Multi-Label Prediction of Protein Subcellular Localizations through Machine Learning and State-of-the-art Structural Embeddings

Alfred Braidy

1/16/2024

1. Abstract

A polypeptide, more commonly referred to as a protein, is a macromolecule consisting of linear sequences of organic compounds called amino acids joined together by covalent bonds. Proteins carry different functions in living organisms, which are often associated with their location inside eukaryotic cells. Experimental approaches for localizing proteins such as green fluorescent protein and microscopic detection are costly and time-consuming. Thus, computational methods have been adopted to facilitate this process by generating accurate predictions. This research explores various machine learning models to predict protein subcellular localization using UniProt's Swiss-Prot database and Meta's high-performing ESM2 structural embeddings. We extracted our dataset's protein embeddings from ESM2's model code publicly posted on GitHub. Multiclassification Scikit-learn models were trained and tested to achieve optimal predictions. Our best-performing model, a weighted 4-Nearest Neighbors Classifier, yielded a 57% accuracy and a MacroF1 score of 66%. Our classifier outperformed past state-of-the-art algorithms (such as Deeploc2.0) on several but not all metrics. The results obtained demonstrate the potential of machine learning in producing unprecedented advancements in the proteomics field.

2. Introduction

While cells are considered the basic units of life, the proteins synthesized inside them fulfill nearly all the necessary functions to sustain living organisms. Those chains of amino acids catalyze internal biochemical reactions, mobilize intracellular signals, maintain tissue structure, transport vital molecules, and regulate gene expression [1]. Moreover, the functions of proteins are closely linked to the subcellular environments they operate in which dictate the availability of molecular interaction partners [2]. Different functional proteins lie in one or more distinctive organelles. As such, identifying protein subcellular localization has become crucial for proteomics research. The ability to accurately predict which cellular compartment each protein resides in can expand our knowledge of protein functional annotation and biological pathways as well as enhance a plethora of medical therapies, notably drug design and delivery.

For this research, protein sequences were represented as a string of letters according to the order and nature of their amino acid chains. Each letter denotes one of the 20 naturally occurring amino acids making up all existing proteins. The input data utilized is thus language data. According to the universal amino acid nomenclature, the one-letter codes are expressed as seen in Table 1 [3].

Table 1: Amino acids and their corresponding one letter codes

Amino acid name	One letter code	Amino acid name	One letter code
Alanine	A	Leucine	L
Arginine	R	Lysine	K
Asparagine	N	Methionine	M
Aspartic acid	D	Phenylalanine	F
Cysteine	C	Proline	P
Glutamic acid	E	Serine	S
Glutamine	Q	Threonine	T
Glycine	G	Tryptophan	W
Histidine	H	Tyrosine	Y
Isoleucine	I	Valine	V

This project aims to label each of the protein sequences with one or more of ten possible eukaryotic cellular compartments, making this task both a multiclassification and multilabel problem. Moreover, this task involves supervised learning which means that the amino acid chains are paired with their corresponding localization. The ten subcellular environments adopted consist of the following sublocations [4]:

- Nucleus: Envelope, inner and outer membrane, matrix, lamina, chromosome, nucleus speckle
- Cytoplasm: Cytosol and cytoskeleton
- Extracellular: Outside the cell membrane
- Mitochondrion: Envelope, inner and outer membrane, matrix, intermembrane space
- Cell membrane: Apical, apicolateral, basal, basolateral, lateral, cell membrane, cell projection
- Endoplasmic reticulum (ER): ER membrane and lumen, microsome, rough ER, smooth ER, Sarcoplasmic reticulum
- Plastid: Plastid membrane, stroma, and thylakoid*
- Golgi apparatus: Golgi apparatus membrane and lumen
- Lysosome/Vacuole: Contractile, lytic and protein storage vacuole, vacuole lumen and membrane, lysosome lumen and membrane
- Peroxisome: Peroxisome matrix and membrane

*Plastids are only found in plant cells

The aforementioned organelles can be visually represented in the eukaryotic cell diagrams found in Figures 1 and 2 depicting animal cell and plant cell structure, respectively.

