Protein Superimposition with Machine Learning and Deep Learning

Theodora Gaiceanu
Lund University
Lund, Sweden
Email: th6636ga-s@student.lu.se

Diego Figueroa
Lund University
Lund, Sweden
Email: di4642fi-s@student.lu.se

***Abstract*—Protein superimposition based on surface shape can allow comparison of widely divergent proteins and identify proteins with similar functions. However, state-of-the-art approaches use optimization methods that are computationally expensive, which means that a more efficient method is needed. In this project, Random Forest, Multi-layer Perceptron and Convolutional Neural Networks regressor models have been used in order to study the generation of either three rotation angles to align two shapes, or a similarity score (Zeal score) between the two shapes. The models predicting the three rotation angles obtained a Mean-Squared Error between 0.0308 and 0.0426, and the models outputting the Zeal score obtained a Mean-Squared Error between 0.0016 and 0.0029. Nevertheless, more data is needed, in order to increase the performance in real cases. The developed program will be part of a pipeline for design of new proteins.**

1. Introduction

Proteins are molecules of large dimensions (macromolecules) with a complex structure [1]. These molecules play a significant role in drug development industry ([2], [3]). The drugs based on proteins are used for treating several diseases, such as cancer, autoimmunity/inflammation, exposure to infectious agents, genetic disorders ([3], [4]). In this context, the shape of a protein is very important, as it determines its function ([5], [6]). Moreover, proteins with similar shapes have the property of having similar functions as well [7]. Therefore, proteins superimposition could be used to analyze the relations between proteins and it is a good way to find functionally related proteins [7]. In addition, a shape-based approach for aligning two proteins is a robust method that can work regardless the amino acid sequence similarity. [7]

In order to perform protein superimposition, one needs some shape descriptors. State of the art projects have used Zernike descriptors to represent the shape of a protein ([8], [9]). Unlike Cartesian coordinates, that can be extremely complex for proteins ([10]), Zernike descriptors are low-dimensional representations of the shape of a protein, using 121 features, and they are size, rotation and translation invariant [7]. The Zernike descriptors are based on the Zernike-Canterakis polynomials ([8], [11]). The Zernike-Canterakis polynomials are given by [8]:

$$Z_{nl}^{m}(r,\theta,\gamma) = R_{nl}(r)\Upsilon_{l}^{m}(\theta,\gamma), \tag{1}$$

where n is the order, m is the repetition, (θ, γ) are the polar coordinates, and $\Upsilon_{l}^{m}(\theta,\gamma)$ are the spherical harmonics [12] of the l^{th} degree with $l \leq n$, $m \in [-l, l]$ and $(n-l)$ even and positive. $R_{nl}(r)$ is the radial polynomial with r radius which is defined in order that $Z_{nl}^{m}(x)$ to be orthonormal polynomials when written in Cartesian coordinates [8].

The current state-of-art method for protein alignment using Zernike Descriptors is ZEAL [7]. It is an interactive tool that uses a surrogate optimization method to align proteins with similar shapes. The Zernike descriptors were used to represent the shape of the proteins ([7], [13]). The output of ZEAL is the Zeal score, which is actually the similarity score between two proteins [7]. It is a very efficient and robust method, but it takes approximately 300 seconds for only one sample to compute. Therefore, there is a need for a more computational efficient approach. In this study the authors present a proposal of a method to use machine learning or deep learning models that, given two 3D shapes (Zernike descriptors), generates the rotation to align the two shapes or the similarity score. The model learns from a large dataset of protein pairs to predict the rotation values for a correct overlap, reducing in this way the computational cost (compared to ZEAL).

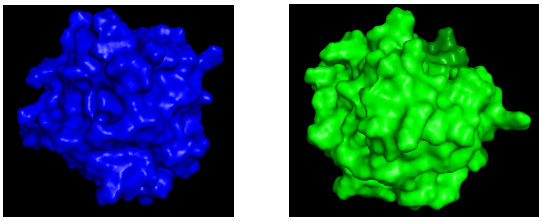
The project was based on this approach and it followed multiple steps: (I) data preparation (described in Section 2), (II) model preparation (detailed in Section 3), (III) model evaluation (discussed in Section 5), (IV) model deployment (discussed in the last part of Section 5). The conclusions of the project work are detailed in Section 6.

