# Classification of Ubiquitin-Ligase Proteins: A Convolutional Neural Network Approach

**Chris Burgoyne**

## Introduction

Proteins are important macromolecules found in cells. They are formed by the polymerization of amino acids according to sequences of genetic code and perform a variety of duties that support life. Proteins are made from chains of amino acids. There are 20 common amino acids and they come together in unique combinations to form certain classes of proteins. These classes can be differentiated by the functions they perform, the commonalities of structure among the proteins in the class, and commonalities in the composition of amino acids in class-specific proteins[1].

Classification of proteins is an important and well studied task[2]. When a new protein is identified, its properties can be proposed based on the class which it is predicted to belong to. Protein classification can be a complex process, but one of the simpler ways that proteins can be classified is by examining the sequence of the protein[3]. While the exact sequence alone might not be particularly informative, one can infer information about the sequence based on the amino acids present in it.

One class of proteins that is difficult to classify based on the sequence is the Ubiquitin Ligases (UBL). A UBL is a protein that enables movement of ubiquitin from a ubiquitin carrier to another location by a unique mechanism. In particular, ubiquitination by E3 ligases regulates diverse cell activity and is of fundamental importance in cell biology[4]. The only recent attempt to classify UBL proteins by sequence alone used the Support-Vector Machine classification algorithm from machine learning to classify them in differentiation from non-UBL proteins[5]. The study used a Reduced Amino Acid (RED) alphabet encoding system for inferring features from the protein sequence, a known approach in the extraction of features from protein sequences[6]. While the method appears to have some relevance, it has since been found that the methodology overstates the accuracy of the SVM using this encoding system. At this stage, the problem of classifying UBL proteins does not have an accurate solution.

In this paper, a novel system using a machine learning approach is presented for the classification of UBL proteins in differentiation from other non-UBL proteins. The method uses an amino acid encoding system similar to the RED system mentioned above. The classification system uses a Convolutional Neural Network to create a model that differentiates UBL from non-UBL proteins. The results show promise and future work will likely produce a model that can classify UBL proteins with high accuracy. The code base can be found at https://github.com/koos-burgoyne/UBL.

## Methods

### *Dataset*

The dataset contains sequences from three different classes. There are 3,258 confirmed UBL proteins, along with a set of 26,790 confirmed non-UBL proteins. In addition to this, there is a set of 26,060 non-UBL proteins that have a zinc finger. These are considered a possible source of false-positive

classifications for UBL proteins. The total number of sequences in the dataset is thus 56,590. As can be seen in Table 1 this presents an issue with class-imbalance. The imbalanced frequencies of the classes can potentially skew the model during the training process to predict mostly the most frequently occurring class in the dataset. This issue is dealt with in the methodology section below but potentially limits the amount of data that can be used for training.

***Table 1:*** *Counts of proteins by class in the dataset*

| | UBL | non-UBL | Zinc-Finger | Total |
|---|---|---|---|---|
| Count | 3,740 | 26,790 | 26,060 | 56,590 |

*Features*

The dataset consists of amino acid sequences which are merely assemblages of characters. While preliminary investigation of the data suggested that this might provide enough information to perform some sort of basic classification, a fundamental tenet of machine learning is that features that provide variance in relation to the predictor are useful in determining the classification of an instance of data. In this study, each instance of data is a sequence of amino acids representing a protein. Beyond the sequence itself, the features inferred from the sequence are determined by the choice of alphabet encoding system.

***Table 2:*** *VHSE descriptors for the 20 natural amino acids that were used to infer 8 of the 28 features for protein sequences in this study (source:Xie et al., 2019)*

