Department: DEIB, Politecnico di Milano

Machine Learning for drug-target interaction estimation in Drug Discovery: from replication to data integration analysis

Vittorio Pio Remigio Cozzoli
Student, Politecnico di Milano

Piervito Creanza
Student, Politecnico di Milano

Davide Gadioli
Assistant Professor, Politecnico di Milano

Gianmarco Accordi
PhD Student, Politecnico di Milano

Gianluca Palermo
Full Professor, Politecnico di Milano

***Abstract*— This project focuses on the analysis and the retraining of a convolutional neural network, GenScore, designed to calculate the affinity degree (score) between a protein and a ligand. The main objective is to adapt this network to work with a new set of data produced by Politecnico di Milano. This new data has a distinctive feature: the pockets of the proteins have a smaller cutout compared to what was originally planned for the network. On one hand, this modification makes the data more manageable and less computationally burdensome; on the other hand, it becomes necessary to investigate how this model reacts to a stimulus so different from what was originally conceived for it. This requires a very patient and particularly meticulous examination of the new results produced, in order to understand their accuracy in terms of precision and efficiency. Through this retraining, we therefore hope to improve the ability of the network to work with different sizes of cutouts of the protein pockets, thus increasing its versatility and applicability in various scientific contexts. We hope that the results of this project, in our small contribution, may have significant implications for future research in the field of bioinformatics and computational chemistry.**

Introduction

"Machine Learning for drug-target interaction estimation in Drug Discovery: from replication to data integration analysis" was born from the need to have a deeper and more coherent understanding of models based on neural networks that are adopted in the field of bioinformatics and computational medical chemistry, so that it could be possible to improve these models when facing new different datasets.

Specifically, the project consists in understanding how these models learn and how flexible they are in relation to perturbations regarding modifications of the structure of the data which they were designed to interact with. In collaboration with Professor Gianluca Palermo and the PhD students Davide Gadioli and Gianmarco Accordi, the project was carried forward by two third-year Computer Engineering students at Politecnico di Milano: Vittorio Pio Remigio Cozzoli and Piervito Creanza.
For a total duration of about three and a half months, the project has focused on two main objectives:

- **Replicating the GenScore paper results:** despite the model being open source, some of its scripts were not general enough to be used on different machines and datasets. This constitutes a known problem in the world of research. Not by chance, in fact, there are several articles in the literature where attempts have been made to replicate on other data what has been done with such models.
- **Adapting the model to a new set of data:** after coming to grips with the first goal, we started tackling a second model issue, namely its high sensitivity to data. GenScore works very well if the data is processed 'ad hoc', but it is not robust to variations. Our ambition was to enhance the model's ability to adapt to inputs different from those originally intended. This could lead to the use of new data sources, distinct from traditional experimental ones, creating new opportunities and enabling novel research opportunities. A glimpse of these new opportunities is provided in the *SECOND TRAINING PHASE* section.

RELATED WORK

Before starting with the actual project, it was useful as well as necessary to understand why this model was created and how. In particular, it was important to understand the needs in the field of bioresearch that led to the development of neural networks built ad hoc with the intent to calculate the degree of affinity between a protein and a ligand.

In this regard, it was essential to review two scientific papers (which can be found in the *REFERENCES* section). The former associated with RTMScore, the predecessor of GenScore, and the latter associated to GenScore itself. This review had the further intent of investigating the two different approaches adopted by them.

In the present case, the first paper discusses the development of a novel scoring function, RTMScore, for protein-ligand interactions. This function was developed using machine learning approaches and a unique tailored residue-based graph representation strategy. It also uses several graph transformer layers for learning protein and ligand representations, and a mixture density network to obtain residue-atom distance likelihood potential. The RTMScore was validated on the CASF-2016 benchmark and outperformed most state-of-the-art methods in both docking and screening powers. It showed robust performance, maintaining its docking power on cross-docked poses and improving performance as a rescoring tool in larger-scale virtual screening.

However, achieving robust performance and wide applicability of scoring functions remains a challenge.

