possible to learn predictors from noisy labels. Inspired by such innovations, we sought to modernize proteasomal cleavage predictors by benchmarking a wide range of recent methods, including LSTMs, transformers, CNNs, and denoising methods, on a recently introduced cleavage dataset. We found that increasing model scale and complexity appeared to deliver limited performance gains, as several methods reached about 88.5% AUC on C-terminal and 79.5% AUC on N-terminal cleavage prediction. This suggests that the noise and/or complexity of proteasomal cleavage and the subsequent biological processes of the antigen processing pathway are the major limiting factors for predictive performance rather than the specific modeling approach used. While biological complexity can be tackled by more data and better models, noise and randomness inherently limit the maximum achievable predictive performance. All our datasets and experiments are available at `https://github.com/ziegler-ingo/cleavage_prediction`.

1 Introduction

Proteasomal cleavage digestion of antigens is a major step of the antigen processing pathway, as by cleaving proteins in smaller peptides it determines what may be subsequently presented by the major histocompatibility complex (MHC) to T cells, potentially triggering an immune response [Blum et al., 2013]. Therefore, an important task for computational design of epitope vaccines (EV) is the prediction of this cleavage process, so that this information can be used by existing computational approaches [Dorigatti and Schubert, 2020a,b] to improve the efficacy of the vaccine.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Due to the difficulty of collecting large quantities of data *in vitro*, proteasomal cleavage events are usually inferred implicitly from MHC ligandomics data [Purcell et al., 2019] by matching eluted ligands to their progenitor protein to recover sequence information surrounding the terminals [Keşmir et al., 2002]. This procedure, however, does not give an indication of which amino acid sequences *cannot* result in a cleavage event, since missed cleavage sites are not observed in MHC ligands. Therefore, decoy negative samples are usually generated synthetically either by randomly shuffling the amino acids in a short window around the cleavage site or by considering artificial negative sites located around observed cleavage events [Calis et al., 2014]. Even though such negative samples are not entirely reliable, the growing availability of this kind of data Vita et al. [2018] spurred continuous development and improvement of proteasomal cleavage predictors Keşmir et al. [2002], Kuttler et al. [2000], Dönnes and Kohlbacher [2005], Nielsen et al. [2005] which have been recently revised in light of new innovations in the deep learning field [Amengual-Rigo and Guallar, 2021a, Dorigatti et al., 2022, Weeder et al., 2021, Amengual-Rigo and Guallar, 2021b].

As a consequence of these developments, we implemented and tested several binary classification methods on a proteasomal cleavage prediction task, carefully benchmarking a wide choice of architectures, embeddings, and training regimes.

2 Methods

In this benchmark study we consider three main axis of variation: the initial embedding of amino acids, the neural architecture of the predictor, and their training regime via noise handling and data augmentations.

2.1 Embedding

The choice of embedding is crucial as it influences what intrinsic information a model can exploit for classification [Ibtehaz and Kihara, 2021]; we thus consider various embeddings in our analysis, while keeping the base architecture equal. Specifically, we analyze the performance of a randomly initialized embedding layer that is optimized in conjunction with the loss function of the whole model, and the dedicated Prot2Vec [Asgari and Mofrad, 2015] embeddings trained with the well-established Word2Vec [Mikolov et al., 2013a,b] algorithm. Analogous to natural language, we design sequence embeddings by concatenating independently trained forward and backward amino acid representations of each input [Heigold et al., 2016].

Trainable tokenizers learn to form a given number of complex intra-token splits. This leads to a setting where the vocabulary size is now a tunable hyperparameter and thus has a direct impact on the size and quality of subsequently trained embedding representations. We extend our experiment with a vocabulary size 1000 and a vocabulary size $50\,000$ version of the byte-level byte pair encoding [Sennrich et al., 2016, BBPE], as well as a vocabulary size $50\,000$ version of the WordPiece [Schuster and Nakajima, 2012, WP] algorithm.

2.2 Neural architectures

Recurrent: Bidirectional long short-term memory networks (BiLSTM) [Graves and Schmidhuber, 2005] are well suited for a wide range of text classification tasks, thus we based nine of 12 model architectures around BiLSTMs. The fundamental structure for our BiLSTMs is built around the architecture proposed by Ozols et al. [Ozols et al., 2021], in which multiple sequential BiLSTMs are followed by a hidden and an output layer. For eight of our nine BiLSTM-related experiments, we choose two sequential BiLSTMs, where sequence dimensionality is reduced by taking the maximum value of the depth-wise per-residue output of the last layer. For the hidden layer, we used the Gaussian Error Linear Units (GELU) [Hendrycks and Gimpel, 2016] activation function. We additionally include an adjusted five BiLSTM version of a residual architecture between LSTM blocks, which aims to combat the shallow layer problem of deep LSTM architectures while also trying to improve the decoder quality with attention [Liu and Gong, 2019].

Transformers: Besides RNNs, the attention mechanism introduced by Vaswani et al. enabled a whole new architecture capable of processing sequences: the transformer [Vaswani et al., 2017]. We, therefore, integrated ProtTrans' T5-XL encoder-only model [Elnaggar et al., 2022] featuring 1.2

billion parameters, as well as ESM2 transformer [Lin et al., 2022] in its 150 million parameter version. Additionally, we include a fine-tuning performance of ESM2 by adding a linear layer projection from its vocabulary-sized per-residue Roberta Language Model Head [Liu et al., 2019a, Rives et al., 2021] to our binary classification target.