2. Dataset

The dataset used in this project was generated by the authors. This was done by processing 209 proteins. An example of a surface view for a protein can be observed in Figure 1a. Figure 1b illustrates the same protein, but rotated. The proteins were described by using the 121 Zernike descriptors. Then, for each protein, a set of rotation angles were considered (from -180 to 180 degrees, with a step

TABLE 1. GENERATED DATASET

Zernike descriptors original protein	Zernike descriptors rotated protein	rotation values	Zeal score
z_o_0 ... z_o_{120}	z_r_0 ... z_r_{120}	ψ, θ, ϕ	zeal score



(a) Original protein (b) Rotated protein

Figure 1. Surface representation of a protein (blue) and its rotated version (green)

of 20, in ψ, θ, and ϕ directions). The protein was rotated according to these values, resulting a new rotated protein. The Zernike descriptors of the rotated protein (relative to the starting position) were computed and stored. An example of the Zernike descriptors of one protein and of its rotated version can be observed in Figure 2. In this case, the rotation was done with -180 degrees in the ψ direction. Afterwards, the Zeal score between the original protein and the rotated version was computed, in order to observe how similar were the two considered orientations.

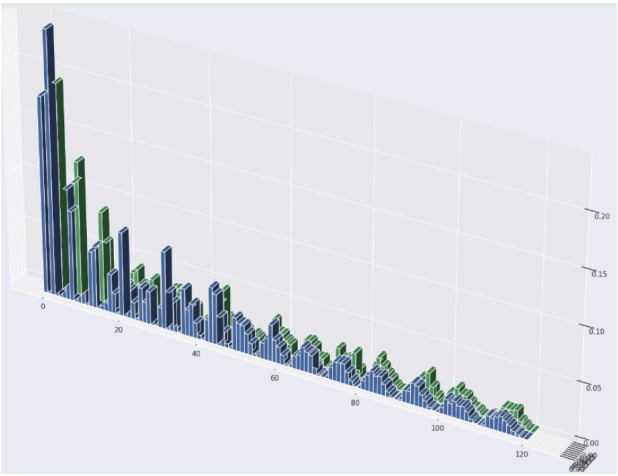


Figure 2. Zernike descriptors of a protein (blue bars) and of the rotated version (green bars) - rotation with -180 degrees in ψ direction

Summarising, the dataset has the following columns: 121 columns with the Zernike descriptors of the original protein, 121 columns with the Zernike descriptors of the rotated protein, the three rotation values (ψ, θ, ϕ directions), and the Zeal score for each original protein and the corresponding rotated version. A schematic representation of the generated dataset can be seen in Table 1. In order to generate this amount of data, a multi-core cluster was used. There were 10 cores computing in parallel. The whole process lasted a few hours.

3. Materials and methods

Two approaches have been tried in order to accomplish the task of the project: (1) feed into the model only the Zernike descriptors of the original and the rotated protein, receive at output the corresponding rotation values in ψ, θ, ϕ directions; (2) feed into the model the Zernike descriptors of the original and the rotated protein and their corresponding rotation angles, receive at output the Zeal score. As it can be seen, in both approaches the output consists of three or one continuous values. Therefore, this problem is a regression problem.

3.1. Random Forest

Random Forest regressor is a very simple classical regressor used in ML problems. It is an example of an ensemble (a model that uses several base models) consisting of multiple decision trees [14]. Some randomness is introduced in each decision tree to reduce the correlation between each tree [14]. The multiple decision trees are fitted on several sub-samples of the dataset. The accuracy is increased by averaging (which is also a way to reduce over-fitting).

As three rotation values should be predicted in the first approach, a multioutput Random Forest regressor was used. In case of the second approach, where only one value should be predicted (the Zeal score), a simple Random Forest regressor was used. Both models have 100 trees (estimators).

The reason for using a Random Forest regressor is that it is a simple Machine Learning method that has been used in research project for creating a baseline model for further analysis [15].

3.2. Multi-layer Perceptron

The multi-layer Perceptron (MLP) is a first step to introduce the Deep Learning approach. MLP is a type of feed-forward artificial neural network that has one input layer, one or several hidden layers, and an output layer [16]. MLP is used in supervised learning and it works in the following way: by training on the input features and labels, it learns a non-linear function that can predict a target [17]. The model used in regression problems is called MLP regressor.