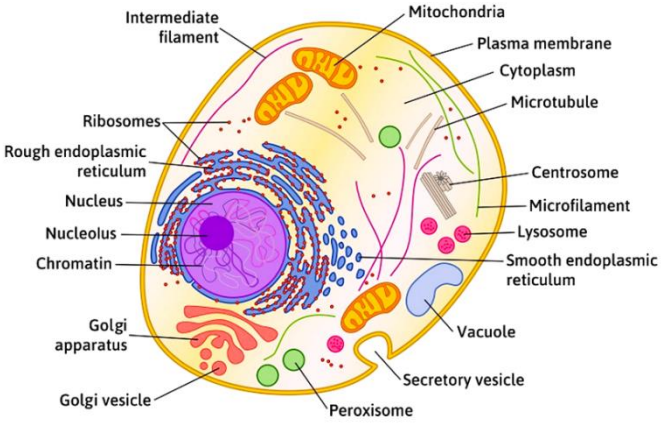

Figure 1: Animal Cell diagram
Image source: OpenStax via Wikimedia Commons

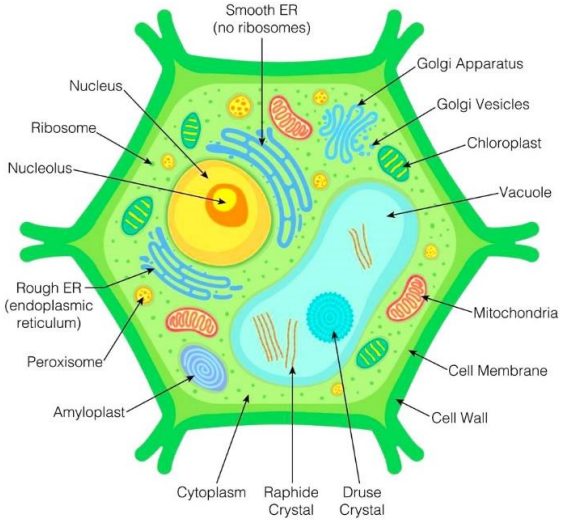

Figure 2: Plant Cell diagram
Image source: iStock

3. Background

Through numerous experimental methods, a considerable number of protein subcellular localizations have been verified and stored in online databases, including Swiss-Prot which will be adopted in this project. Since experimentally identifying protein subcellular localization is laborious, many machine learning techniques have been developed to efficiently generate predictions. Past computational approaches, such

as DeepLoc1.0, have successfully minimized the reliance on homology between amino acid sequences when processing the input data [5]. This breakthrough is particularly relevant when dealing with novel proteins deprived of annotated homologues. This tool was later enhanced to include multi-localization predictions. The upgraded version, Deeploc2.0, could thus correctly assign multiple labels for proteins located in more than one organelle, a major milestone in advancing this field [4].

In August 2022, Meta's protein structure prediction AI ESM-2 (Evolutionary Scale Modeling) was developed to generate the 3D structure of proteins based on their amino acid sequences [6]. ESM2 outperformed all tested single-sequence protein language models [6]. Although ESM2 doesn't address our problem directly, we can benefit from their advancements by utilizing their highly accurate structural embeddings. These embeddings transformed the amino acid sequence data into specific representations which ESM2 used to train their successful model. Our research will leverage ESM2's structural embeddings and input them into various machine learning models in an attempt to reach optimal results.

4. Data collection

The Swiss-Prot dataset was used to train and test the machine learning models. This database is extracted from The Universal Protein Resource (UniProt), the most comprehensive catalog of protein sequence and functional annotation [7]. Deeploc2.0 filtered the sequences and localization annotations using the following criteria: eukaryotic, not fragments, encoded in the nucleus, more than 40 amino acids, and experimentally annotated subcellular localizations [4]. The data included each protein's UniProt accession number (ACC), its amino acid sequence as one-letter codes, its eukaryotic kingdom, and its presence or absence from each organelle using binary representation. As shown in Figure 3, these proteins are unequally distributed among the ten subcellular locations mentioned in the introduction.

Due to computational constraints on Google Colab notebooks, the dataset was further filtered to exclude sequences with more than 700 amino acids. This reduced the number of samples from 28303 to 21592 while maintaining a similar subcellular localization distribution as seen in Figure 4. It is worth noting that the majority of proteins in the dataset (around 52%) are found in the nucleus and cytoplasm which is where protein synthesis takes place.