| AA | $VHSE_1$ | $VHSE_2$ | $VHSE_3$ | $VHSE_4$ | $VHSE_5$ | $VHSE_6$ | $VHSE_7$ | $VHSE_8$ |
|---|---|---|---|---|---|---|---|---|
| Ala A | 0.15 | −1.11 | −1.35 | −0.92 | 0.02 | −0.91 | 0.36 | −0.48 |
| Arg R | −1.47 | 1.45 | 1.24 | 1.27 | 1.55 | 1.47 | 1.30 | 0.83 |
| Asn N | −0.99 | 0.00 | −0.37 | 0.69 | −0.55 | 0.85 | 0.73 | −0.80 |
| Asp D | −1.15 | 0.67 | −0.41 | −0.01 | −2.68 | 1.31 | 0.03 | 0.56 |
| Cys C | 0.18 | −1.67 | −0.46 | −0.21 | 0.00 | 1.20 | −1.61 | −0.19 |
| Gln Q | −0.96 | 0.12 | 0.18 | 0.16 | 0.09 | 0.42 | −0.20 | −0.41 |
| Glu E | −1.18 | 0.40 | 0.10 | 0.36 | −2.16 | −0.17 | 0.91 | 0.02 |
| Gly G | −0.20 | −1.53 | −2.63 | 2.28 | −0.53 | −1.18 | 2.01 | −1.34 |
| His H | −0.43 | −0.25 | 0.37 | 0.19 | 0.51 | 1.28 | 0.93 | 0.65 |
| Ile I | 1.27 | −0.14 | 0.30 | −1.80 | 0.30 | −1.61 | −0.16 | −0.13 |
| Leu L | 1.36 | 0.07 | 0.26 | −0.80 | 0.22 | −1.37 | 0.08 | −0.62 |
| Lys K | −1.17 | 0.70 | 0.70 | 0.80 | 1.64 | 0.67 | 1.63 | 0.13 |
| Met M | 1.01 | −0.53 | 0.43 | 0.00 | 0.23 | 0.10 | −0.86 | −0.68 |
| Phe F | 1.52 | 0.61 | 0.96 | −0.16 | 0.25 | 0.28 | −1.33 | −0.20 |
| Pro P | 0.22 | −0.17 | −0.50 | 0.05 | −0.01 | −1.34 | −0.19 | 3.56 |
| Ser S | −0.67 | −0.86 | −1.07 | −0.41 | −0.32 | 0.27 | −0.64 | 0.11 |
| Thr T | −0.34 | −0.51 | −0.55 | −1.06 | 0.01 | −0.01 | −0.79 | 0.39 |
| Trp W | 1.50 | 2.06 | 1.79 | 0.75 | 0.75 | −0.13 | −1.06 | −0.85 |
| Tyr Y | 0.61 | 1.60 | 1.17 | 0.73 | 0.53 | 0.25 | −0.96 | −0.52 |
| Val V | 0.76 | −0.92 | 0.17 | −1.91 | 0.22 | −1.40 | −0.24 | −0.03 |

doi:10.1371/journal.pone.0074506.t001

In this study, a novel system based on the RED[6] alphabet encoding system was developed. Each character in the sequence is an important piece of information that can be used to infer more information about the sequence. In this study, the information inferred from each amino acid was based on work in which the numeric VHSE (principal component score vector of hydrophobic, steric, and electronic properties) descriptor of proteasomal cleavage sites were found for each amino acid[7]. Each of the 8 descriptors provided a score for each of the 20 amino acids. These scores were encoded into 8 features for every amino acid in the protein sequence.

In addition to this, the amino acids themselves were one-hot encoded. The alternative to this is to convert the characters in the protein sequence to integers but this single dimension numeric representation suggests that characters with are adjacent to one another in the alphabet are numerically closer to each other than numbers which are not adjacent. The true representation of amino acid similarity is of course not as simple as that, and to avoid having the model learn on such data each character representing an amino acid was one-hot encoded. This means that for each character in a protein sequence there are 20 additional features in which only one of them is the value 1, while all other values are 0.