Similarly to the Random Forest model, a multioutput MLP regressor was used for the first approach (with the three rotation values as output), while a simple MLP regressor was used for the second approach (with Zeal score as target). For our task, the best performance was achieved when using a MLP regressor with 43 hidden layers with 100 neurons in each layer. The number of hidden layers was chosen after analyzing the model performance with a

different amount of hidden layers. As it can be observed from Figure 3, the model has the best performance when using 43 hidden layers. The activation function used for the hidden layers is rectified linear unit (ReLU) and the weight optimizer is Adam.

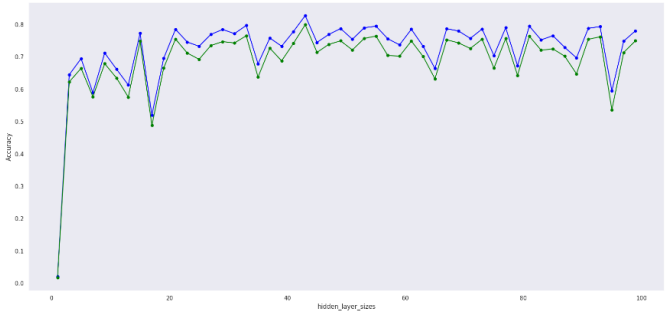


Figure 3. The accuracy of the MLP regressor model using a different amount of hidden layers

The reason for using a MLP regressor is that it is the most simple neural network model. Therefore, its performances can give an insight of whether an ANN approach is suitable or not for this project.

3.3. Convolutional Neural Networks

One of the main advantages of the Convolutional Neural Networks (CNN) over classical neural networks is the reduction of the number of parameters through convolution [18] , [19]. The training process with CNNs has muliple steps: the first one is convolution, in which maps of features are resulted [20]. Then feature learning is done by using a nonlinear activation function, like ReLU. Afterwards, the dimensions of the feature maps are reduced and the spatial invariance is kept through a pooling stage [20], [19]. The last stage is either classification or regression, depending on the usage of the model. This is done by one or more fully-connected layers. In case of a regression problem, the output layer should have a linear activation function.

Once again, a simple CNN regressor with three outputs was used for the case predicting the rotation values, while a CNN regressor with one output was used for the case predicting the Zeal score.

The architecture of the CNN regressor model predicting the rotational values can be analyzed in Table 2. 1D convolutional layers were used because the features consist of a vector (composed only by the Zernike descriptors or by the Zernike descriptors and the three rotational values) . As it can be seen, the model has two 1D convolutional layers. The model has 58019 parameters in total. The architecture of the CNN regressor model predicting the Zeal score can be observed in Table 3. As it can be noticed, the model has three 1D convolutional layers. The model has 32817 parameters in total, less than the previous CNN model. The number of convolutional layers has been chosen in the following way: firstly a model with a single convolutional layer was used, then more convolutional layers were added, observing at the same time the model performance. Finally, the optimal number of convolutional layers with respect to the model performance was chosen.

TABLE 2. SUMMARY OF THE IMPLEMENTED MODEL ARCHITECTURE FOR THE APPROACH WITH ROTATION VALUES AS TARGET

Layer (type)	Output Shape	Parameters
Reshape	(None, 242, 1)	0
1D convolutional	(None, 238, 64)	384
MaxPooling	(None, 79, 64)	0
Dropout	(None, 79, 64)	0
1D convolutional	(None, 77, 32)	6176
MaxPooling	(None, 25, 32)	0
Dropout	(None, 25, 32)	0
Flatten	(None, 800)	0
Dense	(None, 64)	51264
Dropout	(None, 64)	0
Dense	(None, 3)	195

TABLE 3. SUMMARY OF THE IMPLEMENTED MODEL ARCHITECTURE FOR THE APPROACH WITH ZEAL SCORE AS TARGET

Layer (type)	Output Shape	Parameters
Reshape	(None, 245, 1)	0
1D convolutional	(None, 241, 64)	384
MaxPooling	(None, 80, 64)	0
Dropout	(None, 80, 64)	0
1D convolutional	(None, 78, 32)	6176
MaxPooling	(None, 26, 32)	0
Dropout	(None, 26, 32)	0
1D convolutional	(None, 24, 16)	1552
Flatten	(None, 384)	0
Dense	(None, 64)	24640
Dropout	(None, 64)	0
Dense	(None, 1)	65

Dropout layers were used in order to reduce overfitting. Early stopping was used as well. Also, the model utilized a batch size of 16. Other choices for batch sizes were 8 and 32, but the model performed worse. The used optimizer is Adam, the same as in the MLP regressor.