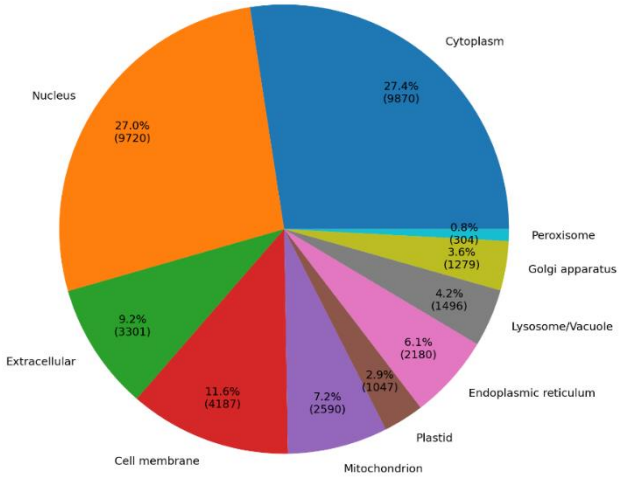

Figure 3: Original dataset location count

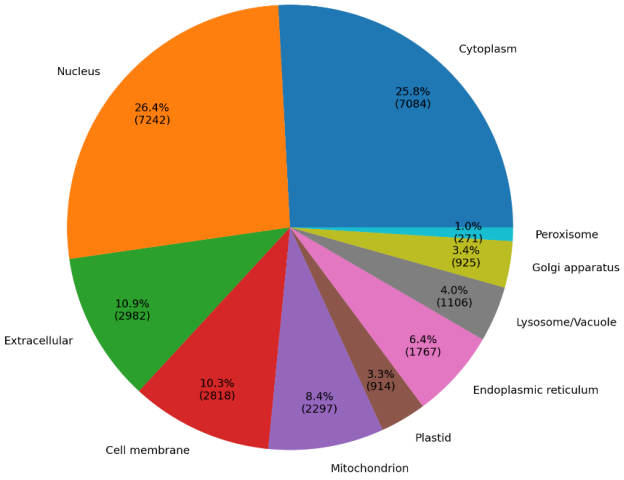

Figure 4: Filtered dataset location count

As seen in Figure 5, the number of localizations per protein in the dataset ranges from one to five. An inversely proportional relationship can be observed where the number of proteins decreases drastically as the number of localizations per protein increases, making 76% of the proteins in the dataset single-localization sequences. Lastly, the data was split into training and testing datasets, with the training dataset comprising 80% of the protein samples and the testing dataset comprising the remaining 20%.