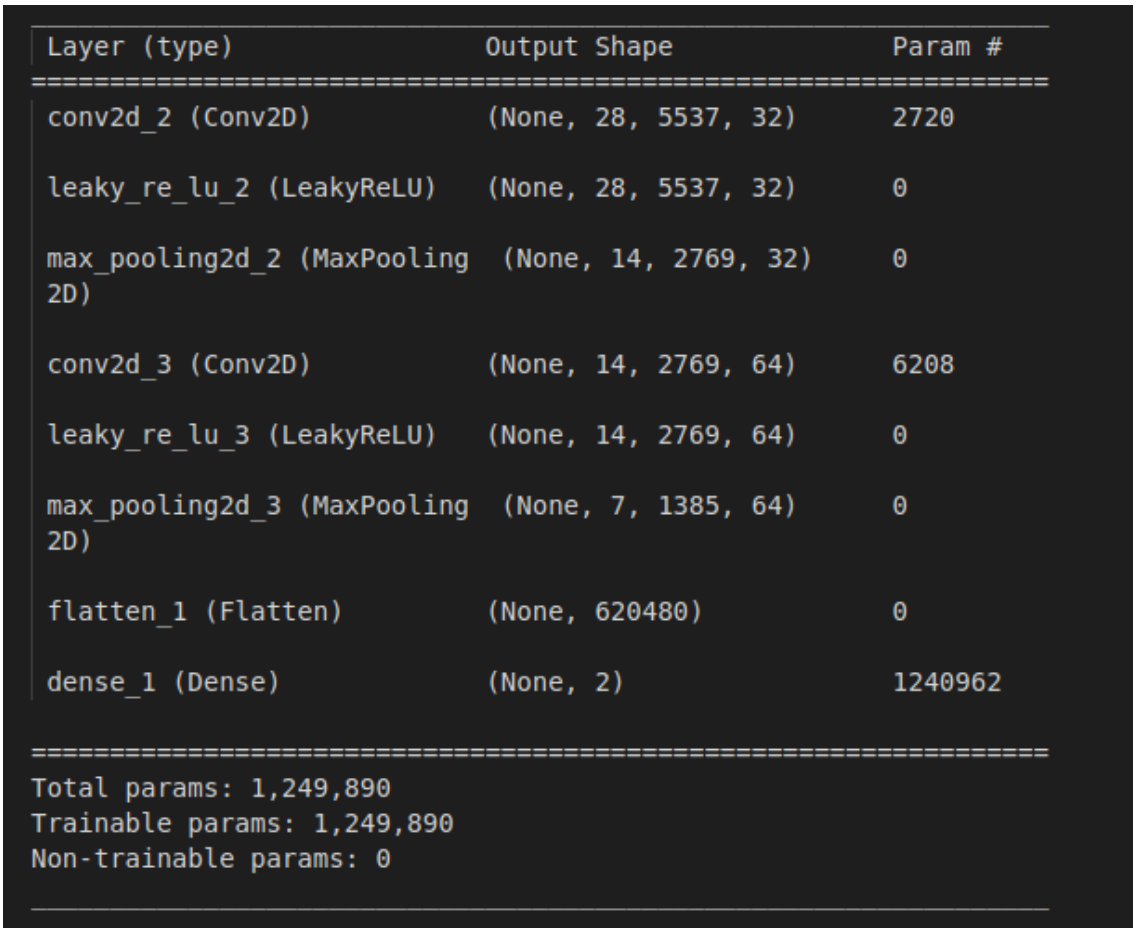
The total number of features is thus 28, where 8 are the VHSE descriptor values and 20 are the one-hot encoded amino acid characters. In addition to this, machine learning models like the Convolutional Neural Network (CNN) typically need to have a uniform input size across all data instanes, hence each protein in the dataset was padded to the length of the maximum length protein in the dataset. Padding in this manner means that numeric values needed to be chosen to represent the absence of data. In the case of the VHSE descriptors, the padding value was set to -999: while a common choice for padding value in machine learning settings is 0, this falls within the range of the existing VHSE descriptor values. To prevent the model from learning based on a value that does not reduce to the absence of data, the padding values for the 8 VHSE features were set far outside the range of possible values. In the case of one-hot encoding, a value of 1 infers the presence of data, so leaving the padding as 0 was deemed sufficient.

The final shape of the dataset is thus multi-dimensional. For each of the *n* instances of data, there are 28 features (rows) and *m* columns, where *m* is the length of the maximum length protein sequence in the input dataset. Thus the process of inferring features from the sequences in the dataset results in a derived dataset that is *n* x 28 x *m*. The labels for the n data instances were stored separately and were also one-hot encoded for the CNN training process.

*Methodology*

The process of classifying proteins is difficult by human standards. Where sequences are long, to process enough information to perform the task of classifying proteins presents a difficult task, especially given that the amino acids can appear to be randomly distributed to the human eye. Machine learning presents methods of numeric modeling that can learn and generalize said learning in circumstances that would be difficult for humans to perform in. Due to the complexity of the protein classification process, a complex model such as the CNN seems fit for the task.

The CNN structure chosen for this task was a simple sequential model in which subsequent layers of convolution, activation, and pooling are performed to reduce the size of the dataset. Information that is important to the classification process is maintained by increasing the logical complexity of the data through pooling. Due to the dimensionality of the input data, in this case two dimensions, the model completes with a flattening of the data after it has been processed by the sequential convolutional layers, and then passed to a dense layer that processes the final data via a set of activation functions and classifies it. The model structure chosen for this study is shown in Figure 1.



| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_2 (Conv2D) | (None, 28, 5537, 32) | 2720 |
| leaky_re_lu_2 (LeakyReLU) | (None, 28, 5537, 32) | 0 |
| max_pooling2d_2 (MaxPooling 2D) | (None, 14, 2769, 32) | 0 |
| conv2d_3 (Conv2D) | (None, 14, 2769, 64) | 6208 |
| leaky_re_lu_3 (LeakyReLU) | (None, 14, 2769, 64) | 0 |
| max_pooling2d_3 (MaxPooling 2D) | (None, 7, 1385, 64) | 0 |
| flatten_1 (Flatten) | (None, 620480) | 0 |
| dense_1 (Dense) | (None, 2) | 1240962 |