Hereinafter, this is when, a few years later, the same research group gave birth to GenScore, an evolution of RTMScore, which has gained attention in the context of the application of machine learning algorithms to protein-ligand scoring functions due to its high predictive accuracy and affordable computational cost. In fact, a known issue of these models is the fact that most of these scoring functions are task-specific, making the development of a balanced scoring function a truly demanding challenge. In order to address this, the authors of the paper propose a new parameterization strategy that introduces an adjustable binding affinity term into the training of a mixture density network. This strategy improves the scoring and ranking performance while maintaining superior docking and screening power. The authors also explore the impacts of key elements on prediction accuracy and task preference, demonstrating that a model's performance can be balanced through an appropriate approach. The study underscores the potential utility of this innovative parameterization strategy and the resulting scoring framework in future structure-based drug design.

INNOVATION OF OUR PROJECT

In the *RELATED WORK* section, we meticulously reviewed the origins and motivations behind the creation of RTMScore and GenScore. It is equally important to highlight and outline why our project also needed to come to life.

However, before understanding the reasons that led us to start this project, it is important and necessary to understand what a protein and a ligand are, respectively, and how these two biological entities interact with each other. This understanding provides a concrete view of the differences that distinguish the two original models from our retrained model.

What are proteins and ligands?

Proteins or protids (from the Greek *protos*, "primary") represent a large group of organic compounds formed by sequences of amino acids linked together through peptide bonds. We can imagine amino acids as the bricks for building proteins and peptide bonds as the glue that holds them together.

The main characteristic of proteins that allows them to have a diversified set of functions is the ability to bind other molecules in a specific way. The region of the protein that allows binding with the other molecule is known as the ***binding site*** and is often a depression or ***"pocket"*** on the molecular surface. This binding ability is mediated by the tertiary structure of the protein [Figure 1], which defines the pocket of the binding site, and by the chemical properties of the side chains of the surrounding amino acids.

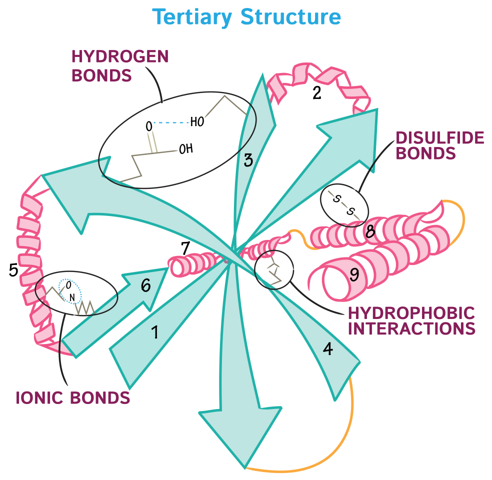

Figure 1. Tertiary structure of a protein with its bonds.

A ligand (from Latin *ligare*, "to bind"), on the other hand, is a molecule capable of binding a biomolecule and forming a complex capable of performing or inducing a biological function. Strictly speaking, it is usually an effector molecule capable of binding to a target protein through a weak interaction such as an ionic bond, a hydrogen bond, or a Van der Waals interaction. The target biomolecule is most often a receptor.

The conformation of a protein, although it is normally as stable as possible for the sequence of its amino acids, is not immutable, and therefore undergoes small modifications in terms of spatial conformation (alteration of its three-dimensional structure) due to interaction with ligands or other proteins. This characteristic is at the base of the functionality of most proteins, as the conformational variation is reflected in a new functional state (the function of a macromolecule is indeed closely related to its structure). The strength of the bond is called ***"affinity"*** and represents the core of this whole project.

Almost all known proteins interact with other proteins or with other types of molecules, always defined as ligands, through their binding sites; this is at the base of most of the interactions present in a cell, where the association of the ligand with the target biomolecule is called *docking*.

The need of our project

The aforementioned excursus thus represents the theoretical link between the original models and our retrained model. In fact, the crux of this project concerns the way in which the neural network intrinsically work and manage the data of the proteins and ligands with which it must interact. In particular, GenScore was born to work with a specific cutout for the pockets of the various proteins. Remembering that the *"pocket"* of a protein also takes the name of *"binding site"*, it is necessary to take into account the fact that the ligand of an enzyme, which takes the name of substrate, is generally much smaller than the enzyme itself. Although enzymes can be made up of hundreds of amino acids, usually only a small fraction of the residues come into contact with the substrate, and on average only one in four is directly involved in catalysis. The region of the enzyme that binds the substrate and contains the catalytic residues is known as the ***active site***, which is therefore generally smaller than the entire binding site of the enzyme (protein). This is where the experimental need to try a new approach arises: ours.