Convolutional and Perceptron: We take the DeepCleave [Li et al., 2019] attention-enhanced convolutional neural network [LeCun et al., 1998, CNN] architecture into our benchmark analysis. Furthermore, stacking fully connected layers without any convolutional or recurrent features, e.g., in DeepCalpain [Liu et al., 2019b] or Terminitor [Yang et al., 2020], has also been successfully applied to protein data. As baseline, we include a single hidden layer perceptron [Rumelhart et al., 1986] with Rectified Linear Units [Agarap, 2018] as activation function into the analysis.

2.3 Training

Dataset: We used the dataset introduced in [Dorigatti et al., 2022], which contains 229 163 and 222 181 N- and C-terminals cleavage sites respectively. Each cleavage site is captured into a window comprising six amino acids to its left and four to its right, and is associated with six decoy negative samples obtained by considering the three residues preceding and following it, resulting in a total of 1 434 989 and 1 419 501 samples after deduplication for N- and C-terminals. As the decoy negatives are situated in close proximity to real cleavage sites and due to the probabilistic nature of proteasomal cleavage, some of the negative samples are likely to be actual, unmeasured cleavage sites, and may influence the performance of predictors trained using such data.

Noisy labels: To reduce the impact of asymmetric label noise on the performance of our classifiers, we take five recent deep learning-specific denoising approaches into consideration: a noise adaptation layer, which attempts to learn the noise distribution in the data [Goldberger and Ben-Reuven, 2017], co-teaching, where two models are trained simultaneously by deciding for the respective other network which samples from a mini-batch to use for training [Han et al., 2018], and co-teaching-plus [Yu et al., 2019], which updates co-teaching with the disagreement learning approach of decoupling [Malach and Shalev-Shwartz, 2017]. We additionally consider a joint training method with co-regularization (JoCoR) [Wei et al., 2020] and DivideMix [Li et al., 2020a] for benchmarking. DivideMix is a holistic approach originally developed for computer vision and integrates multiple frameworks, such as co-teaching and MixMatch [Berthelot et al., 2019], into one. As MixMatch builds upon MixUp [Zhang et al., 2018], which was developed for image data, we adjust it for sequential data by mixing up the embedded sequence representation [Guo et al., 2019] instead of the pixel input in the data loading process.

Data augmentation: For all models, we apply data augmentation directly on the input sequences to combat overfitting and improve generalizability by masking a random amino acid per sequence as unknown [Shen et al., 2021]. All predictors except ESM2 fine-tuning use adaptive momentum [Kingma and Ba, 2015] as their optimization technique, whereas ESM2 fine-tuning uses adaptive momentum with decoupled weight decay [Loshchilov and Hutter, 2017]. All models without denoising techniques use (binary) cross-entropy loss [Cox, 1958], while all denoising models calculate dedicated losses.

3 Experimental protocol

Evaluation: As previously mentioned, some negative samples may actually result in a proteasomal cleavage event *in vivo* due to the way these negative samples are generated. For this reason, traditional binary classification metrics such as accuracy, precision, recall, etc. are misleading and model evaluation should instead be based on the AUC [Menon et al., 2015]. We reserved a random 10% of each terminal dataset as test dataset used for the final evaluation of the best hyperparameters.

Hyperparameter optimization: Due to computational limitations, we split up the hyperparameter search into three priority groups: group one used Ray Tune's [Moritz et al., 2018] implementation of the asynchronous hyperband algorithm [Li et al., 2020b] and evaluated each configuration in a ten-folds cross-validation (CV), while for groups two and three we chose hyperparameters manually and evaluated each configuration with five-folds CV (group two) or a single run on a held-out validation set (group three). We then used the best hyperparameter combination to train each

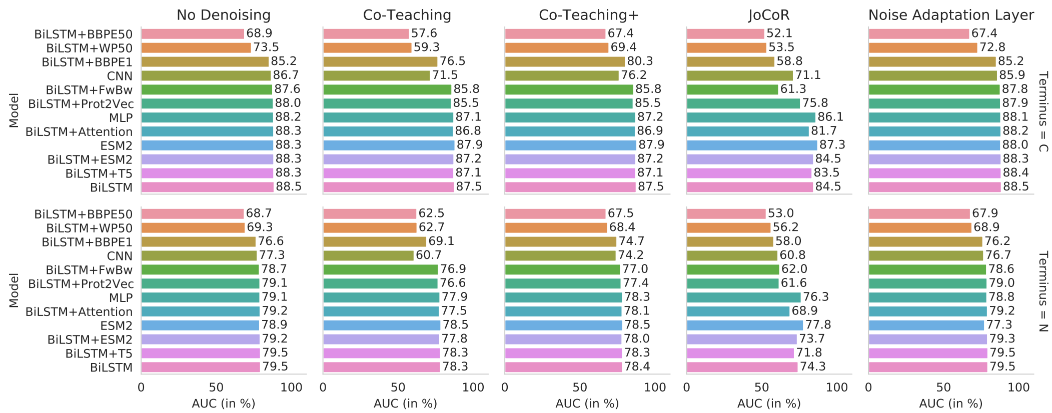

Figure 1: Model performances on C- and N-terminal

architecture with all denoising methods, except for DivideMix where we only trained the overall best performing architecture due to computational limitations. Information on the different architectures is in Appendix A.1, while the exact hyperband ranges and chosen hyperparameters for all models can be found in Appendix A.2.