Total params: 1,249,890
Trainable params: 1,249,890
Non-trainable params: 0

***Figure 1:*** *The model and its layers. A two dimensional convolutional neural network is used, with two sets of convolution, acitivation, and pooling layers before flattening and densifying.*

The choice of layers in this case is arbitrary as the process of developing a model for the classification of UBL proteins is iterative, and as such at this stage the model structure is not known to have a specific impact on the performance of the model. In theory, more layers of convolution, activation and pooling can provide more opportunity for the model to learn and the optimal model might thus have a number of layers that is at least the magnitude of the unique features in the dataset (and one for all the one-hot encoded features). As this results in a model that would take a long time to train, a simple model of 2 layer sets (before flattening and densifying) was deemed to be sufficient for the purposes of investigating the performance of this classification approach.

In order to test the effectiveness of this classification process, several test iterations were performed. In all iterations, the data was first split in a percentage ratio of 80:20 for 20% of the data to be retained for validation, while the remaining 80% was split in the same ratio again for training and testing data. Validation was performed by using the model for prediction after training.

This study includes four iterations of testing, each producing its own model. In the first test iteration, where zinc-finger sequences were excluded for the sake of control, the data was resampled to 20% of the original dataset size in the same proportions of class frequencies as the input dataset. This resulted in a resampled dataset with a class-imbalance. The purpose of this resampling was merely to decrease the runtime, so that training could be completed in a more reasonable time-frame.

In the second test iteration, the dataset was resampled to 20% of the original dataset size and had a balance between non-UBL sequences and confirmed UBL sequences. This resampling used all the UBL sequences and included as many non-UBL sequences. The purpose of this test was to investigate the impact of the class imbalance on the accuracy of the classifier.

In the third test iteration, the best performing resampling technique was chosen and the data was processed so that the zinc finger sequences were shuffled into the non-UBL sequences. This was done to test the ability of the model to avoid zinc-finger false positives.

In the fourth and final test, the model was trained with only zinc-finger sequences as the known non-UBL proteins. This was performed to determine the model's ability to exclusively recognize potential false positives when there were no other negatives in the non-UBL class.

For a final step the four models that were developed, one from each iteration of testing, were loaded and passed a set of validation data that was randomly selected from the dataset. The classes were balanced and the zinc-finger negatives were shuffled in with the other negatives. The results of this entire process are reported on below.

**Results**

As can be seen in Figure 2, the training loss for all of the models was approaching convergence. This shows that the models were being trained to some recognizable degree where a classification on novel data could be expected to have some regularity. The models should thus be expected to have some form of usefulness pending further investigation of the error metrics. One interesting note here is that the datasets with non-UBL sequences containig zinc fingers, known to be a source of potential false positives, produced erratic loss during certain epochs.

In the case of the training accuracy in Figure 3, similar results are observed in that the classification accuracy of the training data begins to show possible convergance. The change in accuracy is less extreme for the model working with the unbalanced dataset, but this model too continues to improve in training accuracy. The training accuracy of all four models continues to improve beyond 90%, but the accuracy is still increasing at the $30^{th}$ epoch when the training ends.