The innovative idea of our project consists primarily in investigating how this model reacts to a completely different stimulus in reference to the input data that the neural network is expecting. This analysis aims to understand in terms of performance how much the model is already capable of interacting and managing a completely different dataset from the one originally designed for it. As already mentioned in the *ABSTRACT*, the new data produced shows a slimmer protein cutout, making the input data more manageable and less burdensome in computational terms. Therefore, the objective of this project is precisely to make GenScore capable of working with both types of protein cutouts, thus increasing its versatility and applicability in different scientific contexts.

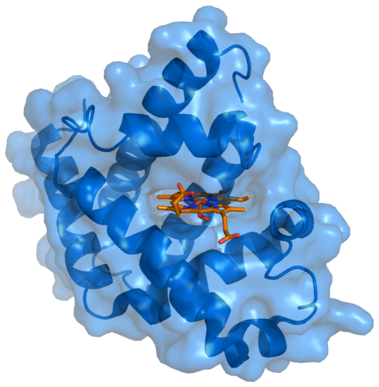

Figure 2. Docking of Myoglobin (blue) with its ligand heme (orange) bound. Based on PDB: 1MBO

PROJECT DEVELOPMENT

Understanding how a neural network works

Before starting the project, time was dedicated to the preliminary phase, during which we first followed a brief introductory course on neural networks and the fundamental concepts related to them. This course provided us with a solid foundation for understanding the basic principles of neural networks and introduced us to the necessary tools for development, such as the `PyTorch` library.

Significant resources were invested in this initial phase to ensure a common understanding of the concepts and tools necessary for the development of the project.

Familiarization with the GenScore code

Once the basic course was completed, we analyzed the source code of the project, available on the related GitHub repo.

The project is characterized by a `model` that deals with the training and the generation of the trained model itself, as well as a series of `scripts` necessary for the `ETL` (Extraction Transformation and Loading) process of the data and the validation of the model too.

Our work focused mainly on these last two phases.

Requirements identification

One of the first obstacles was the need to identify the Python requirements of the project.

Despite the availability of a list of necessary dependencies in the repo, many of the packages within that list were not valid and obsolete or even specific to a particular operating system. Consequently, it was necessary to manually cross-reference the dependencies in order to find substitute packages compatible with our hardware and software configuration.

Configuration of the virtual machine and installation of requirements

After identifying the correct requirements, we proceeded to configure the virtual machine assigned to us by our project manager.

Indeed, the training of a neural network requires large computational resources, particularly concerning hardware acceleration through GPU. The system that was assigned to us guaranteed an NVidia A100 graphic card with 40GB, so that we could carry out the retraining in a reasonable time.

Working on a remote system made it necessary to configure the development environment with the installation of the previously identified requirements and the setting of the remote execution of the code. Therefore, we had to configure an SSH access profile for both the members of the team, in order to gain remote access to the VM.

In addition to this, we chose to use the PyCharm IDE and its remote execution over SSH, in order to be able to keep a local copy of the code and run it on the virtual machine when necessary.

Reimplementation of the conversion of molecules into graphs

Initially, it was necessary to reverse engineer the `mol2graph_rdmda_res.py` script, crucial for the operation of the model, as the version present on the GitHub repo was not fully compatible with the format and structure of our proteins. Moreover, the code had been designed specifically for the machine on which it was running, making it impossible to run on another system.

The reverse engineering phase made it necessary to analyze the operation of the model more deeply in order to understand the structure that the data should have in order to be used in the training process. The total absence of code-side documentation made this phase extremely long, complex, and particularly laborious.

In particular, the code referred to a file named `pdbbind_2020_general.csv`. However, this document was not available either in the related scientific paper of the project or in the repo published on GitHub. Through a careful analysis of the code, several failed attempts, and the support of the project managers, we finally managed to identify the source from which such data had been obtained and we took care of generating the missing file, necessary for the correct operation of the script.