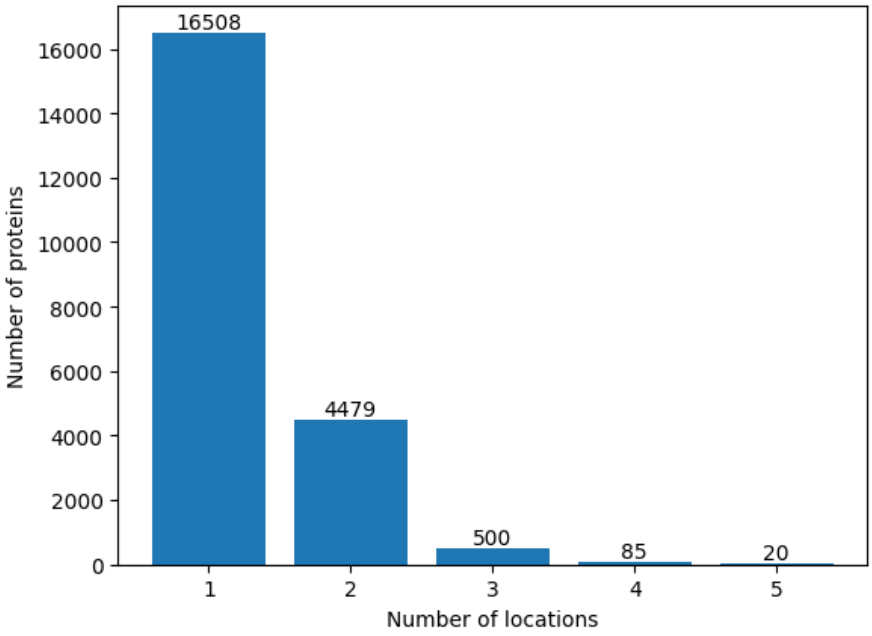

Figure 5: Filtered dataset label count

5. Methodology/Models

5.1. Feature Engineering

Before proceeding further, we dropped the eukaryotic kingdom feature since it is not relevant to this project. We then extracted ESM2's structural embeddings using GitHub, a code-hosting platform for version control and collaboration [6]. The model posted by Meta Research takes the amino acid sequences - strings of one-letter codes in our dataset - as input and generates corresponding structural embeddings which get stored as .pt files named according to their ACC. The embeddings were loaded to the Google Colab notebook using PyTorch which resulted in 21592 rows of horizontal vectors comprising 1280 distinct numbers each – a 21592 x 1280 dataframe. These were then paired with their respective localizations, represented through multi-label binarization. Figure 6 shows the structure of the final training and testing data frames split according to an 80:20 ratio.

Training DataFrame

	0	1	2	3	4	5	6	7	8	9	...	Cytoplasm	Nucleus	Extracellular	Cell membrane	Mitochondrion	Plastid	Endoplasmic reticulum	Lysosome/Vacuole	Golgi apparatus	Peroxisome
9440	0.005717	0.001396	-0.111107	0.088429	-0.193733	0.017546	0.191072	-0.092863	-0.117966	0.138295	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
14049	0.049312	-0.120122	-0.062257	0.112093	-0.009904	0.012469	0.062741	0.031535	0.008083	0.160591	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19983	0.008837	-0.037810	-0.047155	0.021474	-0.054411	-0.031371	0.044677	-0.128819	-0.043257	0.048458	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1949	0.033241	-0.054629	-0.006156	0.014279	-0.035365	-0.069008	0.072206	0.075772	-0.097228	-0.015825	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18629	0.035499	-0.067486	0.047996	0.011795	-0.009502	-0.104738	0.006745	-0.059346	0.006764	0.031353	...	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10955	0.036602	0.041788	-0.010131	0.104212	0.097831	-0.073354	0.036823	0.004947	0.016721	0.075620	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17289	0.009104	-0.055496	-0.006904	0.065672	-0.037068	-0.047591	0.059747	0.063899	-0.081333	0.098402	...	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0
5192	0.068544	-0.047457	-0.022562	0.103369	-0.003527	-0.054030	0.111705	-0.105286	-0.008416	0.127380	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12172	0.089400	-0.178048	-0.079587	0.149839	-0.256418	-0.084755	0.262423	-0.024896	-0.191286	0.069707	...	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
235	0.061255	-0.091330	0.029050	0.138085	0.002738	-0.093879	0.144411	-0.023404	0.077582	0.278129	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

17273 rows × 1290 columns

Testing DataFrame

	0	1	2	3	4	5	6	7	8	9	...	Cytoplasm	Nucleus	Extracellular	Cell membrane	Mitochondrion	Plastid	Endoplasmic reticulum	Lysosome/Vacuole	Golgi apparatus	Peroxisome
20737	0.009154	-0.050906	0.011091	0.050690	-0.092051	-0.074400	0.044424	-0.005939	0.015839	0.107916	...	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
15542	0.040025	-0.077004	-0.027636	-0.013467	0.012374	-0.050603	-0.032668	0.118081	-0.006256	-0.040046	...	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6120	0.019492	-0.028200	0.023915	0.034342	-0.004484	-0.081777	0.029480	-0.145218	-0.040961	0.013290	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
12332	-0.002126	-0.075776	-0.025911	0.013035	0.012650	-0.129954	0.045080	-0.015991	-0.084072	-0.009602	...	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0
1113	0.029522	-0.069974	-0.016469	0.106951	-0.003870	-0.054497	0.029472	-0.100419	0.028103	0.101492	...	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
12045	0.035750	0.010806	0.040243	-0.004151	-0.088175	-0.118272	0.056393	-0.043082	-0.003733	0.090350	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
20359	0.044231	-0.048476	0.038304	0.036690	0.087888	-0.117916	-0.031000	0.009613	0.027994	0.011437	...	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
19031	-0.022484	0.001373	-0.045123	0.013904	-0.069985	-0.054530	0.025948	-0.067359	-0.018641	0.051222	...	1.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
17807	-0.005125	-0.114639	0.000438	0.095862	-0.066290	-0.005393	0.138327	-0.028780	0.013417	0.152424	...	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1993	0.023317	-0.046508	0.036072	0.073995	0.010867	-0.036996	-0.029161	0.108787	0.054726	0.012186	...	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