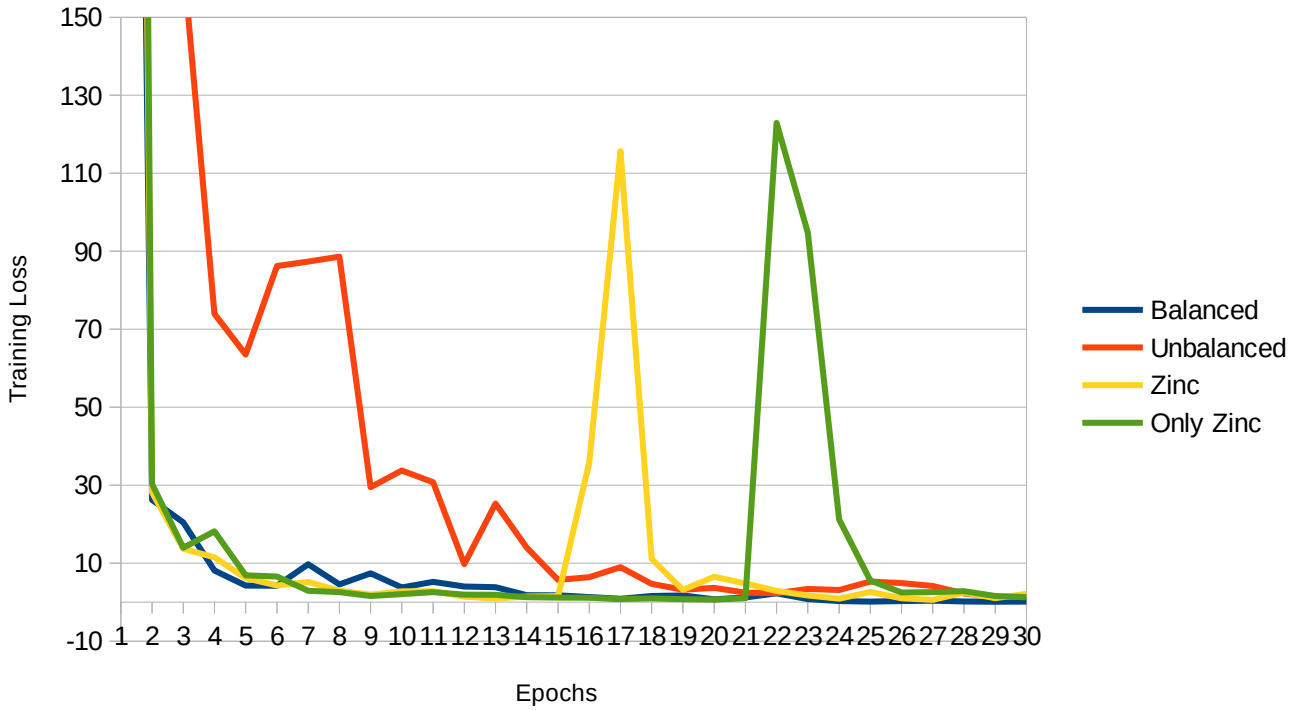



***Figure 2:*** *Training loss for the four compiled datasets. The loss definintely converges, although some variance in the zinc data causes extreme loss during certain epochs; this is reflected in the training accuracy of Figure 3.*

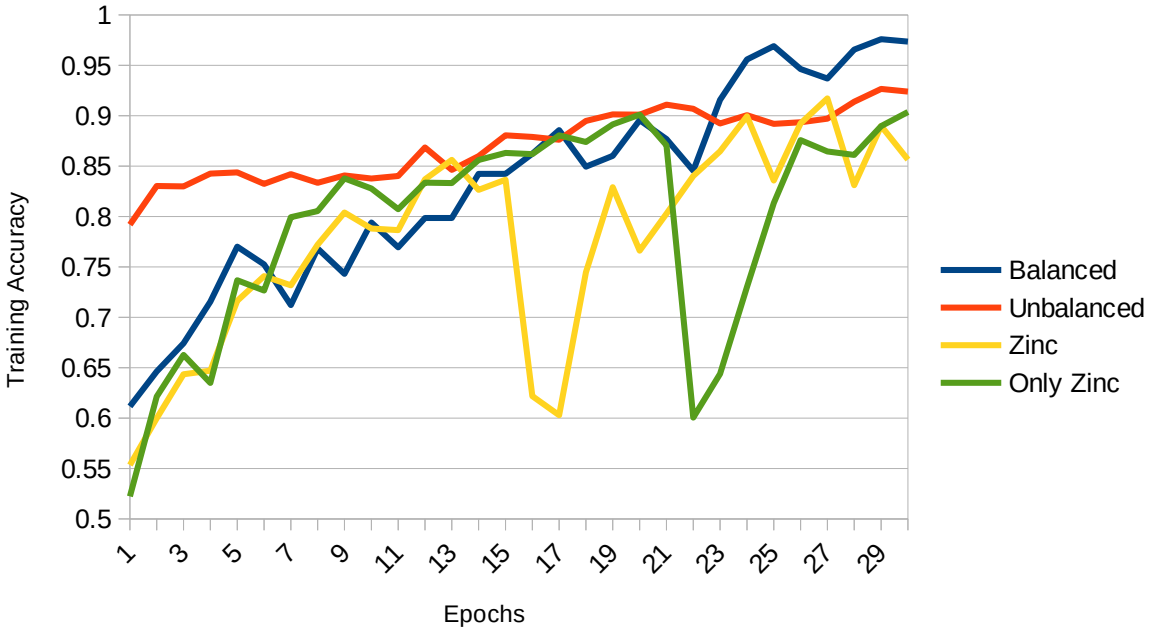



***Figure 3:*** *Training accuracy for the four compiled datasets. Variability in the accuracy achieved with zinc finger sequences in the training data causes extreme variation during certain epochs.*