CNNs were used to study if the convolutional layers could be able to find some relations between the Zernike descriptors of the original and the rotated protein.

4. Software

The code was written in Python programming language, and PyCharm was the used integrated development environment. Python has several advantages (it is free and open-source) which offers a robust alternative to MATLAB and its tools. In addition, Python offers specialized modules for Machine Learning and Deep Learning applications, such as Sci-Kit Learn and Keras. Also, there is a specialized Python library, PyRosetta, that is a software tool used by more than 50 research laboratories (in academy and private companies), and it is the state-of-art for building molecular modeling algorithms [21]. The project was developed using a Linux server with 10 cores.

5. Results and discussion

Before starting the training process, the Zernike descriptors for both the original and the rotated proteins, as well as the rotation angles were scaled by using MinMax scaler. The dataset was divided into 80% training and 20% testing for all considered models.

The Mean Squared Error (MSE) for the considered models is illustrated in Table 4. As it can be observed, the models outputting the three rotation values have higher values for MSE. The worst performance is obtained with multioutput Random Forest regressor (0.0426), and the best one is achieved with CNN regressor (0.0308). The multioutput MLP regressor has a quite similar performance (0.0325) to that of the CNN regressor. The evolution of the MSE for this model can be seen in Figure 4. The training for the CNN regressor with 3 outputs takes 13 epochs. The learning process with the multioutput MLP regressor takes 140 iterations.

TABLE 4. MEAN SQUARRED ERROR SCORE FOR THE CONSIDERED MODELS

Method	Output	MSE
Multioutput Random Forest regressor	Rotation values	0.0426
Random Forest regressor	Zeal score	0.0016
Multioutput MLP regressor	Rotation values	0.0325
MLP regressor	Zeal score	0.0023
CNN regressor (3 outputs)	Rotation values	0.0308
CNN regressor (1 output)	Zeal score	0.0029

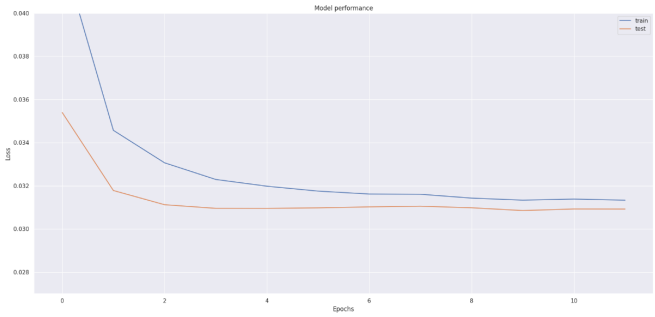


Figure 4. Evolution of the loss function for the CNN regressor with 3 outputs

A second approach was also tried, which used simple regressors to output the Zeal score. In this way, there was more input data to feed into the models: the Zeal descriptors of the original and the rotated proteins, but also the three rotation values. This solution leads to a slightly improvement, as it can be observed in Table 4. The best performance is acquired this time with the Random Forest model (0.0016). The MLP regressor and the CNN regressor have similar performances.

Figure 5 illustrates the evolution of the loss function for the MLP regressor. The learning process takes 140 iterations, exactly the same number of iterations as in the multioutput case. Figure 6 shows the evolution of the loss function for the CNN regressor. The learning process takes 23 epochs (which is more than the duration of the training process with the CNN regressor outputting the rotation values).