4 Results

The overall best performing C-terminal model architecture as measured by AUC was the BiLSTM at 88.55% without any denoising methods, while for the N-term, the BiLSTM+T5 with noise adaptation layer version narrowly outperformed the base BiLSTM version by 0.04 percentage points at a level of 79.54% AUC (Figure 1 and Appendix A.3 and A.4). If denoising techniques were applied, the noise adaptation layer consistently performed best for both the C- and N-terminal. However, in 11 (10) of 12 models for the C-terminal (N-terminal), no denoising method resulted in superior results. Co-teaching-plus dominated co-teaching along all (11) model architectures in the C-terminal (N-terminal). JoCoR appeared to significantly hinder model performance in all architectures, whereas DivideMix also reduced the BiLSTM AUC score by around 2.4 percentage points in both terminals. While the best-performing BiLSTM consisted of 4.6 million parameters, the MLP with 30.529 parameters only lacked 0.38 percentage points AUC behind and additionally beat the pre-trained Prot2Vec as well as DeepCleave architectures, both featuring around 16 million parameters in the C-terminal. All transformer architectures in the ranges of 148 million (ESM2-based) and 1.2 billion (T5) parameters ranked behind the BiLSTM architecture but narrowly outperformed the MLP with AUC scores of around 88.32%.

For the C-terminal, models including trainable tokenizer dropped to their worst-performing state compared to their fixed-vocabulary counterpart, especially when increasing the number of to-be-learned amino acid sub-string combinations. Whereas the BiLSTM with BBPE vocabulary size 1000 drops 3.3 percentage points to 85.25% AUC, the same model architecture with 50 000 learned sub-string combinations was only able to reach 68.92% AUC. A similar but less severe pattern could be observed with WordPiece encodings, where the size 50 000 vocabulary version reached 73.46% AUC. If these models additionally featured denoising methods, the performance loss intensified up to a level of almost random-guessing (52.09% AUC for 50 000 BBPE and JoCoR).

Interestingly, the N-terminal showed a different behavior for certain architecture combinations. BiLSTM+Attention and BiLSTM+Prot2Vec had significantly larger performance drop-offs from their best-performing model for JoCoR-denoising (10.7 and 18 percentage points, respectively) compared to the C-terminal (6.8 and 12.7 percentage points, respectively). On the other hand, the performance loss due to trainable tokenizers paired with JoCoR was less severe in the N-terminal (21.5, 26.5, 23.2 percentage points, respectively) for BBPE-1000, BBPE-50 000, and WP-50 000 than in the C-terminal (29.7, 36.4, 35.1 percentage points, respectively).

Replacing the embedding layer with a forward-backward representation yielded comparable performance to the base BiLSTM architecture. Nonetheless, the base BiLSTM architecture was preferable as the additional forward-backward encoding steps increased training time by a factor of six.

5 Conclusion

Our benchmarking of various deep learning architectures for the task of proteasomal cleavage prediction has shown that several embedding techniques in combination with model architectures of vastly different scale and complexity can reach a performance of around 88.5% AUC for C-terminals and 79.5% AUC for N-terminals. Denoising techniques as well as trainable tokenizers appeared to offer limited to no, or even negative benefit. Such saturated results suggest that different modeling choices of architectures, embeddings, or training regimes are unlikely to yield significantly better predictive performance, and further efforts for proteasomal cleavage prediction should focus on a more comprehensive modeling of the antigen pathway. Another possibility is that these biological processes are simply too noisy and random to allow more accurate predictions, in which case we may already be close to the boundary of what is possible to achieve.

Acknowledgments and Disclosure of Funding

E. D. was supported by the Helmholtz Association under the joint research school “Munich School for Data Science - MUDS“ (Award Number HIDSS-0006). B. S. acknowledges financial support by the Postdoctoral Fellowship Program of the Helmholtz Zentrum München.

References

Janice S. Blum, Pamela A. Wearsch, and Peter Cresswell. Pathways of antigen processing. *Annual Review of Immunology*, 31(1):443–473, March 2013. doi: 10.1146/annurev-immunol-032712-095910.

Emilio Dorigatti and Benjamin Schubert. Graph-theoretical formulation of the generalized epitope-based vaccine design problem. *PLOS Computational Biology*, 16(10):e1008237, October 2020a. doi: 10.1371/journal.pcbi.1008237.

Emilio Dorigatti and Benjamin Schubert. Joint epitope selection and spacer design for string-of-beads vaccines. *Bioinformatics*, 36(Supplement_2):i643–i650, December 2020b. doi: 10.1093/bioinformatics/btaa790.

Anthony W. Purcell, Sri H. Ramarathinam, and Nicola Ternette. Mass spectrometry–based identification of MHC-bound peptides for immunopeptidomics. *Nat Protoc*, 14(6):1687–1707, may 2019. doi: 10.1038/s41596-019-0133-y.

Can Keşmir, Alexander K. Nussbaum, Hansjörg Schild, Vincent Detours, and Søren Brunak. Prediction of proteasome cleavage motifs by neural networks. *Protein Engineering, Design and Selection*, 15(4):287–296, April 2002. doi: 10.1093/protein/15.4.287.

Jorg J. A. Calis, Peter Reinink, Christin Keller, Peter M. Kloetzel, and Can Keşmir. Role of peptide processing predictions in t cell epitope identification: contribution of different prediction programs. *Immunogenetics*, 67(2):85–93, December 2014. doi: 10.1007/s00251-014-0815-0. URL `https://doi.org/10.1007/s00251-014-0815-0`.