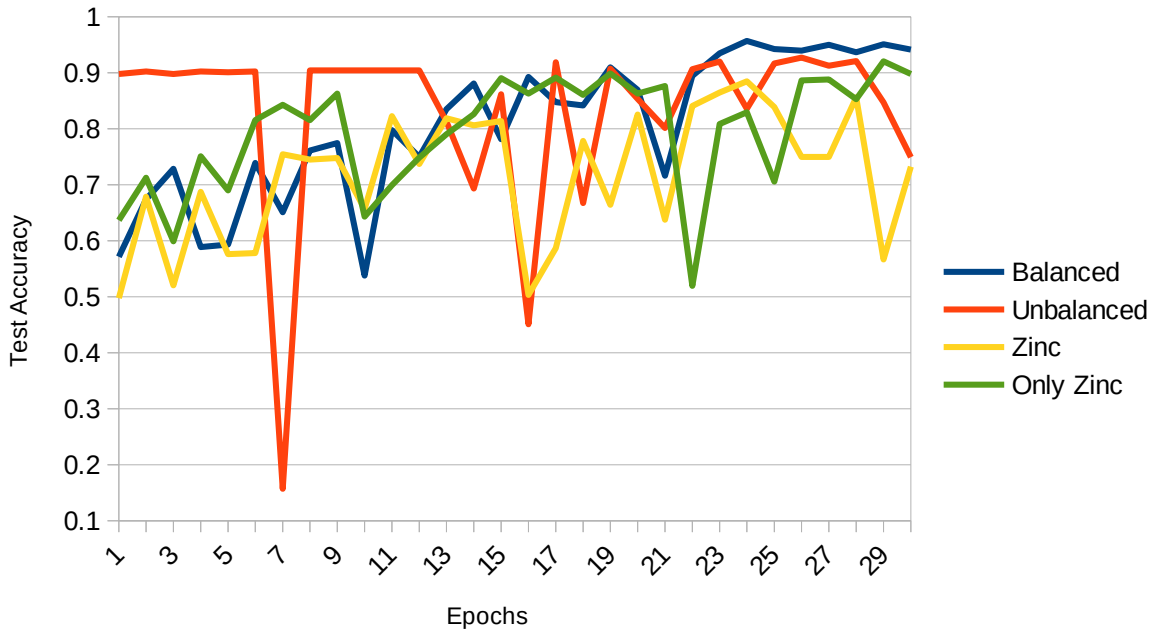



***Figure 4:*** *The test accuracy of the model on the four compiled datasets. As with the training loss and accuracy, this accuracy does appear to be increasing except in this case the unbalanced dataset is quite erratic.*

With the testing accuracy in Figure 4, a slightly different picture emerges. Uniquely for the model training with unbalanced data, the testing accuracy does not appear to improve but remains erratic across the epochs, even decreasing towards the last epoch. This trend is most likely not significant however and the test accuracy ranges between roughly 50% and 90%. For the balanced dataset that does not contain zinc finger negatives, there is a decreasing amount of variability in the test accuracy as the epochs progress. This trend is not reflected in the datasets that contain zinc finger negatives however, where the amount of variability does not decrease over the epochs. The test accuracy for the models trained on data including zinc finger negatives, along with the balanced dataset, does appear to be increasing despite the variability. The trend is towards 90% test accuracy and above, except perhaps in the model trained on data that had zinc finger negatives shuffled in with the other non-UBL sequences. This model does not reach 90% test accuracy and continues with large variability in the final epochs.

The validation accuracy in Figure 5 produced similar results across two tests. In the first test, validation data was reserved from the resampled data for each model as was decribed in the methodology. In the second test, a separate dataset was produced by resampling and passed to each of the models in turn. Both tests showed that the model trained on the balaned dataset produced the best results, while the next best results were produced by the model trained on the dataset with the only non-UBL sequences being the zinc finger sequences. The unbalanced dataset without zinc finger negatives, and the balanced dataset with zinc finger negatives shuffled in with the other non-UBL sequences, both produced comparable results in the two stages of validation. Their accuracy was between 75% and 80%.

