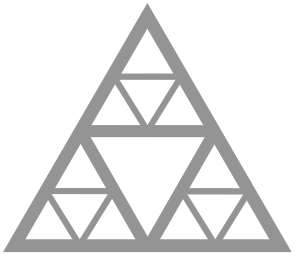


École des Ponts

ParisTech

ÉCOLE NATIONALE DES PONTS ET CHAUSSÉES

Drug Design

Semester project report IMI

Gaspard Beaudouin
Maxime Muhlethaler
Titouan Pottier

Under the supervision of Paraskevi Gkeka, Gabriel Stoltz, Tony Levièvre and Régis Santet
Sanofi
&
CERMICS, Ecole des Ponts

Table des matières

1 Context and Motivation

In this section, we provide context and motivation for our semester project. First, we highlight the pivotal role of Artificial Intelligence in drug discovery, particularly focusing on molecular docking—a critical computational technique.Then, through a case study involving the BCR-ABL protein and HG-7-85-01 ligand, we underscore the importance of understanding protein-ligand interactions for optimizing therapies. Finally, we emphasize the significance of generative models in enhancing molecular docking accuracy and efficiency. Overall, this section sets the stage for our semester project inquiry.

1.1 AI in drug discovery

Artificial Intelligence has made significant breakthroughs in the field of drug discovery, revolutionizing the way new medicines are developed. Drug discovery is a complex and multifaceted process that requires the integration of various scientific disciplines and techniques. AI is now an essential tool in this field, streamlining the discovery process and accelerating the development of new drugs. The different steps are :

— **De Novo Design** : AI techniques, are used in de novo drug design to identify and generate candidate drugs from scratch. By leveraging molecular knowledge and advanced algorithms such as generative models, including generative adversarial networks (GANs) and variational autoencoders (VAEs), AI can design novel compounds that could potentially serve as effective drugs.

— **Target Structure Prediction** : Understanding the structure of biological targets is crucial for drug discovery. AI models based on deep learning like AlphaFold and RoseTTAFold can predict the three-dimensional structure of proteins and other molecular targets based on their amino acid sequences.

— **Drug-Target Interaction** : AI models can assess the interactions between drugs and their targets. By analyzing the molecular properties of both drugs and targets, AI can predict how well a drug will bind to its target.

— **Drug-Target Binding Affinity** : AI can also predict the binding affinity between drugs and their targets. This involves estimating how strongly a drug molecule will bind to its target, which is a critical factor in determining the efficacy of a drug. AI models use various features of the drug and target to predict binding affinity.

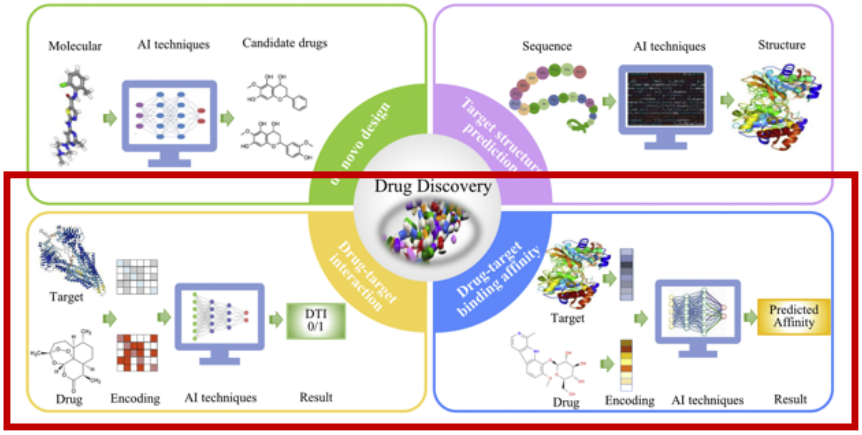


FIGURE 1 – Drug Discovery : a complex and multifaceted process

In this project, we have focused on the two last steps of drug discovery as shown in Figure 1,

which can be summerized into molecular docking known as a computational technique vital for predicting the binding interactions between small molecules (ligands) and target proteins and the ligand's pose including its position and orientation. Molecular docking shown in Figure 2 is essential for determining the potential affinity and stability of the interaction between a ligand and its protein, facilitating the identification of compounds that can effectively bind to **protein sites linked to diseases**. It is crucial for the drug discovery process.

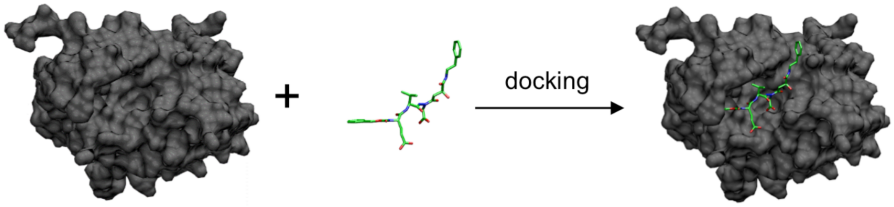


FIGURE 2 – Molecular Docking

1.2 Motivation of molecular docking and studied case

We want to motivate why molecular docking is crucial in drug discovery. To underline that, we have explored in this project the example of the BCR-ABL protein and a ligand.

BCR-ABL is a chimeric protein resulting from the fusion of two genes, the BCR gene and the ABL gene, due to a genetic translocation (exchange of segments between two chromosomes). This genetic fusion leads to the expression of an abnormal protein with constitutive tyrosine kinase activity, meaning excessive enzymatic activity that promotes uncontrolled cell growth.