4319 rows × 1290 columns

Figure 6: First and Last 5 rows of Final Training and Testing Dataframes

5.2.<u>Model Development</u>

We first utilized models from Scikit-learn, a robust library for machine learning in Python, to test the efficiency of ESM2's structural embeddings [8]. In order to adopt these models, the training and testing dataframes were turned into NumPy arrays and each split into input X, the structural embeddings, and labels y, the subcellular localization(s). The data was tested with the following multi-classification models: Decision Tree, Extra Trees, Random Forest, and Nearest Neighbors.

Decision Trees represent a supervised learning approach that predicts the value of a target variable by grasping simple decision rules derived from the features within the dataset [8]. The Extra Trees classifier utilizes a meta estimator, fitting multiple randomized decision trees (referred to as extra-trees) on different sub-samples of the dataset and employing averaging to enhance predictive accuracy and regulate overfitting [8]. Similarly, Random Forests is another averaging algorithm based on randomized decision trees. This classifier however chooses the optimum node split instead of relying on randomness during the construction of a tree [8]. Finally, the Nearest Neighbors classification involves finding a predetermined number of training samples closest in distance to the testing point and predicting the label from these "nearest neighbors" based on a simple majority vote [8]. While the basic Nearest Neighbors classification uses uniform weights, this feature can be altered to give more weight to the nearer neighbors and thus increase their contribution to the prediction [8].

5.3.Evaluation Metrics

Since accuracy does not take into account the class distribution, it can be misleading with imbalanced datasets. To paint a more comprehensive image of the performance of machine learning models, we utilized classification reports that display the precision, recall, and F1 score for the model [9]. Before defining each of these metrics, we need to outline evaluation factors:

- True Positive (TP): number of positive samples classified correctly
- True Negative (TN): number of negative samples classified correctly
- False Positive (FP): number of actual negative samples inaccurately classified as positive
- False Negative (FN): number of actual positive samples inaccurately classified as negative

In our problem, "positive" represents the presence of a protein in the subcellular localization (denoted by 1), and "negative" represents its absence (denoted by 0). In classification problems, precision refers to the ratio of TP to the sum of true and false positives [10]. Additionally, recall is calculated by dividing the TP by the sum of TP and FN [10]. Finally, the F1 score can be defined as the weighted harmonic mean of precision and recall, which is derived by dividing the number of values in a dataset by the sum of reciprocals of each value [10]. The F1 score reflects the model's ability to identify both positive and negative samples. All three of these metrics range from a worst score of 0 to a best score of 1 [9]. The following equations define Precision, Recall, and F1 score in terms of the evaluation factors aforementioned [10]:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$

$$F1 = 2 \times \frac{Precision + Recall}{Precision \times Recall} = \frac{TP}{TP + \frac{1}{2}(FP + FN)}$$

For multiclass classification, the F1 score needs to be averaged since we have multiple scores each corresponding to one class. To compute the overall F1 score for the dataset, we can employ either Micro or Macro averaging. While the Macro F1 score represents the unweighted mean of the per-class F1 scores, the Micro F1 score is calculated by applying the F1 formula to the total number of TP, TN, FP, and FN across all classes [11]. Thus, Macro F1 gives equal weight to all classes and Micro F1 gives equal weight to all samples in the data. For imbalanced datasets like Swiss-Prot, Macro F1 is a better reflection of model performance since Micro F1 would be biased toward the class with most samples – nucleus and cytoplasm in our case.

Another useful metric for classification is Matthew's Correlation Coefficient (MCC). MCC provides a balanced measure of classification performance by considering all four evaluation factors in its formula unlike other classification metrics [12]. MCC values range from -1 (inverse prediction) to 1 (perfect prediction), with 0 representing a random prediction [12]. Its formula can be written as follows:

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

6. Results and Discussion

Among the Sklearn classifiers, Decision Trees was the worst-performing model while Random Forest and Extra Trees had a slightly better performance, respectively. As seen in Figure 7, the Nearest Neighbors classifier exhibited the best performance by far with a 56.9% accuracy. Similar results were observed before hyperparameter tuning, so we only experimented with the Nearest Neighbors model with not much room for improvement for the other classifiers. While an increasing number of neighbors (starting from 1) resulted in a decreasing accuracy with the weight set to "uniform," assigning weights proportional to the distance of neighboring data points altered this pattern. The highest accuracy was subsequently reached when switching *weights* to "distance" and setting *n_neighbors* to 4.