Randi Vita, Swapnil Mahajan, James A Overton, Sandeep Kumar Dhanda, Sheridan Martini, Jason R Cantrell, Daniel K Wheeler, Alessandro Sette, and Bjoern Peters. The Immune Epitope Database (IEDB): 2018 update. *Nucleic Acids Research*, 47(D1):D339–D343, 10 2018. ISSN 0305-1048. doi: 10.1093/nar/gky1006.

Christina Kuttler, Alexander K Nussbaum, Tobias P Dick, Hans-Georg Rammensee, Hansjörg Schild, and Karl-Peter Hadeler. An algorithm for the prediction of proteasomal cleavages. *Journal of Molecular Biology*, 298(3):417–429, May 2000. doi: 10.1006/jmbi.2000.3683.

Pierre Dönnes and Oliver Kohlbacher. Integrated modeling of the major events in the MHC class i antigen processing pathway. *Protein Science*, 14(8):2132–2140, August 2005. doi: 10.1110/ps.051352405.

Morten Nielsen, Claus Lundegaard, Ole Lund, and Can Keşmir. The role of the proteasome in generating cytotoxic t-cell epitopes: insights obtained from improved predictions of proteasomal cleavage. *Immunogenetics*, 57(1-2):33–41, mar 2005. doi: 10.1007/s00251-005-0781-7.

Pep Amengual-Rigo and Victor Guallar. NetCleave: an open-source algorithm for predicting c-terminal antigen processing for MHC-i and MHC-II. *Scientific Reports*, 11(1), June 2021a. doi: 10.1038/s41598-021-92632-y.

Emilio Dorigatti, Bernd Bischl, and Benjamin Schubert. Improved proteasomal cleavage prediction with positive-unlabeled learning. *arXiv preprint arXiv:2209.07527*, 2022.

Benjamin R Weeder, Mary A Wood, Ellysia Li, Abhinav Nellore, and Reid F Thompson. pepsickle rapidly and accurately predicts proteasomal cleavage sites for improved neoantigen identification. *Bioinformatics*, 37(21): 3723–3733, September 2021. doi: 10.1093/bioinformatics/btab628. URL `https://doi.org/10.1093/bioinformatics/btab628`.

Pep Amengual-Rigo and Victor Guallar. NetCleave: an open-source algorithm for predicting c-terminal antigen processing for MHC-i and MHC-II. *Scientific Reports*, 11(1), June 2021b. doi: 10.1038/s41598-021-92632-y. URL `https://doi.org/10.1038/s41598-021-92632-y`.

Nabil Ibtehaz and Daisuke Kihara. Application of sequence embedding in protein sequence-based predictions. *arXiv preprint arXiv:2110.07609*, 2021.

Ehsaneddin Asgari and Mohammad RK Mofrad. Continuous distributed representation of biological sequences for deep proteomics and genomics. *PloS one*, 10(11):e0141287, 2015.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, September 2013a. URL `http://arxiv.org/abs/1301.3781`. arXiv: 1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *arXiv:1310.4546 [cs, stat]*, October 2013b. URL `http://arxiv.org/abs/1310.4546`. arXiv: 1310.4546.

Georg Heigold, Guenter Neumann, and Josef van Genabith. Neural morphological tagging from characters for morphologically rich languages. *CoRR*, abs/1606.06640, 2016. URL `http://arxiv.org/abs/1606.06640`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL `http://aclweb.org/anthology/P16-1162`.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Alex Graves and Jürgen Schmidhuber. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18(5-6):602–610, July 2005. ISSN 08936080. doi: 10.1016/j.neunet.2005.06.042. URL `https://linkinghub.elsevier.com/retrieve/pii/S0893608005001206`.

Matiss Ozols, Alexander Eckersley, Christopher I. Platt, Callum Stewart-McGuinness, Sarah A. Hibbert, Jerico Revote, Fuyi Li, Christopher E. M. Griffiths, Rachel E. B. Watson, Jiangning Song, Mike Bell, and Michael J. Sherratt. Predicting proteolysis in complex proteomes using deep learning. *International Journal of Molecular Sciences*, 22(6), 2021. ISSN 1422-0067. doi: 10.3390/ijms22063071. URL `https://www.mdpi.com/1422-0067/22/6/3071`.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Jiale Liu and Xinqi Gong. Attention mechanism enhanced lstm with residual architecture and its application for protein-protein interaction residue pairs prediction. *BMC bioinformatics*, 20(1):1–11, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv:1706.03762 [cs]*, December 2017. URL `http://arxiv.org/abs/1706.03762`. arXiv: 1706.03762.

Ahmed Elnaggar, Michael Heinzinger, Christian Dallago, Ghalia Rehawi, Yu Wang, Llion Jones, Tom Gibbs, Tamas Feher, Christoph Angerer, Martin Steinegger, Debsindhu Bhowmik, and Burkhard Rost. Prottrans: Toward understanding the language of life through self-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(10):7112–7127, 2022. doi: 10.1109/TPAMI.2021.3095381.

Zeming Lin, Halil Akin, Roshan Rao, Brian Hie, Zhongkai Zhu, Wenting Lu, Allan dos Santos Costa, Maryam Fazel-Zarandi, Tom Sercu, Sal Candido, and Alexander Rives. Language models of protein sequences at the scale of evolution enable accurate structure prediction. *bioRxiv*, 2022. doi: 10.1101/2022.07.20.500902. URL `https://www.biorxiv.org/content/early/2022/07/21/2022.07.20.500902`.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv:1907.11692 [cs]*, July 2019a. URL `http://arxiv.org/abs/1907.11692`. arXiv: 1907.11692.