This protein is involved in the pathogenesis of chronic myeloid leukemia and other types of blood cancer. Treating cancers associated with BCR-ABL often involves targeting the enzymatic activity of the protein.

The ligand is the compound HG-7-85-01, a small molecule that interacts with the BCR-ABL protein. This ligand is a tyrosine kinase inhibitor capable of binding to the protein and blocking its enzymatic activity.

The study of this kind of protein-ligand interaction is motivated by several facts :

— **Understanding Interactions :** By using DiffDock to simulate and visualize the interaction between the BCR-ABL protein and the ligand, you can better understand how the ligand binds to the protein and how it inhibits its activity.

— **Ligand Optimization :** By examining how the ligand binds to the protein, you can identify possibilities for optimizing the ligand's chemical structure to improve its efficacy, selectivity, and reduce side effects.

— **Development of More Effective Therapies :** A detailed understanding of the interaction between BCR-ABL and the ligand allows for the development of more targeted therapies to treat cancers associated with BCR-ABL, potentially leading to more effective and better-tolerated treatments for patients.

— **Experimental Validation :** Docking simulations performed with DiffDock can guide laboratory experiments, suggesting the most promising compounds and targets for in vitro and in vivo testing.

1.3 Generative Models in drug discovery

Moreover, we have focused in this project on generative models that are know to achieve the best results. These are the main reasons :

— Generative models can simulate the entire docking process, considering multiple possible poses and interactions between the ligand and protein, while regression models, on the other hand, typically focus on predicting specific outcomes, such as binding affinity, based on a fixed set of molecular features.

— Generative models can explore a broader range of conformations and poses, leading to a more thorough search for optimal binding configurations.

— Generative models can generate multiple potential docking poses and assess their quality and likelyhood, providing a richer set of outputs for researchers to evaluate, while regression models usually provide a single predicted value (e.g., binding affinity) without offering insight into possible alternative poses as shown in Figure 3.

— Generative models can incorporate dynamic and stochastic elements into their predictions, allowing them to model the inherent uncertainties and variations in molecular interactions, while regression models tend to provide deterministic outputs, which may oversimplify complex docking scenarios.

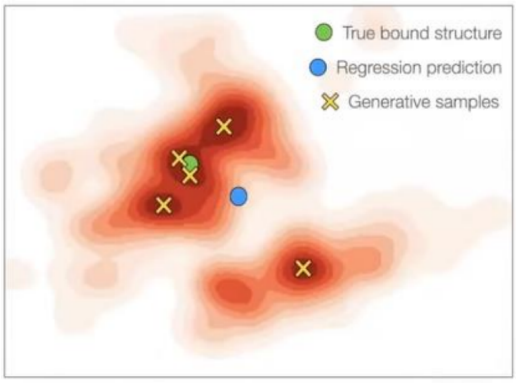


FIGURE 3 – Difference on Molecular Docking between Regression and Generative models [3]

During this project, we explore two main questions : how can generative models be used in molecular docking, and how effective are diffusion models in enhancing the accuracy and efficiency of molecular docking ?

We will begin with an overview of diffusion models and score-based models. Next, we will examine how these models are concretely applied in the molecular docking software DiffDock. Finally, we will analyze some results from DiffDock and compare them with other molecular docking models.

2 Diffusion Models

Diffusion models work by gradually transforming an initial data distribution (noise) into a target data distribution, through an iterative process. This process is guided by models that **learn to reverse these diffusions**, allowing the **generation of samples from noise** by following the **inverse steps of diffusion.**

In this part, we will use mainly the formalism of L. Weng[7]., and from Fidle course from CNRS [1]

We will first introduce Diffusion models, and then I will explain the link with score based diffusion models.

2.1 Diffusion models

First, we'll delve into the theory, followed by a practical case study.

2.1.1 Theory

Forward Diffusion process

Given a data sampled from a real data distribution $x_0 \sim q(x)$, let us define a forward diffusion process in which we add small amount of Gaussian noise to the sample in steps, producing a sequence of noisy samples . Here our data will be images to make it clear. The step sizes are controlled by a variance schedule β_t.

Given an image x_{t-1}, we sample x_t according to this distribution :

$$q\left(x_t \mid x_{t-1}\right) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I\right)$$

hence we have :

$$x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} z_{t-1} \qquad \text{where} \quad z_{t-1} \sim \mathcal{N}(0, I)$$

However, we can show that we can sample directly a noisy image x_t only from x_0, so in only one step because :

$$q\left(x_t \mid x_0\right) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t} x_0, \left(1-\overline{\alpha_t}\right) I\right) \qquad \text{where } \bar{\alpha}_t = \prod_{i=1}^{t} \left(1-\beta_i\right)$$

hence :

$$\boxed{x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t} z_t \qquad \text{where} \quad z_t \sim \mathcal{N}(0, I)} \tag{1}$$

Reverse Diffusion process

It is noteworthy that the reverse conditional probability is tractable when conditioned on x_0 :

$$q\left(x_{t-1} \mid x_t, x_0\right) = \mathcal{N}\left(x_{t-1}; \tilde{\mu}_t\left(x_t, x_0\right), \tilde{\beta}_t I\right)$$

$$\tilde{\mu}_t\left(x_t, x_0\right) = \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t} x_t \quad \tilde{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \beta_t$$

But what we want is to have $q(x_{t-1} \mid x_t)$ to denoise the image, and therefore no longer have the dependency on x_0.