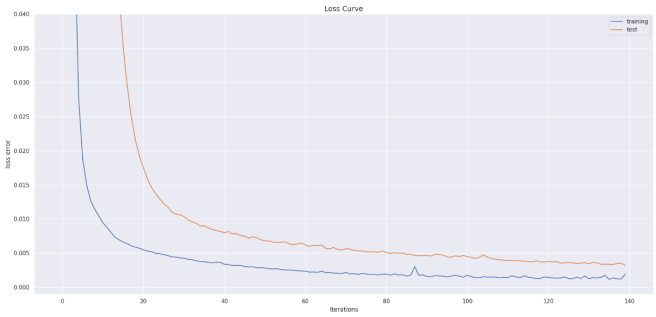


Figure 5. Evolution of the loss function for the MLP regressor with 1 output

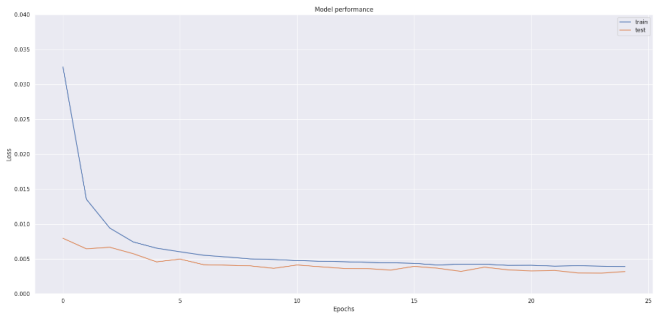


Figure 6. Evolution of the loss function for the CNN regressor with 1 output

Overall, the MLP and the CNN regressor models converged faster than the Random Forest regressor. In addition, the Random Forest model with only one output took less time to train than the model with the three outputs.

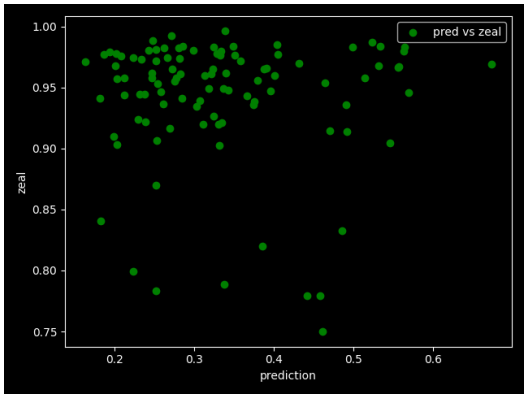


Figure 7. Performance of the models predicting the Zeal score with real data

The models predicting the Zeal score were tested on a real case. The real case is represented by a ”L”-shape protein that is not similar to any shape found in the protein

data bank. The rotation values and the Zeal score are known for this "L"-shape protein. Figure 7 presents the results of the models in this real case. The X-axis represents the prediction (Zeal score) given by the models and the Y-axis represents the true Zeal score. As it can be noticed, the points are not correlated. Normally, a higher predicted score should correspond to a higher Zeal score, but this is not the case. The models predicting the rotation values had a similar behaviour. A reason for this is that more data samples are needed in order to generalize and succeed at such a challenging benchmark.

6. Conclusion

In this project, a method to align two proteins was developed using Random Forest, MLP, and CNN regressors. The data was generated by the authors and it consisted of the Zernike descriptors of the original proteins, the Zernike descriptors of the rotated proteins, the rotation values and the Zeal score between the original, and the rotated proteins. A Python script was developed to further generation of more sample data using a multicore cluster.

Two approaches were tried: in the first, only the Zernike descriptors of the original and the rotated proteins were fed into the models and the output consisted of the rotational values in the ψ, θ, and ϕ directions; in the second approach, the rotational values were also added in the input and the models predicted the Zeal score. The models from the first approach performed well, but none of them managed to get a MSE value of 0.01. The best multioutput regressor model was the one using CNN (MSE - 0.0308). With the second approach, the models performed better, the Random Forest regressor having the best result (MSE - 0.0016). However, when tested in a real case scenario, even these models failed to obtain good results. These results suggest that more data is needed. Moreover, for the approach predicting the rotational values, a possible improvement would be to use 2D convolutional layers, having as input two features vectors (one with the Zernike descriptors of the original proteins and the other with the Zernike descriptors of the rotated proteins). In this way, 2D convolutional layers could make a better use of the Zernike descriptors properties and may find better correlations.

In addition, the data preparation step revealed that data generation is a tedious and computational expensive process.

In the future, the project could be used to compare two proteins given the correct alignment, or it could be extended to more problems like protein design where it is also needed to align proteins to a desired shape.

Acknowledgments

The authors would like to thank to their supervisor, Daniel Varela, postdoctoral scholar at Andrelab, a laboratory inside the Biochemistry and Structural Biology Department of Lund University.