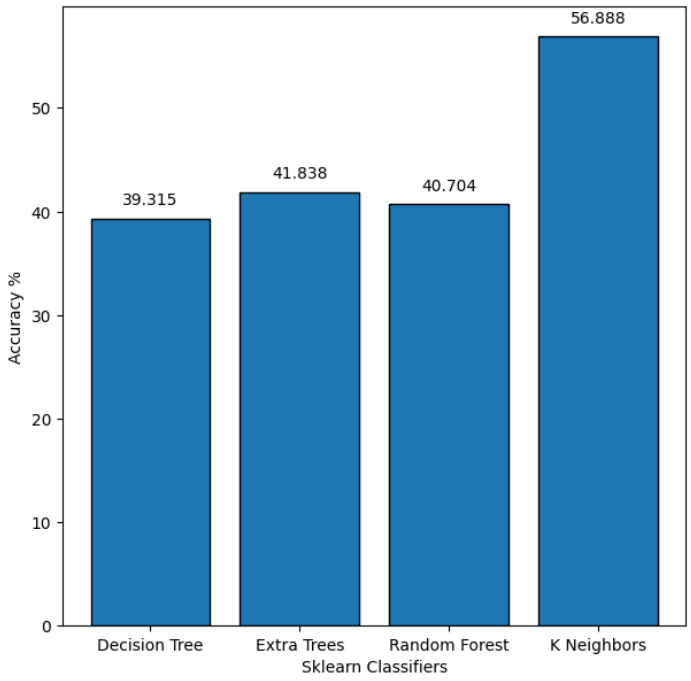

Figure 7: Sklearn Models Accuracy after Hyperparameter Tuning

All the metrics mentioned in the Methodology section were utilized to evaluate the performance of the Nearest Neighbors classifier. It is worth mentioning that the accuracy metric here requires the exact location(s) to be predicted for each sample, so a prediction with only one of multiple actual positive labels as output would be considered inaccurate. Figure 8 displays each localization's 2x2 confusion matrix: a table layout displaying the number of accurate and inaccurate instances from the model's predictions. Confusion matrices help measure a classification model's performance by including the TP, TN, FP, and FN values which are used to calculate evaluation metrics [10]. Per-class metrics were computed individually from these matrices before then averaging the scores for the whole model. The results were compiled in Table 1 which also includes the values obtained by Deeploc2.0 for the same metrics and dataset (Swiss-Prot). It is worth noting that Deeploc2.0 included the per-class MCC values in their paper instead of averaging them and did not disclose the precision and recall values.