But as $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}z_t$, so $x_0 = \frac{1}{\sqrt{\bar{\alpha}_t}}x_t - \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_{t_t}}}z_t$ we no longer have a dependency on x_0, but only on z_t :

$$\begin{aligned}
\tilde{\mu}_t(x_t, x_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\
\tilde{\mu}_t(x_t, x_0) &= \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}(\frac{1}{\sqrt{\bar{\alpha}_t}}x_t - \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_{t_t}}}z_t) + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t \\
\tilde{\mu}_t(x_t, x_0) &= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}z_t\right)
\end{aligned}$$

So we can introduce $\hat{\mu}_t$ such that : $\tilde{\mu}_t(x_t, x_0) = \hat{\mu}_t(x_t, z_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}z_t\right)$

Eventually :

$$\boxed{q(x_{t-1} \mid x_t, z_t) = \mathcal{N}\left(x_{t-1}; \hat{\mu}_t(x_t, z_t), \tilde{\beta}_t I\right)}$$

$$\hat{\mu}_t(x_t, z_t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}z_t\right) \approx \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}z_\theta(x_t, t)\right)$$

Thus, we will not attempt to denoise the image x_t directly towards the image x_{t-1}, but we will try to learn the noise z_t that was used to transition from x_0 to x_t.

The inverse process corresponds to

$$\boxed{x_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}z_t\right) + \tilde{\beta}_t z \quad \text{where } z \sim \mathcal{N}(0, I)} \tag{2}$$

We can use the variational lower bound to optimize the negative log-likelihood of $p_\theta(x_{t-1} \mid x_t)$. But empirically, the training is easier with a simplified objective :

$$L^{\text{simple}}(\theta) = \mathbb{E}_{t\sim[1,T],x_0,t}\left[\|z_t - z_\theta(x_t, t)\|^2\right]$$

$$L^{\text{simple}}(z_{1:T}, x_0, \theta) \approx \frac{1}{T}\sum_{t=1}^{T}\|z_t - z_\theta(x_t, t)\|^2$$

During training, we will minimize this loss in order to predict, for a given image x_t and time t, the noise z_t that generated it from x_0.

Algorithm 1 : Training

Algorithm 1 Training algorithm

repeat
  $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ ▷ Sample initial data
  $t \sim \text{Uniform}(\{1, \dots, T\})$ ▷ Randomly choose a time step
  $z \sim \mathcal{N}(0, I)$ ▷ Sample noise
  $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1 - \bar{\alpha}_t} z$ ▷ Noising
  Take gradient descent step :

$$\theta \leftarrow \theta - \eta \nabla_\theta \left\| z - z_\theta \left(x_t, t \right) \right\|^2, \text{where} \quad \eta \in \mathbb{R}_+$$

until converged

Algorithm 2 : Sampling

Algorithm 2 Sampling algorithm

$x_T \sim \mathcal{N}(0, I)$ ▷ Initialize at the final step
for $t = T, \dots, 1$ **do**
  if $t > 1$ **then**
    $z \sim \mathcal{N}(0, I)$ ▷ Sample noise
  else
    $z = 0$ ▷ No noise at the first step
  end if
  $x_{t-1} = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} z_\theta(x_t, t)) + \sigma_t z$
end for
return x_0

2.1.2 Practical session : MNIST fashion

This practical session is extracted from CNRS Fidle course : "Diffusion model, text to image", and show an example of image generation. Our dataset is composed of 60 000 32 x 32 images of fashion objects.

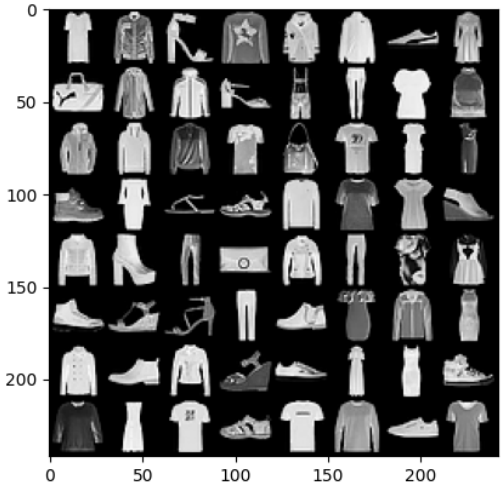


FIGURE 4 – fashion MNIST dataset

First, we want to make **forward diffusion process**. We create a function that will compute every betas of every steps (following a specific shedule). We will only create a function for the linear schedule (original DDPM) and the cosine schedule (improved DDPM).

After that, we define the function $q(x_t|x_0)$:

```
q_sample(constants_dict, batch_x0, batch_t, noise=None)
```

We can now visualise this process for a gradually increasing t on different fashion objects :

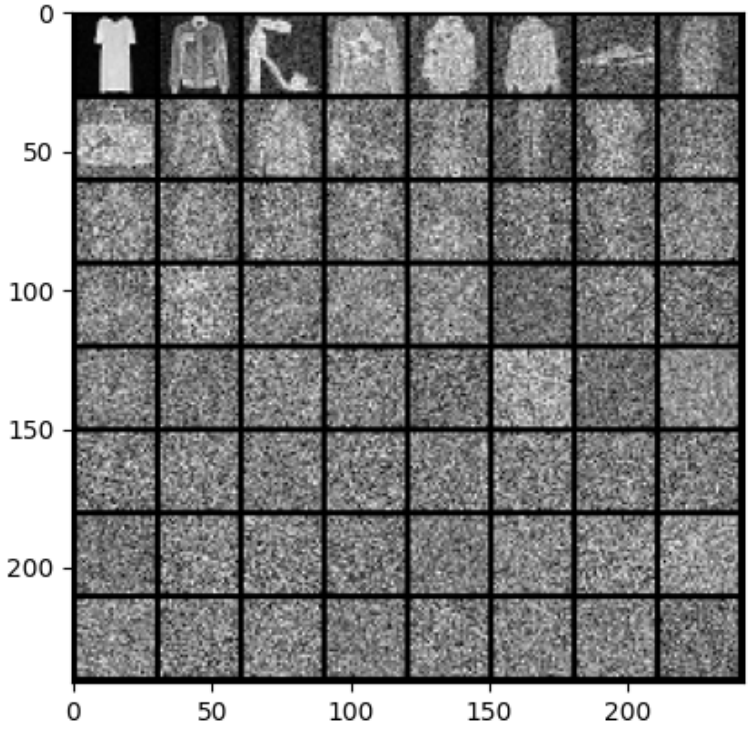


FIGURE 5 – Forward process

We quickly obtain an entirely noisy image.