Listing 1. Arguments inserted from CLI to make the code more general.

```
p.add_argument('-d', '--dir', default=
    ".", help='The directory of the
    refined set')
p.add_argument('-dg', '--
    general_set_dir', default=".",
    help='The directory of the general
     set')
p.add_argument('-c', '--cutoff', type=
    float, help='the cutoff to
    determine the pocket')
p.add_argument('-o', '--outprefix',
    default="out", help='The output
    bin file.')
p.add_argument('-r', '--ref', default=
    "/home/shenchao/pdbbind/
    pdbbind_2020_general.csv", help='
    The reference file to query the
    label of the complex.')
p.add_argument('-usH', '--useH',
    default=False, action="store_true"
    , help='whether to use the
    explicit H atoms.')
p.add_argument('-uschi', '--
    use_chirality', default=False,
    action="store_true", help='whether
     to use chirality.')
p.add_argument('-p', '--parallel',
    default=False, action="store_true"
    , help='whether to obtain the
    graphs in parallel')
```

The exhaustive phase of reverse engineering finally allowed the identification of the necessary modifications to be implemented so that the aforementioned script could be compatible with our new dataset.

Once this phase was completed, we implemented an improved conversion script, more general and therefore compatible with different types of systems and data. This made it necessary to resolve some bugs present in the original code and introduce new parameters and functions.

Listing 2. Reference to the missing file pdbbind_2020_general.csv.

```
  p.add_argument('-r', '--ref',
      default="/home/shenchao/pdbbind/
      pdbbind_2020_general.csv", help=
      'The reference file to query the
       label of the complex.')
# [...]
def label_query(pdbid, df):
    logKd = df.loc[pdbid, "-logKd/Ki"]
    return logKd
```

Thanks to the new script, it was possible to convert the cut molecules of the `refined-set` into graphs and proceeded to save them as `NumPy` binaries.

Listing 3. Portion of code dedicated to data export.

```
np.save("%s\_ids" % args.outprefix, (
    ids, labels))
th.save(graphs\_p, "%s\_prot.pt" %
    args.outprefix)
th.save(graphs\_l, "%s\_lig.pt" % args
    .outprefix)
```

First training of the model

After generating the graphs, we finally moved on to the actual retraining of the neural network. This procedure also required a lot of time dedicated to reverse-engineering the code, requiring several iterations on the `mol2graph_rdmda_res.py` script mentioned earlier.

It is important to underline that convolutional neural models are based on specific principles related to linear algebra, specifically particular matrix products.

For this reason, a particularly significant problem we encountered was precisely the multiplication of two matrices that turned out to be non-conformable.

This was due to a bug in the original graph conversion script, which failed in setting some default parameters. The absence of these parameters therefore caused the failure to generate some bits concerning the chirality of the protein, generating a smaller matrix than expected. Therefore, it was necessary to refactor the code again to solve this problem.

After the long debugging phase, it was possible to start an initial phase of training of the neural network. The process required about 150 epochs, where in the last seventy the best validation value remained constant, causing the early stopping termination, due to reaching a *local minimum* [Figure 3].

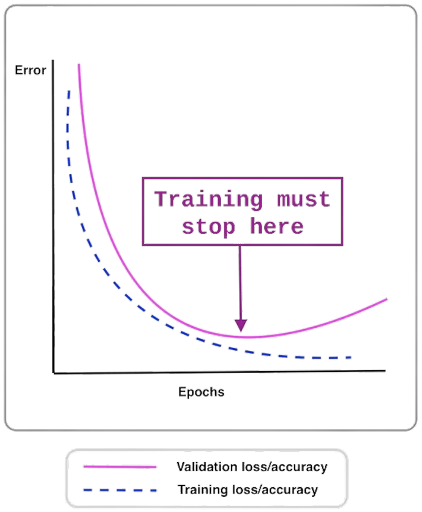

Figure 3. Early Stopping function.