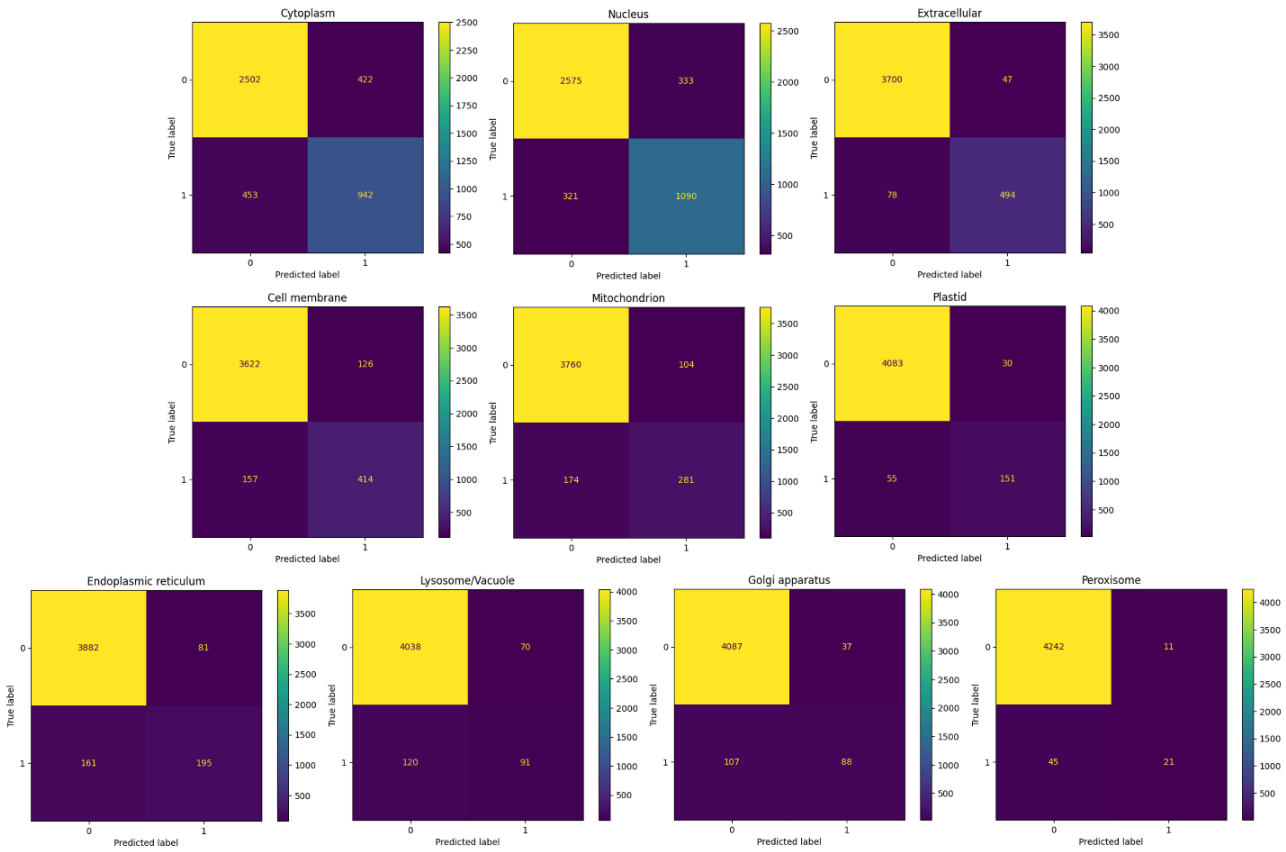

Figure 8: Confusion Matrices for each Localization obtained by the Nearest Neighbor Classifier

As seen in Table 2, our Nearest Neighbors classifier outperformed Deeploc2.0 in a considerable number of metrics including accuracy. The classifier notably matched Deeploc2.0's Macro F1 score of 0.66, a valuable indicator of model performance for our imbalanced dataset. On the other hand, the marginally lower Micro F1 score of 0.72 could be due to the data filtering we conducted which altered class distribution. As for MCC, the Nearest Neighbors classifier outperformed Deeploc2.0's transformer model in five localizations or half of the total classes in the dataset. The most substantial favorable difference was observed for the "Lysosome/Vacuole" and "Golgi apparatus" localizations.

Moreover, the average number of predicted labels by our classifier was 1.16 per protein compared to an actual average of 1.26 – an underestimation of 0.1 labels. This error may be due to the dataset being largely skewed towards proteins with a single localization. For this metric, Deeploc2.0's model accurately predicted an average of 1.27 labels per protein for an actual average of 1.27 in their unfiltered dataset. Despite the disadvantage faced by an imprecise average number of labels predicted and a smaller number of sequences to train due to a 24% reduction of the dataset, the Nearest Neighbors classifier was remarkably still able to achieve a higher accuracy than Deeploc2.0's transformer model.

Table 2: Nearest Neighbors Classifier and Deeploc2.0 model performance evaluated on different metrics

Evaluation Metric	Model	
	Deeploc2.0	Nearest Neighbors Classifier
Accuracy	0.55	**0.57**
MicroF1	**0.73**	0.72
MacroF1	**0.66**	**0.66**
MCC per location:		
Cytoplasm	**0.62**	0.53
Nucleus	**0.69**	0.66
Extracellular	0.85	**0.87**
Cell membrane	0.66	**0.71**
Mitochondrion	**0.76**	0.64
Plastid	**0.90**	0.77
Endoplasmic reticulum	0.56	**0.59**
Lysosome/Vacuole	0.28	**0.47**
Golgi apparatus	0.36	**0.55**
Peroxisome	**0.56**	0.45