We then when to code the **reverse diffusion process**, which is made by a deep learning model : Unet with attention.

We also define a function to retrieve x_{t-1} from x_t and the predicted z_t :

```
p_sample(constants_dict, batch_xt, predicted_noise, batch_t)
```

For the **training**, we used 3 epochs and T=10000, an L1 smooth loss, and an ADAM optimizer, and then, we implemented Algorithm 1.

Eventually, We create the **sampling** function (Algorithm 2). Given trained model, it should generate all the images we want.

We can see on the Figure 6, the generation of 64 new samples. Each sample represent a different fashion object. Results are much less detailed than the original data, but we can clearly understand the shapes of each object. We can see t-shirts, dresses, shoes, boots ...

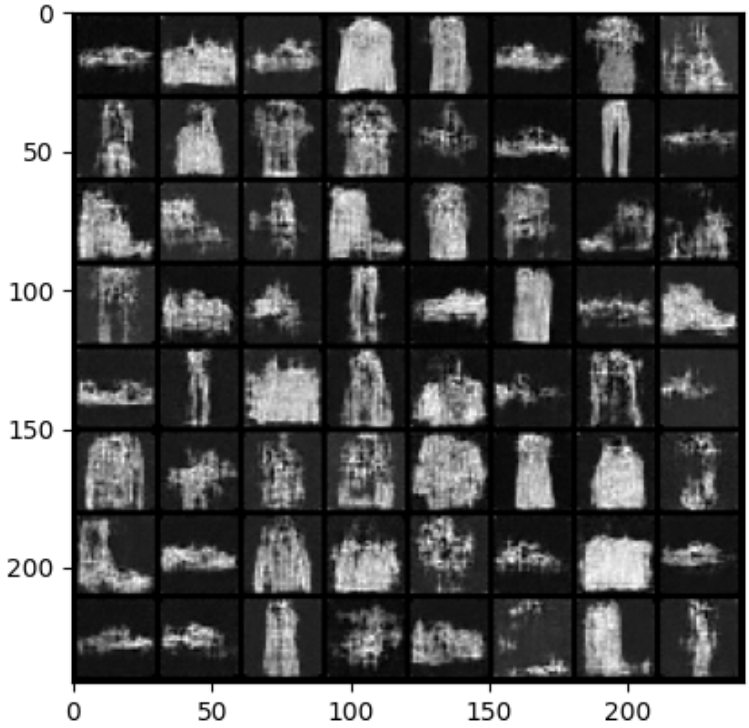


FIGURE 6 – Sampling results

2.2 Connection with score based diffusion models

We now want to make a link between diffusion models and score based diffusion models effectively used in our molecular docking software.

Beginning with an overview of the theory, we'll then proceed to a practical demonstration.

2.2.1 Theory

Score models learn score functions (gradients of logarithmic probability densities) over many data distributions perturbed by noise through **Stochastic Differential Equation (SDE)**, and then generate samples via the reverse SDE.[6]

Langevin dynamics, a concept from physics used to model molecular systems, can be leveraged to simplify sampling from a probability density $q(x)$. Instead of modeling the density function directly, we model the score function, which allows us to avoid the complexities of intractable normalizing constants. Using stochastic gradient Langevin dynamics, we perform stochastic gradient descent to generate samples from $q(x)$. This process involves a Markov chain of updates based solely on the gradients.

$$x^n = x^{n-1} + \frac{\delta}{2}\nabla_x log(q(x^{n-1})) + \sqrt{\delta}z^n \quad \text{where} \quad z^n \sim \mathcal{N}(0, I)$$

where δ is the step size. When $N \to \infty$, x^N equals to the true probability density $q(x)$.

Score-based generative modeling method can produce samples via Langevin dynamics using gradients of the data distribution estimated with score matching. The **score** of each sample $\mathbf{x}$'s density probability is defined as its gradient $\nabla_{\mathbf{x}} \log q(\mathbf{x})$. A score network $\mathbf{s}_\theta : \mathbb{R}^D \to \mathbb{R}^D$ is trained to estimate it, $\mathbf{s}_\theta(\mathbf{x}) \approx \nabla_{\mathbf{x}} \log q(\mathbf{x})$.

Although the majority of data appears to reside in a higher-dimensional space, it actually tends to cluster within a lower-dimensional manifold. This clustering leads to regions of low data density, where the accuracy of estimated score functions is compromised due to a lack of sufficient data points for effective score matching. To mitigate this issue, we introduce small Gaussia, noise to perturb the data points before training score-based models on these altered data points. By applying a sufficiently high magnitude of noise, we can expand the data distribution into these sparse regions, thereby enhancing the accuracy of the score estimates. This strategy improves the stability during the training of the score estimator network. Moreover, by introducing noise at varying levels and training a noise-conditioned score network, we can simultaneously estimate scores for all perturbed data across different noise intensities, further refining our method.