Alexander Rives, Joshua Meier, Tom Sercu, Siddharth Goyal, Zeming Lin, Jason Liu, Demi Guo, Myle Ott, C. Lawrence Zitnick, Jerry Ma, and Rob Fergus. Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. *Proceedings of the National Academy of Sciences*, 118(15):e2016239118, 2021. doi: 10.1073/pnas.2016239118. URL `https://www.pnas.org/doi/abs/10.1073/pnas.2016239118`.

Fuyi Li, Jinxiang Chen, André Leier, Tatiana Marquez-Lago, Quanzhong Liu, Yanze Wang, Jerico Revote, A Ian Smith, Tatsuya Akutsu, Geoffrey I Webb, Lukasz Kurgan, and Jiangning Song. DeepCleave: a deep learning predictor for caspase and matrix metalloprotease substrates and cleavage sites. *Bioinformatics*, 36(4): 1057–1065, 09 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz721. URL `https://doi.org/10.1093/bioinformatics/btz721`.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-Based Learning Applied to Document Recognition. *Proc. of the IEEE*, pages 1–46, 1998.

Ze-Xian Liu, Kai Yu, Jingsi Dong, Linhong Zhao, Zekun Liu, Qingfeng Zhang, Shihua Li, Yimeng Du, and Han Cheng. Precise prediction of calpain cleavage sites and their aberrance caused by mutations in cancer. *Frontiers in Genetics*, 10, 2019b. ISSN 1664-8021. doi: 10.3389/fgene.2019.00715. URL `https://www.frontiersin.org/articles/10.3389/fgene.2019.00715`.

Chen Yang, Chenkai Li, Ka Ming Nip, René L Warren, and Inanc Birol. Terminitor: Cleavage site prediction using deep learning models. *bioRxiv*, 2020. doi: 10.1101/710699. URL `https://www.biorxiv.org/content/early/2020/04/23/710699`.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

Abien Fred Agarap. Deep learning using rectified linear units (relu). *CoRR*, abs/1803.08375, 2018. URL `http://arxiv.org/abs/1803.08375`.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=H12GRgcxg`.

Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *Advances in neural information processing systems*, 31, 2018.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR, 2019.

Eran Malach and Shai Shalev-Shwartz. Decoupling" when to update" from" how to update". *Advances in neural information processing systems*, 30, 2017.

Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. Combating noisy labels by agreement: A joint training method with co-regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13726–13735, 2020.

Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *International Conference on Learning Representations*, 2020a. URL `https://openreview.net/forum?id=HJgExaVtwr`.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. *International Conference on Learning Representations*, 2018.

Hongyu Guo, Yongyi Mao, and Richong Zhang. Augmenting data with mixup for sentence classification: An empirical study. *arXiv preprint arXiv:1905.08941*, 2019.

Hongyu Shen, Layne C. Price, Taha Bahadori, and Franziska Seeger. Improving generalizability of protein sequence models with data augmentations. *bioRxiv*, 2021. doi: 10.1101/2021.02.18.431877. URL `https://www.biorxiv.org/content/early/2021/02/18/2021.02.18.431877`.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

D.R. Cox. The Regression Analysis of binary sequences. *Journal of the Royal Statistical Society: Series B (Methodological)*, 2(2), 1958. ISSN 00359246.

Aditya Menon, Brendan Van Rooyen, Cheng Soon Ong, and Bob Williamson. Learning from corrupted binary labels via class-probability estimation. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 125–134, Lille, France, 07–09 Jul 2015. PMLR.

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th USENIX Symposium on Operating Systems Design and Implementation (OSDI 18)*, pages 561–577, 2018.

Liam Li, Kevin Jamieson, Afshin Rostamizadeh, Ekaterina Gonina, Jonathan Ben-Tzur, Moritz Hardt, Benjamin Recht, and Ameet Talwalkar. A system for massively parallel hyperparameter tuning. *Proceedings of Machine Learning and Systems*, 2:230–246, 2020b.

A Appendix

A.1 Architecture information

Table 1: Number of parameters and training time for each model, without considering denoising (see later tables for this)

Models	Time (s/epoch)	Epochs	Parameters	Trainable
BiLSTM	25	15	4 655 984	4 655 984
BiLSTM+Attention	20	20	1 632 391	1 632 391
BiLSTM+Prot2Vec	20	15	16 009 371	5 830 371
CNN	45	60	16 084 198	16 084 198
MLP	4	30	30 529	30 529
BiLSTM+ESM2	330	10	152 998 267	4 858 113
ESM2	900	3	148 140 188	148 140 188
BiLSTM+T5	780	10	1 214 572 801	6 430 977
BiLSTM+BBPE1	13	15	5 319 409	5 319 409
BiLSTM+BBPE50	12	15	12 669 409	12 669 409
BiLSTM+WP50	16	15	12 669 409	12 669 409
BiLSTM+FwBw	120	15	4 315 369	4 315 369

A.2 Hyperparameters

Table 2: BiLSTM

Hyperparameter	Range (Uniformly random choice)	Final value C-terminal	Final value N-terminal
Epochs trained	≤ 25	15	15
Learning rate	$\{5 \times 10^{-5}, 10^{-4}, 3 \times 10^{-4}\}$	3×10^{-4}	3×10^{-4}
Dropout rate	$\{0.45, 0.46, \ldots, 0.51, 0.52\}$	0.5	0.5
Linear layer size	$[120, 181)$	164	179
Embedding dimension	$[50, 201)$	91	76
LSTM size 1	$[220, 281)$	228	252
LSTM size 2	$[450, 520)$	506	518