Second training phase

After finishing the first training, we focused on a second training phase. The aim of this stage was to compare how the model reacted to two different training labels:

- ***-logKd/Ki*** based labels.
- ***energy*** based labels.

The *energy* label of a complex is a measure of the strength of the interaction between the two molecules, in our case, a protein and a ligand. This energy is usually calculated using various computational models and simulations. In our case, the energy data at our disposal was generated using about 1 million GPU hours on various european super computers. Ideally, it could have been possible to go further and continue to generate data; however, it was not possible for our project, as this would have taken several months, amount of time which we unfortunately had not.

On the other hand, *"-logKd/Ki"* refers to the negative logarithm of the dissociation constant (Kd) over the inhibition constant (Ki) of the protein-ligand complex. Kd and Ki are experimental measures of the affinity between a ligand and a protein. The lower Kd/Ki, the higher the affinity, meaning the ligand binds more tightly to the protein. Taking the negative logarithm of Kd over Ki converts these values to a more manageable scale and reverses the direction of the scale so that a larger value indicates a stronger interaction.

While both *"energy"* and *"-logKd/Ki"* labels provide information about the strength of the interaction between a protein and a ligand, they are derived in different ways and can sometimes provide different perspectives. Using an "energetic" approach provides at least two different advantages over classical approaches:

- **More standardised results.** Computer simulations often yield more standardized results compared to experimental lab data, as they can be precisely controlled and replicated, reducing the variability of outcomes.
- **Easiness of massive data production.** Thanks to computational simulations it is theoretically possible to generate an *infinite* amount of information. This can be done also for those protein-ligand combinations where it is not possible to carry out *in-vitro* experiments.
- **Cheapness of the technique.** In contrast to laboratory experiments, computer simulations require fewer resources, such as equipment, materials, and physical space, making them more cost-effective and logistically convenient.

Because of these advantages, it was interesting to compare how the neural network would perform using *energy* labels, as the original model was instead conceived to be trained on *-logKd/Ki* labels.

This second training phase was performed on 2322 protein-ligand pairs from PDBbind database.

In order to compare the two techniques, it was necessary to carry out two more trainings: one using the new *energetical approach* and the other one using the classical *-logKd/Ki* labels.

The two trainings required about 550 epochs each, always stopping due to reaching the early stopping condition.

Model validation: the metrics of CASF-2016

With the end of the training phase and the generation of a trained model, we have moved on to the next phase, which is the evaluation of the model. This phase consists of using a particular benchmarking tool that requires the calculation of various metrics, the CASF-2016 Benchmark.

The CASF-2016 Benchmark, also known as the Comparative Assessment of Scoring Functions 2016, is a tool used for evaluating the performance of scoring functions in structure-based drug design. It provides an objective and standardized platform for assessing various scoring functions.

The benchmark consists of four key metrics:

1) **Power Scoring:** this measures the ability of a scoring function to accurately predict the binding affinity of a protein-ligand complex.
2) **Ranking Power:** this evaluates the ability of a scoring function to correctly rank different protein-ligand complexes based on their binding affinities.
3) **Docking Power:** this assesses the ability of a scoring function to correctly rank the generated docking pose.
4) **Screening Power:** this tests the ability of a scoring function to correctly identify active compounds from a large database. Within the ***Screening Power*** metric, we also distinguish:
   - **Forward Screening Power:** its goal is to identify the true binders for a given target.
   - **Reverse Screening Power:** its goal is to identify a potential target for a given active compound.

The CASF-2016 Benchmark includes a test set of 285 protein-ligand complexes with high-quality crystal structures and reliable binding constant. The results obtained from this benchmark can provide valuable guidance for end users to make smart choices among available scoring functions. Moreover, it is an open-access benchmark, allowing other researchers to utilize it to test a wider range of scoring functions.

Model validation: application of metrics and results

Our work has primarily focused on the validation of the model using the metrics of Forward Screening Power and Power Scoring, as these were the metrics we were asked to analyze, as also anticipated by the original scripts of the model.