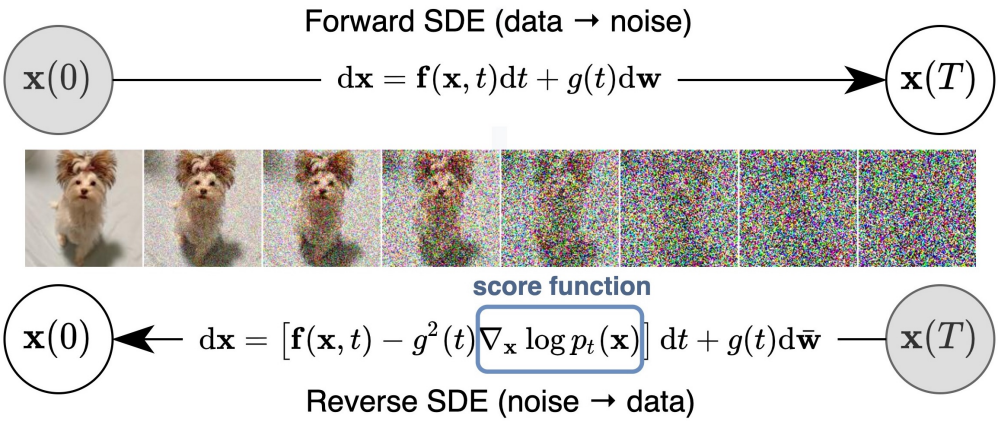
Adding increasing noise corresponds to the forward diffusion process.

If we use the diffusion process annotation, the score approximates $s_\theta(x_t, t) \approx \nabla_{x_t} \log q(x_t)$. Given a Gaussian distribution $x \sim \mathcal{N}(\mu, \sigma^2 I)$, we have :

$\nabla_x \log p(x) = \nabla_x \left(-\frac{1}{2\sigma^2}(x-\mu)^2\right) = -\frac{x-\mu}{\sigma^2} = -\frac{z}{\sigma}$ where $z \sim \mathcal{N}(0, I)$.

Recall that $q(x_t \mid x_0) \sim \mathcal{N}(\sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)I)$ and therefore :

$$s_\theta(x_t, t) \approx \nabla_{x_t} \log q(x_t) = \mathbb{E}_{q(x_0)}\left[\nabla_{x_t} q(x_t \mid x_0)\right] = \mathbb{E}_{q(x_0)}\left[-\frac{z_\theta(x_t, t)}{\sqrt{1-\bar{\alpha}_t}}\right] = -\frac{z_\theta(x_t, t)}{\sqrt{1-\bar{\alpha}_t}}$$



Process of Diffusion and Reverse Diffusion in Score-based Generative Models [6]

We can see that learning the score at each time step $s_\theta(x_t, t)$ is the same as learning the noise $z_\theta(x_t, t)$

We use stochastic differential equations (SDE) to noise our data.

In general, Stochastic Differential Equations (SDEs) take the following form :

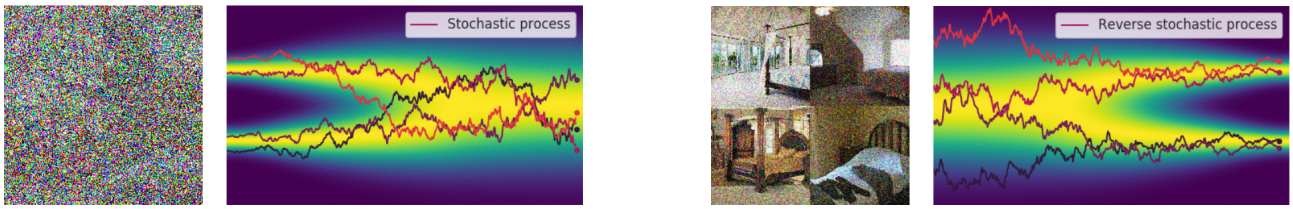
$$dX_t = f(X_t, t)\, dt + g(t, X_t)\, B_t$$

However, SDEs commonly adopt the subsequent form, which we will consider in detail :

$$\boxed{dX_t = f(X_t, t)\, dt + g(t)\, B_t} \tag{3}$$

After that, we can denoise the data at each step with the reverse SDE (because $s_\theta(x_t, t) \approx \nabla_{x_t} \log q(x_t)$) :

$$\boxed{dX_t = \left[f(X_t, t) - g^2(t)\nabla_x \log q(X_t)\right]\, dt + g(t)\, B_t} \tag{4}$$



Forward SDE Reverse SDE

FIGURE 7 – SDE : from Random Noise to Structured Representation with Score Functions [6]

2.2.2 Pratical session

This practical session is extracted from the notebook of Valentin de Bortoli (see GitHub) : We restrict ourselves to a simple two-dimensional setting and want to generate a spiral distribution from a gaussian distribution.

— Forward Noising Process with the following SDE :

$$dX_t = -\frac{1}{2}X_t dt + dB_t, \quad X_0 \sim \pi,$$

.

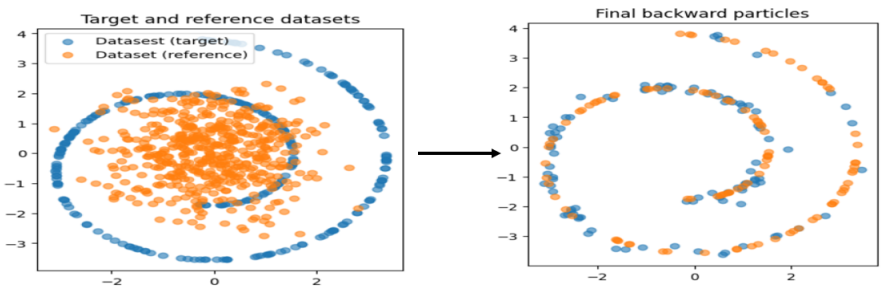
— Backward Denoising Process using the reverse SDE :

$$dX_t = \left(\frac{1}{2}X_t + \nabla \log q_{T-t}(X_t)\right) dt + dB_t, \quad X_0 \sim \mathcal{N},$$

.