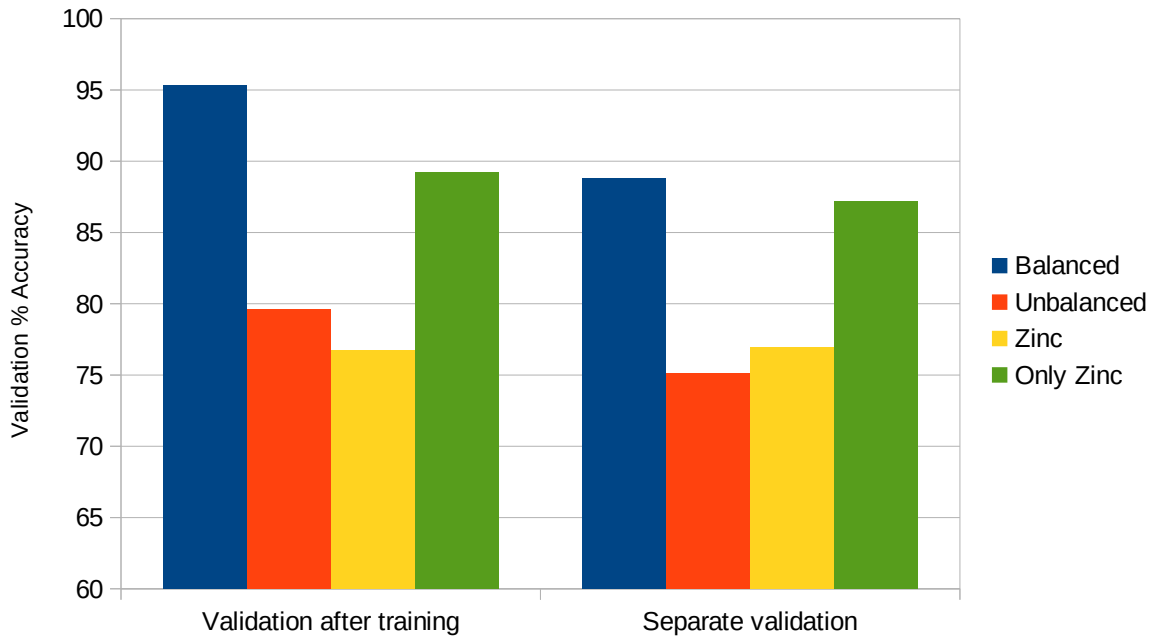



***Figure 5:*** *The accuracy of each model on validation data. In one instance, the validation data was reserved from the sampled dataset created at the time of model training/testing; in another instane, separate validation was performed using a dataset that was sampled once all the models had been created.*

Finally, observing the predicted versus the observed results in the form of a confusion matrix can reveal interesting trends across the models trained with the different datasets. In Table 3 below, a series of four confusion matrices is presented from the final separate validation data that was presented to each of the models. With the unbalanced dataset, the dominant non-UBL class predictably receives the most predictions. This is corrected in the dataset that was balanced. In the dataset with the zinc finger shuffled in, the model struggles with false positives. In the model trained using only zinc fingers as negatives, the performance improves markedly.

***Table 3:*** *A series of confusion matrices for the predictions produced by the four models.*

| | Predicted | |
|---|---|---|
| Observed | Not UBL | UBL |
| Not UBL | 2806 | 414 |
| UBL | 1226 | 2003 |

*Unbalanced data without zinc finger*

| | Predicted | |
|---|---|---|
| Observed | Not UBL | UBL |
| Not UBL | 3211 | 9 |
| UBL | 816 | 2413 |

*Balanced data without zinc finger*

| | Predicted | |
|---|---|---|
| Observed | Not UBL | UBL |
| Not UBL | 1864 | 1356 |
| UBL | 135 | 3094 |

*Balanced data with zinc finger shuffled in*

| | Predicted | |
|---|---|---|
| Observed | Not UBL | UBL |
| Not UBL | 2825 | 395 |
| UBL | 433 | 2796 |

*Balanced data with zinc finger as only negative*

**Discussion**

This initial series of tests reveals a number of interesting traits of the dataset and the model used to classify UBL protein sequences in differentiation from non-UBL protein sequences. Firstly, the model predictably performs bettter when the dataset is balanced. In the case where the model was trained on data that was not balanced between the classes, the dominant class was classified more frequently than the less numerous class. Interestingly, this pattern came through in the results from the balanced dataset as well. The ratio of predictions between the classes was remarkably similar to the unbalanced dataset, however the difference here was that the accuracy with the balanced dataset was higher.

Secondly, the results from the model that was trained on non-UBL sequences that had the zinc finger sequences shuffled in performed similarly poorly to the model trained on the unbalanced dataset. In this case however, the false positive qualities of the zinc finger sequences clearly confused the model and it was unable to correctly classify many of the non-UBL proteins. The number of false positives in this test was the highest of all the tests by a factor of four.

Thirdly, the results from the model that was trained on non-UBL sequences that were sourced only from the zinc-finger dataset were far better than the results from the unbalanced data and the data with the zinc finger sequences shuffled in to the non-UBL sequences. While the false positive rate was similar to the true negative rate at roughly 13%, the accuracy of this model on the separate validation data was ~87%. As was seen with the training and testing data, the accuracy had not yet converged with this model and it is likely that there will be an improvement in the validation accuracy with a greater number of training epochs on this dataset.