Because of this, it was necessary to make changes to the original scripts that concerned the evaluation of these metrics. In particular:

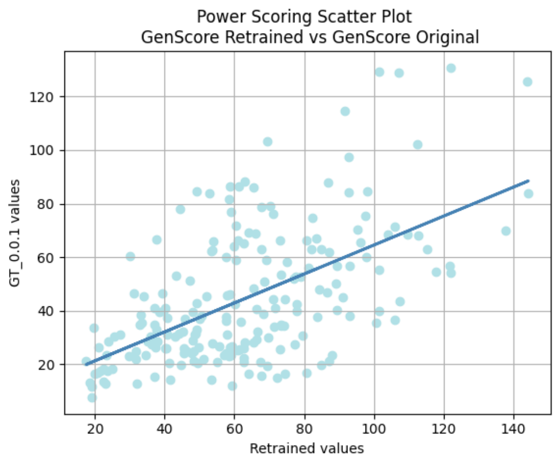

Figure 4. Original vs Retrained models power scoring performances.

- **Forward Screening Power:** starting from the script `casf2016_screening2.py`, we adapted the code into a new script, `casf2016_ForwardScreeningPower.py`, introducing the input of parameters from the terminal, instead of having them hard-coded within the code. In addition to this, we took care of improving the logic of writing results to the filesystem, creating files with data gradually, instead of all at the end. This allows not having to regenerate all the results in case of abnormal program termination. Another area of intervention was the management of parameters passed to functions, now more readable and effective.
- **Ranking Power & Scoring Power:** for this test, a script is used that generates a score for each protein-ligand pair. These data is then used by the CASF to create a report on the predictive abilities of the model. The original script `casf2016_scoring_ranking2.py` has been updated with the input of parameters from the terminal and improved path management, now taking the name of `casf2016_ScoringPower_RankingPower`.

The results obtained from the validation of the model using the CASF metrics will be illustrated in the respective RESULTS section of the document.

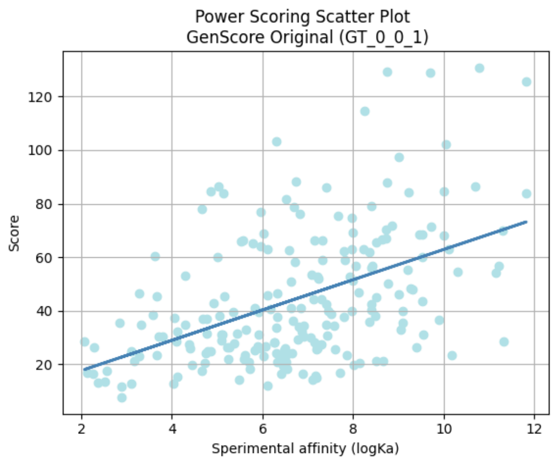

Figure 5. Original model vs experimental affinity.

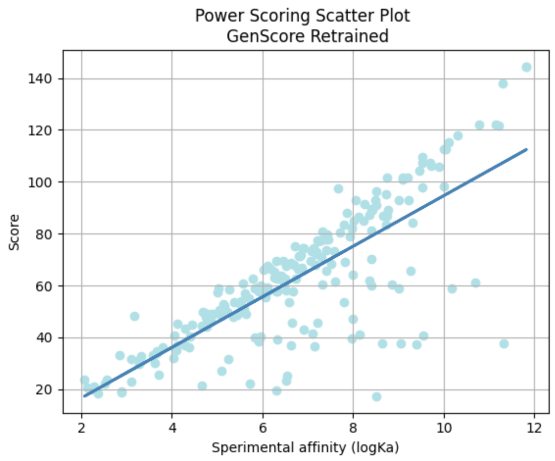

Figure 6. Retrained model vs experimental affinity.

USED TOOLS

The realization of the project required the use of a multitude of tools. In particular:

- **PyCharm:** for writing the code and its remote execution, we chose to use the PyCharm IDE, as extensively described in the previous section.
- **Terminal:** to interact with the VM and execute remote commands, SSH was used along with the respective terminal apps, including for example iTerm2 and Termius.
- **Termius:** to graphically execute SFTP operations on the files present on the VM, making their management easier. In addition to this, Termius provides autocomplete functions for terminal sessions.
- **GitHub:** to version the project and backup its source code (*repo link*).
- **Google Colab:** in order to try ad hoc scripts in an extemporaneous manner, such as those used for generating the scatter plots for the validation of the retrained model shown in the RESULTS section.