We can then do forward noising process with the first equation, and reverse the process thanks to $s_\theta(x_t, t) \approx \nabla_{x_t} \log q(x_t)$ we learned.

T (final time of the SDE) is set to 1 , and N is set to 100, then each timestep is $\Delta_t = \frac{T}{N} = 0.01$. This value means the model updates or evaluates at intervals of 0.01 units of time. The model reverses these noise additions to predict noise perturbations at each step. We train our model with some layers of MLPs (we get embedding for the time input using positional encoding, then we concatenate these embeddings and apply another layer of MLP).



Reverse Diffusion Process

On the left, orange points represent our original points obtained throw our forward process (where we added some noise). On the right picture, we apply the reverse diffusion process and we can see orange points are now situated on the spiral.

3 Diffdock and Results

First, we will explore the link between diffusion models and drug design. Then, we will discuss various methods of measuring and quantifying our results. Finally, we will apply these methods to a more concrete case study.

3.1 Diffdock, a diffusion model

DiffDock is a software containing several AI models, including diffusion models to simulate the binding process between ligands and proteins. By temporally reversing stochastic diffusion equations, DiffDock generates predictions about the optimal position and orientation of ligand molecules relative to the target protein. In diffusion models applied to images, the data that was noisy were pixels ; here, what we noise are the positions, orientations, and atom torsions. This process allows us to predict, from a protein and a ligand, the optimal binding site. At the beginning of the reverse process, ligands are initialized with random positions, orientations, and torsions, then by denoising step by step, the ligands converge towards an optimal position, orientation, and torsion.

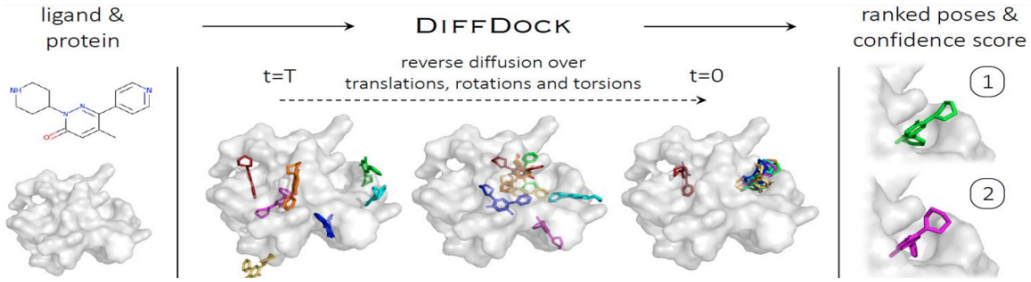


FIGURE 8 – Illustration of reverse diffusion on translations, rotations, and torsions [3]

From a mathematical modeling perspective, we then work in the product space :

$$\mathbb{R}^3 \times SO(3) \times \mathbb{T}^m \quad \text{where} \quad \mathbb{T} = SO(2)$$

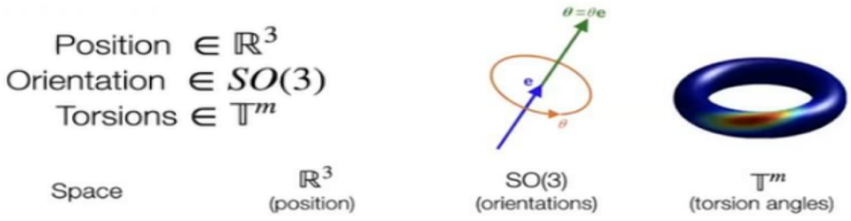


FIGURE 9 – Illustration of the product space [3]

A unique aspect of DiffDock is that it doesn't rely on a single diffusion model for predicting binding, but instead utilizes three independent models, which are trained simultaneously and completely independently. One model is used for predicting position only, a second model for predicting rotations, and a third model for predicting torsions. These three models are trained in parallel, yielding results for positions, orientations, and torsions, which are then concatenated to form our final binding prediction. [3]

Using the form of equation (3) with $f = 0$ and $g(t) = \sqrt{\frac{d\sigma^2(t)}{dt}}$, we employ the following SDE to independently generate the position, orientation, and torsion of the ligand on the protein :

$$d\mathbf{X}_t = \sqrt{\frac{d\sigma^2(t)}{dt}} d\mathbf{B}_t \quad \text{where} \quad \sigma^2 = \sigma^2_{\text{tr}}, \sigma^2_{\text{rot}}, \text{ or } \sigma^2_{\text{tor}} \quad \text{for } \mathbb{R}^3, SO(3), \text{ and } \mathbb{T}^m$$

with an hyperparameter $\sigma^2(t)$, which represents the variance of the stochastic process over time, and it influences how random displacements are generated and propagate in the space of three-dimensional or two-dimensional rotations.

3.2 DiffDock Confidence Score

In addition to the three parallel diffusion models for predicting optimal binding sites, the DiffDock software also offers a scoring model based on deep learning tools. During the training of the diffusion models, a scoring model is also trained, which determines the reliability of the results obtained. Specifically, this score is based on the root mean square distance (RMSD) between the prediction and the theoretical optimal site.