Table 3: BiLSTM+Attention

Hyperparameter	Range (Uniformly random choice)	Final value C-terminal	Final value N-terminal
Epochs trained	≤ 25	20	20
Learning rate	$\{3 \times 10^{-5}, 5 \times 10^{-5}, 8 \times 10^{-5}, 10^{-4}\}$	10^{-4}	10^{-4}
Dropout rate	$\{0.45, 0.46, \ldots, 0.51, 0.52\}$	0.5	0.5
Linear layer size	$[100, 181)$	147	150
Embedding dimension	$\{120, 124, \ldots, 216, 220\}$	216	216
LSTM size	$[64, 131)$	108	111
Attention heads	$\{1, 2, 4\}$	4	1

Table 4: BiLSTM+Prot2Vec

Hyperparameter	Range (Uniformly random choice)	Final value C-terminal	Final value N-terminal
Epochs trained	≤ 60	60	60
Learning rate	$\{8 \times 10^{-5}, , 10^{-4}, 3 \times 10^{-4}, 5 \times 10^{-4}\}$	3×10^{-4}	3×10^{-4}
Dropout rate	$\{0.45, 0.46, \ldots, 0.51, 0.52\}$	0.5	0.5
Linear layer size	$[120, 180)$	145	139
LSTM size	$[480, 531)$	480	531

Table 5: CNN

Hyperparameter	**Range** (Uniformly random choice)	**Final value** C-terminal	N-terminal
Epochs trained	≤ 60	60	60
Learning Rate	$\{8\times10^{-5}, 10^{-4}, 3\times10^{-4}, 5\times10^{-4}\}$	3×10^{-4}	3×10^{-4}
Dropout Rate	$\{0, 0.02, \ldots, 0.08, 0.1\}$	0.04	0.08
Linear layer size 1	$[64, 101)$	79	89
Linear layer size 2	$[15, 33)$	24	15
Attention heads 1	$\{1, 2, 3, 4, 5, 6\}$	3	4
Attention heads 2	$\{1, 2, 3, 4, 5, 6\}$	2	3
Filter size 1	-	1	1
Number filters 1	$[220, 301)$	220	249
Number filters 2	$[220, 301)$	262	229
Filter size 2a	$\{1, 3, 5, 7\}$	3	1
Filter size 2b	$\{15, 17, 19, 21, 23, 25\}$	17	15
Filter size 3c	$\{13, 15, 17, 19, 21, 23\}$	13	13
Number filters 3	$[350, 431)$	398	400
Filter size 3a	$\{11, 13, 15, 17, 19\}$	11	13
Filter size 3b	$\{13, 15, 17, 19, 21, 23\}$	15	21
Filter size 3c	$\{11, 13, 15, 17, 19\}$	19	15

Table 6: MLP

Hyperparameter	**Range** (Uniformly random choice)	**Final value** C-terminal	N-terminal
Epochs trained	≤ 60	30	30
Learning rate	$\{10^{-4}, 5\times10^{-4}, 8\times10^{-4}, 10^{-3}\}$	10^{-3}	10^{-3}
Dropout rate	$\{0.1, 0.12, \ldots, 0.24, 0.26\}$	0.24	0.24
Linear layer size	$[120, 201)$	144	167

Table 7: BiLSTM+ESM2

Hyperparameter	**Final value** C-terminal	N-terminal
Epochs trained	10	10
Learning rate	3×10^{-4}	3×10^{-4}
Dropout rate	0.5	0.5
Linear layer size	128	128
LSTM size	512	512

Table 8: ESM2

Hyperparameter	**Final value** C-terminal	N-terminal
Epochs trained	3	3
Learning rate	2×10^{-5}	2×10^{-5}
Dropout rate	0.5	0.5

Table 9: BiLSTM+T5

Hyperparameter	**Final value** C-terminal	N-terminal
Epochs trained	10	10
Learning rate	3×10^{-4}	3×10^{-4}
Dropout rate	0.5	0.5
Linear layer size	128	128
LSTM size	512	512

Table 10: BiLSTM+BBPE1, BiLSTM+BBPE50, BiLSTM+WP50

Hyperparameter	**Final value**	
	C-terminal	**N-terminal**
Epochs trained	15	15
Learning rate	10^{-4}	10^{-4}
Dropout rate	0.5	0.5
Embedding dimension	150	150
Linear layer size	128	128
LSTM size	512	512

Table 11: BiLSTM+FwBw

Hyperparameter	**Final value**	
	C-terminal	**N-terminal**
Epochs trained	15	15
Learning rate	10^{-4}	10^{-4}
Dropout rate	0.5	0.5
Linear layer size 1	128	128
LSTM size 1	128	128
LSTM size 2	512	512
Sequence encoding embedding dimension	100	100
Sequence encoding BiLSTM size	200	200

Table 12: Co-Teaching, Co-Teaching+, JoCoR

Hyperparameter	**Final value**		
	Co-teaching	**Co-teaching+**	**JoCoR**
Number scale-up epochs	10	10	10
Noise rate	0.2	0.2	0.2
Forget rate	0.2	0.2	0.1
Exponent	1	1	1