References

[1] A. Blanco and G. Blanco, "Chapter 3 - proteins," in *Medical Biochemistry*, A. Blanco and G. Blanco, Eds. Academic Press, 2017, pp. 21–71. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780128035504000033

[2] J. Ha, H. Park, J. Park, and S. B. Park, "Recent advances in identifying protein targets in drug discovery," *Cell Chemical Biology*, vol. 28, no. 3, pp. 394–423, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2451945620304761

[3] H. Lagassé, A. Alexaki, V. Simhadri, N. Katagiri, W. Jankowski, Z. Sauna, and C. Kimchi-Sarfaty, "Recent advances in (therapeutic protein) drug development," *F1000Research*, vol. 6, no. 113, 2017.

[4] C. Krejsa, M. Rogge, and W. Sadee, "Protein therapeutics: new applications for pharmacogenetics," *Nat Rev Drug Discov*, vol. 5, p. 507–521, 2006.

[5] B. Alberts, A. Johnson, and J. Lewis, *Molecular Biology of the Cell, 4th edition*. New York: Garland Science, 2002. [Online]. Available: https://www.ncbi.nlm.nih.gov/books/NBK26830/

[6] A. Gramada and P. Bourne, "Multipolar representation of protein structure," *BMC Bioinformatics*, vol. 7, May 2006.

[7] F. Ljung and A. Ingemar, "ZEAL: protein structure alignment based on shape similarity," *Bioinformatics*, vol. 37, no. 18, pp. 2874–2881, 03 2021. [Online]. Available: https://doi.org/10.1093/bioinformatics/btab205

[8] V. Venkatraman, L. Sael, and D. Kihara, "Potential for protein surface shape analysis using spherical harmonics and 3D Zernike descriptors," *Cell biochemistry and biophysics*, vol. 54, pp. 23–32, 02 2009.

[9] S. Daberdaku and C. Ferrari, "Exploring the potential of 3D Zernike descriptors and SVM for protein–protein interface prediction," *BMC Bioinformatics*, vol. 19, no. 35, 2018.

[10] M. Bayati, M. Leeser, and J. Bardhan, "High-performance transformation of protein structure representation from internal to cartesian coordinates," *Journal of Computational Chemistry*, vol. 41, 07 2020.

[11] N. Canterakis, "3D Zernike moments and Zernike affine invariants for 3D image analysis and recognition," *Proc 11th Scandinavian Conference on Image Analysis*, pp. 85–93, 1999.

[12] H. Dym and H. McKean, *Fourier series and integrals*. Academic Press, 1972.

[13] (2022, May) Zealweb. [Online]. Available: http://zeal.andrelab.org/webapps/home/

[14] A. Lindholm, N. Wahlström, F. Lindsten, and T. B. Schön, *Machine Learning - A First Course for Engineers and Scientists*. Cambridge University Press, 2022. [Online]. Available: http://smlbook.org/

[15] G. Rodríguez, M. J., V. Montequín, A. Aranguren Ubierna, R. Santana, B. Sierra, and A. Zelaia, "Award price estimator for public procurement auctions using machine learning algorithms: Case study with tenders from spain," *Studies in Informatics and Control*, vol. 30, pp. 67–76, 12 2021.

[16] T. Sunthornnapha, "Utilization of MLP and linear regression methods to build a reliable energy baseline for self-benchmarking evaluation," *Energy Procedia*, vol. 141, pp. 189–193, 2017, power and Energy Systems Engineering. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1876610217354425

[17] (2022, May) Neural network models (supervised). [Online]. Available: https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[18] A. Ghosh, A. Sufian, F. Sultana, A. Chakrabarti, and D. Debashis, *Fundamental Concepts of Convolutional Neural Network*, 01 2020, pp. 519–567.

[19] V. Passricha and R. K. Aggarwal, "Convolutional neural networks for raw speech recognition," in *From Natural to Artificial Intelligence*, R. Lopez-Ruiz, Ed. Rijeka: IntechOpen, 2018, ch. 2. [Online]. Available: https://doi.org/10.5772/intechopen.80026

[20] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, 2016.

[21] (2022, May) Pyrosetta. [Online]. Available: https://www.pyrosetta.org/