Other models exist to assess the reliability of binding predictions, such as GNINA.[4] (GNU implementation of the Integrated Now-Modeling Architecture). This model is based on deep learning methods like CNNs, and physical principles such as calculating the potential energy of the complex or molecular interactions to predict the affinity of a binding site.

The scoring model of GNINA is therefore based on calculating binding energies, such as the following :

- Van der Waals energy (Lennard-Jones equation) :

$$E_{\text{vdW}} = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right] \tag{5}$$

- Electrostatic energy (Coulomb's equation) :

$$E_{\text{elec}} = \frac{1}{4\pi\epsilon_0} \frac{q_1 \cdot q_2}{r} \tag{6}$$

- Binding energy (sum of contributions) :

$$E_{\text{liaison}} = E_{\text{vdW}} + E_{\text{elec}} + \dots \tag{7}$$

- Total energy (energy of the ligand-protein complex) :

$$E_{\text{total}} = E^{\text{ligand-protein}}_{\text{liaison}} + E^{\text{ligand}}_{\text{internal}} \tag{8}$$

GNINA thus provides a deep learning model to calculate an affinity score based on the total energy of the protein-ligand complex. GNINA offers a physics-oriented approach to scoring the predicted binding site.

However, GNINA also offers an improvement system for predictions. Starting from a predicted binding site, with a docking model like DiffDock, GNINA minimization allows for locally modifying the prediction through gradient descent on position, orientation, and torsions towards a prediction that would minimize the potential energy of the site, or equivalently, maximize ligand-protein interactions.

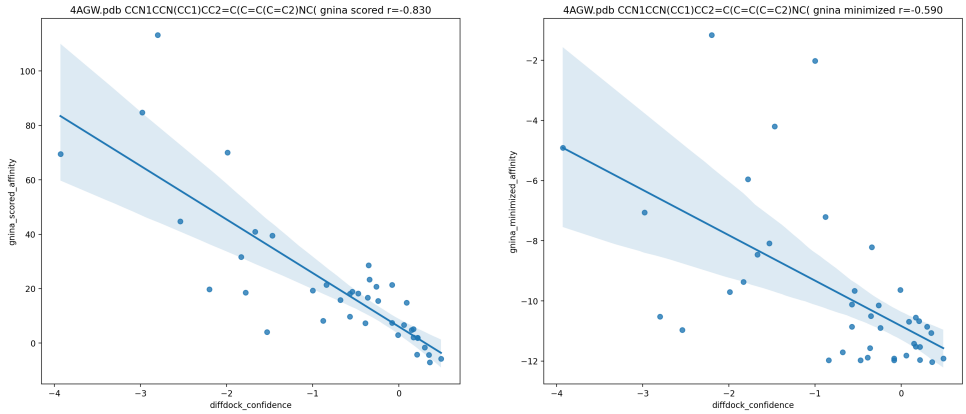
3.3 Results

DiffDock is an online accessible software on Google Colab and already trained. Therefore, we can apply the diffusion models to a case study to obtain the most interesting binding sites according to the DiffDock scoring model.[2]

3.3.1 DiffDock Confidence Score

We select the top 40 predictions with the highest DiffDock scores and plot their DiffDock score against their GNINA affinity. A higher DiffDock score indicates a more coherent prediction according to DiffDock, while a lower GNINA affinity suggests a more coherent prediction according to GNINA. We observe a correlation between the GNINA affinity and the DiffDock score, suggesting that when the prediction is reliable according to DiffDock, it is also reliable according to GNINA.

Next, we refine the predictions using the GNINA process towards local minima of potential energies, which locally moves the predicted binding sites towards a site that maximizes interactions between the protein and the ligand. We still observe the same correlation.



Comparaison of DiffDock Confidence with Gnina Predicted Affinity

3.3.2 Interactions graph

To analyze our results, we can focus on the interactions between the ligand and the protein when the ligand is placed according to the optimal prediction. We observe various types of bonds that ensure the stability of the ligand on the binding site, such as hydrogen bonds, Van der Waals interactions, and Pi-Stacking bonds.

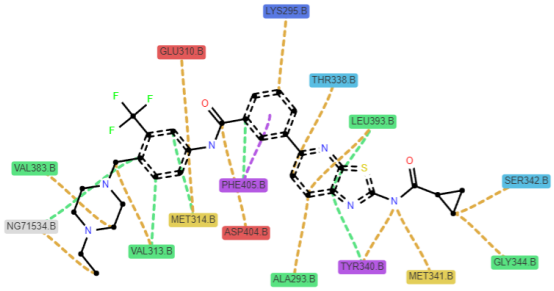


FIGURE 10 – Interactions graph in our studied case

This interactions graph allows for a quantitative comparison of interactions in different protein-ligand complexes, thereby facilitating :

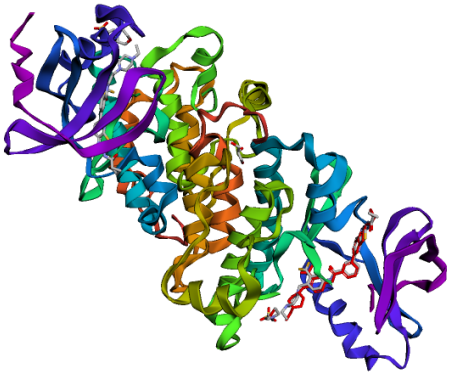
— **The analysis of binding modes :** Identifying how different ligands interact with the same binding sites on a protein, or comparing interactions within different conformations of the same protein.