RESULTS

The final part of the project involved the analysis of the results obtained from the validation of the retrained model through the benchmarking provided by the CASF-2016 metrics.

For this reason, comparison charts were created in order to juxtapose the different results in the *Power Scoring* test of the retrained model with:

- **Original Model:** it was considered useful to compare, through a scatter plot, the Scoring Power values obtained from the two respective models. This with the aim of verifying whether or not there was a positive correlation between their results.
- **Experimental Affinity:** in a very similar way to the previous one, the result obtained from the model was compared with the actual values measured in the laboratory, specifically the value of *binding affinity* defined as $\log K_a$.

First training results

As shown in the scatter plots represented in [Figures 4, 5, 6], the retrained model behaves in a truly positive manner in terms of *power scoring* in spite of the new cutout dataset used. In particular, the table in [Figure 11] gathers the numeric results obtained by the *Forward Screening Power* metric.

Second training results

In terms of *Power Scoring*, as shown in [Figures 7, 8], the new energetical approach didn't match our positive expectations. In fact, in contrast to the *-logKd/Ki* one, the new training method shows a very weak negative correlation and a very high value dispersion. This can be attributed to numerous factors, such as:

- **Size of the training set:** since an energetical label wasn't available for each of the PDBind complexes, only a set of about 2K protein-ligand pairs was used. This is a relatively small number, if compared either to the original amount of protein-ligand pairs used to train the original model or the number of parameters of the neural network itself.
- **Quality of the energy labels:** the used energy labels had already shown a fairly low positive

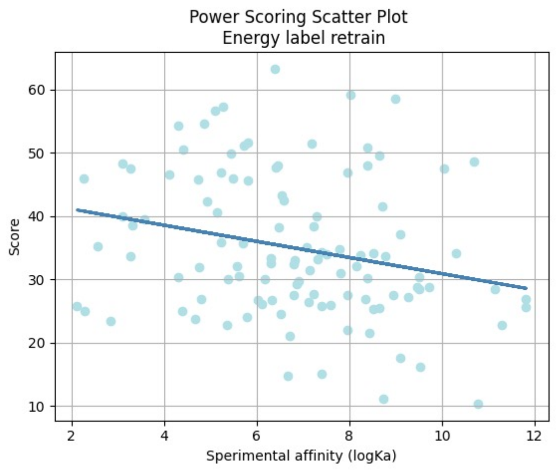

Figure 7. Second retrain using energy labels.

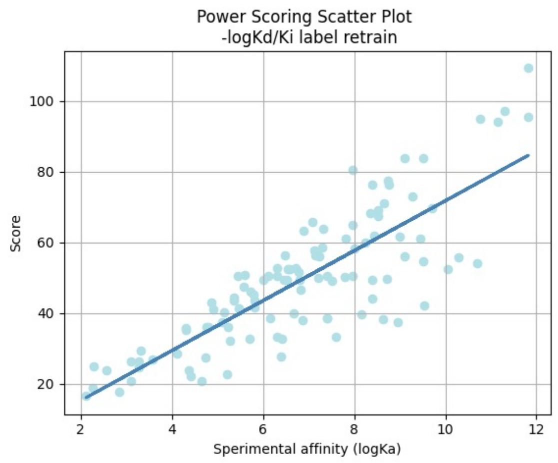

Figure 8. Second retrain using -logKd/Ki labels.

correlation for massive quantity of data in other experiments, thus negatively affecting our results.

- **Model inadequacy:** since the model was conceived for *-logKd/Ki* labels, using a different set could have compromised its results.

Although the two approaches show an opposite correlation in terms of *power scoring*, when accounted for *forward screening power* performances, both the two new trainings show very similar but truly poor results if compared to the previous one. This shows the importance of using an adequately large and consistent set of data in order to successfully train the model as already highlighted in the power scoring analysis.

It would be interesting to perform a new training when a larger set of *energy labels* will be available.

More specific results can be found in [Figures 13 12, 9 and 10].