Bold values indicate the best score

7. Conclusion

In our research, we sought to maximize the accuracy of protein subcellular localization predictions from amino acid sequences by leveraging ESM2's high-performing structural embeddings. The dataset utilized, a filtered version of Swiss-Prot, consisted of proteins with one or more assigned organelle and a total of 10 possible localizations. To accomplish this multiclassification and multilabel task, we trained and tested several machine learning models including Decision Tree, Extra Trees, Random Forests, and Nearest Neighbors. Our best-performing model, the 4-Nearest Neighbors classifier which prioritizes closer data points in making predictions, achieved an accuracy of 57% and a MacroF1 score of 66% after hyperparameter tunning. This model outperformed Deeploc2.0 in numerous metrics including accuracy. It also closely matched Deeploc2.0's F1 scores but fell short in the MCC scores for half the organelles. These figures highlight the potential of ESM2's structural embeddings in providing valuable localization knowledge for the considerable number of uncharacterized proteins.

Future research in this domain could make use of the whole Swiss-Prot dataset and process longer chains of protein with more computing power. Another suggestion would be to evaluate the embeddings on deep learning algorithms which may optimize predictions even more. An increasing accuracy in predicting protein subcellular localization is of utmost importance in proteomics research. Successful outcomes of this research study could engender significant advancements in drug development, disease research, and the biomedical field as a whole.

Acknowledgments

I would like to thank the Inspirit AI program for providing me with the opportunity to conduct this research project. I would also like to specifically thank my mentor Ayush Pandit for his guidance throughout the process of writing this paper and his help with all the problems I encountered.

References

1. Libretexts, "Functions of protein," Medicine LibreTexts, Available at: https://med.libretexts.org/Courses/Metropolitan_State_University_of_Denver/Introduction_to_Nutrition_(Diker)/06%3A_Proteins/6.05%3A_Proteins_Functions_in_the_Body.
2. Scott, M. S., Calafell, S. J., Thomas, D. Y., & Hallett, M. T. (2005). Refining protein subcellular localization. *PLoS Computational Biology*, *1*(6). https://doi.org/10.1371/journal.pcbi.0010066
3. J. A. Dobado, "One-letter and three-letter codes for amino acids," Chemistry Online, https://www.chemistry-online.com/generalities/one-letter-and-three-letter-codes-for-amino-acids/.
4. Thumuluri, V. *et al.* (2022) *DeepLoc 2.0: Multi-label subcellular localization prediction using protein language models*, *OUP Academic*. Available at: https://academic.oup.com/nar/article/50/W1/W228/6576357.
5. Almagro Armenteros, J.J. *et al.* (2017) *DeepLoc: Prediction of protein subcellular localization using Deep Learning*, *OUP Academic*. Available at: https://academic.oup.com/bioinformatics/article/33/21/3387/3931857.
6. Z. Lin *et al.*, "Evolutionary-scale prediction of atomic level protein structure with a language model," *bioRxiv*, 2022. doi:10.1101/2022.07.20.500902
7. The UniProt Consortium (2017). UniProt: the universal protein knowledgebase. *Nucleic acids research*, *45*(D1), D158–D169. https://doi.org/10.1093/nar/gkw1099
8. Pedregosa, F. *et al.* (2011) *Scikit-learn: Machine Learning in {P}ython*, 12, pp. 2825–2830. Available at: http://jmlr.org/papers/v12/pedregosa11a.html.
9. *Classification report* (2019) *Yellowbrick: Machine Learning Visualization*. Available at: https://www.scikit-yb.org/en/latest/api/classifier/classification_report.html.
10. Bajaj, A. (2023) *Performance metrics in machine learning [complete guide]*, *neptune.ai*. Available at: https://neptune.ai/blog/performance-metrics-in-machine-learning-complete-guide.
11. Allwright, S. (2022) *Micro vs Macro F1 score, what's the difference?*, *Stephen Allwright*. Available at: https://stephenallwright.com/micro-vs-macro-f1-score/.
12. *Matthews's correlation coefficient: Definition, formula and advantages* (2022) *Voxco*. Available at: https://www.voxco.com/blog/matthewss-correlation-coefficient-definition-formula-and-advantages/.