What we can deduce from these results is that the model used in this study performs best when being trained on data that is made up of clearly seperable classes. When the sequences with a zinc finger were shuffled in to the other non-UBL sequences, the false positive rate was 30.47%. This rate, along with the poor accuracy, is not acceptable. It seems highly likely from this series of tests that a model that is trained on three classes would perform the best with all of the three datasets, where UBL protein sequences are one class, non-UBL sequences without a zinc finger are another class, and non-BUL sequences *with* a zinc finger are a third class. If the sequences that are used to train the data can be separated in this manner, it is likely that a model can be produced which will perform to a high degree of accuracy when presented with novel sequences.

With regards to the erratic loss and accuracy in figures 2,3 and 4, it seems likely that the cause of this is due to certain sequences that have a zinc finger are difficult for the model to classify. While the inner workings of the model were not visible so this could not be exactly deduced, this erratic pattern only occurred with the zinc finger sequences. Further work could be done to try to identify which of the zinc finger sequences are particularly problematic and they could be isolated for further testing.

In order to improve the model further, certain steps can be taken. One obvious measure would be to increase the number of epochs. At this stage, only 30 epochs were used because the training time of over 4 hours per model was prohibitive. In future testing, more epochs should be used to seek convergence in the train/test accuracies as well as loss.

Another obvious improvement that could be made is to use more features. Unlike other work[5] that has been done which includes a limited number of features, future work with CNN modeling could include more physical and chemical features of amino acids like hydrophobicity, polarity, and more. This could bolster the number of features in the derived dataset[*]. Increasing the number of features in a thoughtful manner could serve to increase the amount of variance in the data with respect to the predictor, making the model more accurate.

If possible, more sequences should be acquired to improve the size of the dataset. With only 3740 UBL sequences being available, the number of sequences that can be used for training with a balanced dataset becomes severely limited. In order to improve the ability of the model to classify correctly, a larger sample size would be useful.

Consideration of the model itself is also appropriate here. The model layers were abitrarily chosen because of the cost of training each of the models. Due to the nature of the results, with trends being clearly visible, two layer sets of convolution, activation and pooling were thought sufficient for this series of tests. Future testing could include more layers within the model, increasing the ability of the model to recognise more complex features within the dataset. With a greater number of layer sets in the model, the accuracy would most likely improve.

## Conclusion

This study presented a novel approach to classifying Ubiquitin Ligase proteins versus non-UBL proteins. The model chosen was a Convolutional Neural Network that included a series of layers that examined the dataset as a two dimensional series of features for every sequence. The model structure was put through a series of four separate tests, with the models developed being then subjected to further validation. The performance indicates that further work is warranted and could lead to the publication of a novel classification method. The accuracies in the best case exceeded 85% and showed promise for performing the required task.

While the dataset development process and the model structure presented here were not meant to produce penultimate accuracy, the results they did produce were inviting of further investigation. Future work will most likely produce a model that can perform the task required to a high degree of accuracy.

* https://www.sigmaaldrich.com/US/en/technical-documents/technical-article/protein-biology/protein-structural-analysis/amino-acid-reference-chart

**References:**


1. Chandonia, J.M., Guan, L., Lin, S., Yu, C., Fox, N.K. and Brenner, S.E., 2021. SCOPe: improvements to the structural classification of proteins–extended database to facilitate variant interpretation and machine learning. *Nucleic acids research*.
2. Uversky, V., 2014. *Protein families: relating protein sequence, structure, and function*. John Wiley & Sons.
3. Strodthoff, N., Wagner, P., Wenzel, M. and Samek, W., 2020. UDSMProt: universal deep sequence models for protein classification. *Bioinformatics*, *36*(8), pp.2401-2409.
4. Wang, D., Ma, L., Wang, B., Liu, J. and Wei, W., 2017. E3 ubiquitin ligases in cancer and implications for therapies. *Cancer and Metastasis Reviews*, *36*(4), pp.683-702.
5. McDermott, J.E., Cort, J.R., Nakayasu, E.S., Pruneda, J.N., Overall, C. and Adkins, J.N., 2019. Prediction of bacterial E3 ubiquitin ligase effectors using reduced amino acid peptide fingerprinting. *PeerJ*, *7*, p.e7055.
6. Arnold, R., Brandmaier, S., Kleine, F., Tischler, P., Heinz, E., Behrens, S., Niinikoski, A., Mewes, H.W., Horn, M. and Rattei, T., 2009. Sequence-based prediction of type III secreted proteins. *PLoS pathogens*, *5*(4), p.e1000376.
7. Xie, J., Xu, Z., Zhou, S., Pan, X., Cai, S., Yang, L. and Mei, H., 2013. The VHSE-based prediction of proteasomal cleavage sites. *PLoS One*, *8*(9), p.e74506.