Table 13: DivideMix

Hyperparameter	**Final value**
Number warm-up epochs	1
α	0.5
λ_u	0
Probability threshold	0.5
Temperature	0.5
Number scale-up epochs	5

Table 14: Noise adaptation layer

Hyperparameter	**Final value**
Number warm-up epochs	1
β	0.8

A.3 Results without denoising methods

Table 15: Model performances on C- and N-terminals

Priority	Models	C-terminal AUC	C-terminal ACC	N-terminal AUC	N-terminal ACC
1	BiLSTM	88.55 ± 0.12	79.50 ± 0.11	79.50 ± 0.11	83.51 ± 0.11
	BiLSTM+Attention	88.28 ± 0.08	79.24 ± 0.11	79.24 ± 0.11	83.36 ± 0.13
	BiLSTM+Prot2Vec	87.99 ± 0.14	79.10 ± 0.11	79.10 ± 0.11	83.22 ± 0.13
	CNN	86.66 ± 0.17	77.30 ± 0.82	77.30 ± 0.82	82.89 ± 0.22
	MLP	88.17 ± 0.11	79.08 ± 0.11	79.08 ± 0.11	83.33 ± 0.12
2	BiLSTM+ESM2	88.34 ± 0.05	79.24 ± 0.10	79.24 ± 0.10	83.35 ± 0.09
	ESM2	88.32 ± 0.16	78.91 ± 0.18	78.91 ± 0.18	82.63 ± 0.64
	BiLSTM+T5	88.32 ± 0.05	79.48 ± 0.11	79.48 ± 0.11	83.45 ± 0.08
3	BiLSTM+BBPE1	85.25	76.56	76.56	82.88
	BiLSTM+BBPE50	68.92	68.67	68.67	82.03
	BiLSTM+WP50	73.46	69.28	69.28	82.08
	BiLSTM+FwBw	87.59	78.71	78.71	83.15

A.4 Results with denoising methods

Table 16: BiLSTM with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	87.50	86.64	41	15	4 655 984	4 655 984
Co-Teaching+	87.50	86.64	41	15	4 655 984	4 655 984
JoCoR	84.53	85.49	41	15	4 655 984	4 655 984
DivideMix	86.25	84.02	210	15	4 656 149	4 656 149
Noise Adaptation Layer	88.49	87.02	22	15	4 656 149	4 656 149

Table 17: BiLSTM with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	78.28	83.21	44	15	5 096 135	5 096 135
Co-Teaching+	78.37	83.17	44	15	5 096 135	5 096 135
JoCoR	74.26	82.12	44	15	5 096 135	5 096 135
DivideMix	77.08	81.52	210	15	5 096 315	5 096 315
Noise Adaptation Layer	79.48	83.43	23	15	5 096 315	5 096 315

Table 18: BiLSTM+Attention with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	86.82	86.20	40	20	1 632 391	1 632 391
Co-Teaching+	86.91	86.16	40	20	1 632 391	1 632 391
JoCoR	81.70	85.04	38	20	1 632 391	1 632 391
Noise Adaptation Layer	88.23	86.90	18	20	1 632 539	1 632 539

Table 19: BiLSTM+Attention with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	77.48	83.13	43	20	1 718 233	1 718 233
Co-Teaching+	78.09	83.13	42	20	1 718 233	1 718 233
JoCoR	68.86	82.40	41	20	1 718 233	1 718 233
Noise Adaptation Layer	79.21	83.37	22	20	1 718 384	1 718 384

Table 20: BiLSTM+Prot2Vec with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	85.47	85.78	32	15	16 009 371	5 830 371
Co-Teaching+	85.47	85.78	31	15	16 009 371	5 830 371
JoCoR	75.80	82.20	32	15	16 009 371	5 830 371
Noise Adaptation Layer	87.93	86.60	15	15	16 009 517	5 830 517

Table 21: BiLSTM+Prot2Vec with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	76.64	83.00	40	15	16 772 049	6 593 049
Co-Teaching+	77.44	82.90	40	15	16 772 049	6 593 049
JoCoR	61.62	81.91	40	15	16 772 049	6 593 049
Noise Adaptation Layer	78.96	83.15	19	15	16 772 189	6 593 189

Table 22: CNN with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	71.51	82.35	102	60	16 084 198	16 084 198
Co-Teaching+	76.18	82.91	102	60	16 084 198	16 084 198
JoCoR	71.10	82.92	102	60	16 084 198	16 084 198
Noise Adaptation Layer	85.91	85.50	49	60	16 084 223	16 084 223

Table 23: CNN with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	60.71	81.88	102	60	15 237 057	15 237 057
Co-Teaching+	74.15	82.05	102	60	15 237 057	15 237 057
JoCoR	60.81	81.88	102	60	15 237 057	15 237 057
Noise Adaptation Layer	76.68	82.78	47	60	15 237 073	15 237 073

Table 24: MLP with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	87.14	86.45	6	30	30 529	30 529
Co-Teaching+	87.16	85.85	6	30	30 529	30 529
JoCoR	86.11	85.56	5	30	30 529	30 529
Noise Adaptation Layer	88.07	86.73	3	30	30 674	30 674

Table 25: MLP with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	77.89	83.21	6	30	35 405	35 405
Co-Teaching+	78.34	83.11	5	30	35 405	35 405
JoCoR	76.31	82.30	5	30	35 405	35 405
Noise Adaptation Layer	78.82	83.32	4	30	35 573	35 573