CONCLUSIONS

This project wants to propose two new approaches to the training of state-of-the-art convolutional neural networks conceived to calculate the degree of affinity between various protein-ligand complexes: on one hand, using datasets of cutout protein pockets, while, on the other hand, using energy labels rather than classical ones.

As can be seen in the RESULTS section, the first of these new approaches led us to achieve very good scoring results, even though the GenScore model was not developed to work with this type of training set.

However, our second experiment didn't return the kind of results we hoped for.

It would be interesting to see if these same results and approaches, for better or worse, could be replicated with other similar models, such as RTMScore, the predecessor of GenScore.

This could be the goal of further research works.

ACKNOWLEDGMENTS

We thank Professor Gianluca Palermo, Davide Gadioli and Gianmarco Accordi for their support and dedication in helping us throughout the whole project.

We want to dedicate a special thank also to Politecnico di Milano, which allowed us to take part in this project and gave us the opportunity to see the world of academic research much more closely.

REFERENCES

1. Chao Shen, Xujun Zhang, Yafeng Deng, Junbo Gao, Dong Wang, Lei Xu, Peichen Pan*, Tingjun Hou*, and Yu Kang*, "**Boosting Protein–Ligand Binding Pose Prediction and Virtual Screening Based on Residue–Atom Distance Likelihood Potential and Graph Transformer**", *J. Med. Chem. 2022, 65, 15, 10691–10706*, doi: 10.1021/acs.jmedchem.2c00991
2. Chao Shen, Xujun Zhang, Chang-Yu Hsieh, Yafeng Deng, Dong Wang, Lei Xu, Jian Wu, Dan Li, Yu Kang, Tingjun Hou, and Peichen Pan, "**A generalized protein–ligand scoring framework with balanced scoring, docking, ranking and screening powers**", *Chem Sci. 2023 Aug 2; 14(30): 8129–8146*, doi: 10.1039/d3sc02044d

Second Training's PS (energy)	
Analysed metric	Value
The regression equation: logKa =	-0.05 + 8.49 * Score
Number of favorable sample (N)	109
Pearson correlation coefficient (R)	- 0.248
Standard deviation in fitting (SD)	2.12

Figure 9. Second Training's power scoring results with energy labels

Second Training's PS (-logKd/Ki)	
Analysed metric	Value
The regression equation: logKa =	0.10 + 2.10 * Score
Number of favorable sample (N)	109
Pearson correlation coefficient (R)	0.822
Standard deviation in fitting (SD)	1.25

Figure 10. Second Training's power scoring results with -logKd/Ki labels

First Training's FSP (-logKd/Ki)	
Analysed metric	Value
Average enrichment factor among top 1%	22.44
Average enrichment factor among top 5%	7.64
Average enrichment factor among top 10%	4.33
The best ligand is found among top 1% candidates for **34** cluster(s) with a success rate of	60.7%
The best ligand is found among top 5% candidates for **39** cluster(s) with a success rate of	69.6%
The best ligand is found among top 10% candidates for **41** cluster(s) with a succes rate of	73.2%

Figure 11. First Training's Forward Screening Power results

Second Training's FSP (-logKd/Ki)	
Analysed metric	Value
Average enrichment factor among top 1%	1.43
Average enrichment factor among top 5%	1.27
Average enrichment factor among top 10%	1.16
The best ligand is found among top 1% candidates for **3** cluster(s) with a success rate of	5.4%
The best ligand is found among top 5% candidates for **7** cluster(s) with a success rate of	12.5%
The best ligand is found among top 10% candidates for **14** cluster(s) with a succes rate of	25.0%

Figure 12. Second Training's forward screening power results with -logKd/Ki labels

Second Training's FSP (energy)	
Analysed metric	Value
Average enrichment factor among top 1%	1.01
Average enrichment factor among top 5%	1.21
Average enrichment factor among top 10%	1.19
The best ligand is found among top 1% candidates for **2** cluster(s) with a success rate of	3.6%
The best ligand is found among top 5% candidates for **6** cluster(s) with a success rate of	10.7%
The best ligand is found among top 10% candidates for **13** cluster(s) with a succes rate of	23.2%

Figure 13. Second Training's forward screening power results with energy labels