Table 26: BiLSTM+ESM2 with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	87.19	86.52	660	10	152 998 267	4 858 113
Co-Teaching+	87.19	86.52	660	10	152 998 267	4 858 113
JoCoR	84.52	84.93	660	10	152 998 267	4 858 113
Noise Adaptation Layer	88.32	86.88	330	10	152 998 396	4 858 242

Table 27: BiLSTM+ESM2 with denoising on N-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	77.76	83.25	660	10	152 998 267	4 858 113
Co-Teaching+	78.03	83.15	660	10	152 998 267	4 858 113
JoCoR	73.71	81.96	660	10	152 998 267	4 858 113
Noise Adaptation Layer	79.29	83.37	360	10	152 998 396	4 858 242

Table 28: ESM2 with denoising on C-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	87.93	85.56	2160	3	148 140 188	148 140 188
Co-Teaching+	87.93	85.56	2100	3	148 140 188	148 140 188
JoCoR	87.31	86.25	2100	3	148 140 188	148 140 188
Noise Adaptation Layer	87.97	86.58	960	3	148 140 222	148 140 222

Table 29: ESM2 with denoising on N-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	78.53	80.54	2220	3	148 140 188	148 140 188
Co-Teaching+	78.53	80.54	2220	3	148 140 188	148 140 188
JoCoR	77.83	83.02	2160	3	148 140 188	148 140 188
Noise Adaptation Layer	77.31	82.13	930	3	148 140 222	148 140 222

Table 30: BiLSTM+T5 with denoising on C-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	87.11	86.40	1500	10	1 214 572 801	6 430 977
Co-Teaching+	87.11	86.40	1500	10	1 214 572 801	6 430 977
JoCoR	83.48	83.88	1500	10	1 214 572 801	6 430 977
Noise Adaptation Layer	88.36	86.85	720	10	1 214 572 930	6 431 106

Table 31: BiLSTM+T5 with denoising on N-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	78.30	83.22	1560	10	1 214 572 801	6 430 977
Co-Teaching+	78.32	83.26	1560	10	1 214 572 801	6 430 977
JoCoR	71.76	82.28	1500	10	1 214 572 801	6 430 977
Noise Adaptation Layer	79.54	83.48	780	10	1 214 572 930	6 431 106

Table 32: BiLSTM+BBPE1 with denoising on C-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	76.55	83.84	33	15	5 319 409	5 319 409
Co-Teaching+	80.31	83.44	34	15	5 319 409	5 319 409
JoCoR	58.81	82.20	33	15	5 319 409	5 319 409
Noise Adaptation Layer	85.15	85.45	17	15	5 319 538	5 319 538

Table 33: BiLSTM+BBPE1 with denoising on N-terminal

Denoising methods	**AUC**	**ACC**	**Time (s/epoch)**	**Epochs**	**Parameters**	**Trainable parameters**
Co-Teaching	69.12	82.19	34	15	5 319 409	5 319 409
Co-Teaching+	74.65	82.37	34	15	5 319 409	5 319 409
JoCoR	58.03	81.88	33	15	5 319 409	5 319 409
Noise Adaptation Layer	76.15	82.58	17	15	5 319 538	5 319 538

Table 34: BiLSTM+BBPE50 with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	57.62	82.28	30	15	12 669 409	12 669 409
Co-Teaching+	67.40	82.35	30	15	12 669 409	12 669 409
JoCoR	52.09	82.20	30	15	12 669 409	12 669 409
Noise Adaptation Layer	67.42	82.38	16	15	12 669 538	12 669 538

Table 35: BiLSTM+BBPE50 with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	62.52	81.88	31	15	12 669 409	12 669 409
Co-Teaching+	67.45	81.98	32	15	12 669 409	12 669 409
JoCoR	53.02	81.88	31	15	12 669 409	12 669 409
Noise Adaptation Layer	67.90	81.95	15	15	12 669 538	12 669 538

Table 36: BiLSTM+WP50 with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	59.31	82.25	40	15	12 669 409	12 669 409
Co-Teaching+	69.37	82.57	40	15	12 669 409	12 669 409
JoCoR	53.45	82.20	39	15	12 669 409	12 669 409
Noise Adaptation Layer	72.80	82.98	20	15	12 669 538	12 669 538

Table 37: BiLSTM+WP50 with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	62.66	81.92	37	15	12 669 409	12 669 409
Co-Teaching+	68.44	81.97	37	15	12 669 409	12 669 409
JoCoR	56.25	81.88	36	15	12 669 409	12 669 409
Noise Adaptation Layer	68.91	81.99	20	15	12 669 538	12 669 538

Table 38: BiLSTM+FwBw with denoising on C-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	85.84	84.92	270	15	4 315 369	4 315 369
Co-Teaching+	85.84	84.92	270	15	4 315 369	4 315 369
JoCoR	61.26	82.20	270	15	4 315 369	4 315 369
Noise Adaptation Layer	87.75	86.09	120	15	4 315 498	4 315 498

Table 39: BiLSTM+FwBw with denoising on N-terminal

Denoising methods	AUC	ACC	Time (s/epoch)	Epochs	Parameters	Trainable parameters
Co-Teaching	76.86	82.45	270	15	4 315 369	4 315 369
Co-Teaching+	77.02	82.88	270	15	4 315 369	4 315 369
JoCoR	62.00	81.88	270	15	4 315 369	4 315 369
Noise Adaptation Layer	78.64	83.09	120	15	4 315 498	4 315 498