— **Virtual screening :** Filtering and ranking potential ligands based on their similarity to known binding sites for active ligands, aiding in identifying promising candidates for experimental studies.

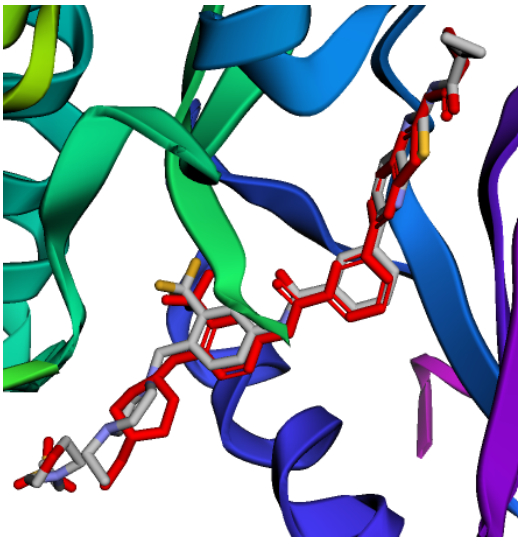
— **Rational drug design :** Guiding ligand modification to enhance binding affinity or selectivity by targeting identified key interactions.

3.3.3 Visualization of results

The obtained results can be easily visualized.



(a) The protein and its ligand



(b) Zoom on the predicted pose and the ground truth

FIGURE 11 – Best confidence score result achieved with DiffDock in our studied case

The results are very promising. As seen in the image, our predictions closely match the actual interactions between the ligand and the BCR-ABL protein. The binding position, torsion, and orientation of the ligand are nearly identical to the real structure, demonstrating the high accuracy and reliability of our approach. This level of precision enhances the confidence in using DiffDock in drug discovery.

3.4 Comparison with other molecular docking software

In this subsection, we highlight how diffusion models, particularly exemplified by DiffDock, outperform other docking algorithms. Supported by empirical evidence from our experiments and corroborated by results published in the DiffDock paper [3].

Diffdock achieve a new state-of-the-art 38% top-1 prediction with RMSD<2A on PDBBind blind docking benchmark, considerably surpassing the previous best search-based (23%) and deep learning methods (20%) as shown in Figure 12 and 13.

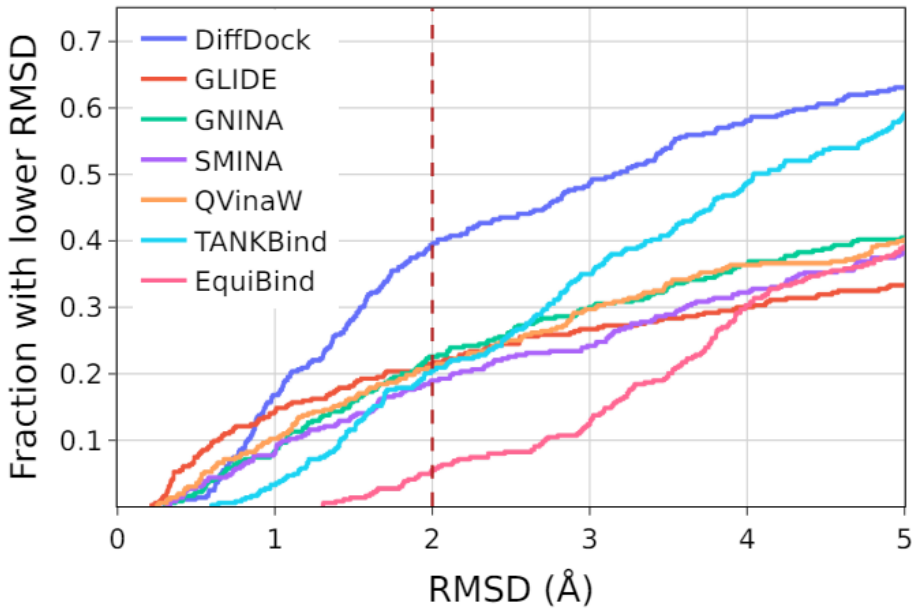


FIGURE 12 – Comparison between multiple docking process through the RMSD of the top 1 prediction [3]

	Holo crystal proteins				Apo ESMFold proteins				
	Top-1 RMSD		Top-5 RMSD		Top-1 RMSD		Top-5 RMSD		Average
Method	%<2	Med.	%<2	Med.	%<2	Med.	%<2	Med.	Runtime (s)
GNINA	22.9	7.7	32.9	4.5	2.0	22.3	4.0	14.22	127
SMINA	18.7	7.1	29.3	4.6	3.4	15.4	6.9	10.0	126*
GLIDE	21.8	9.3							1405*
EQUIBIND	5.5	6.2	-	-	1.7	7.1	-	-	0.04
TANKBIND	20.4	4.0	24.5	3.4	10.4	5.4	14.7	4.3	0.7/2.5
P2RANK+SMINA	20.4	6.9	33.2	4.4	4.6	10.0	10.3	7.0	126*
P2RANK+GNINA	28.8	5.5	38.3	3.4	8.6	11.2	12.8	7.2	127
EQUIBIND+SMINA	23.2	6.5	38.6	3.4	4.3	8.3	11.7	5.8	126*
EQUIBIND+GNINA	28.8	4.9	39.1	3.1	10.2	8.8	18.6	5.6	127
DIFFDOCK (10)	35.0	3.6	40.7	2.65	**21.7**	**5.0**	**31.9**	**3.3**	10
DIFFDOCK (40)	**38.2**	**3.3**	**44.7**	**2.40**	20.3	5.1	31.3	**3.3**	40

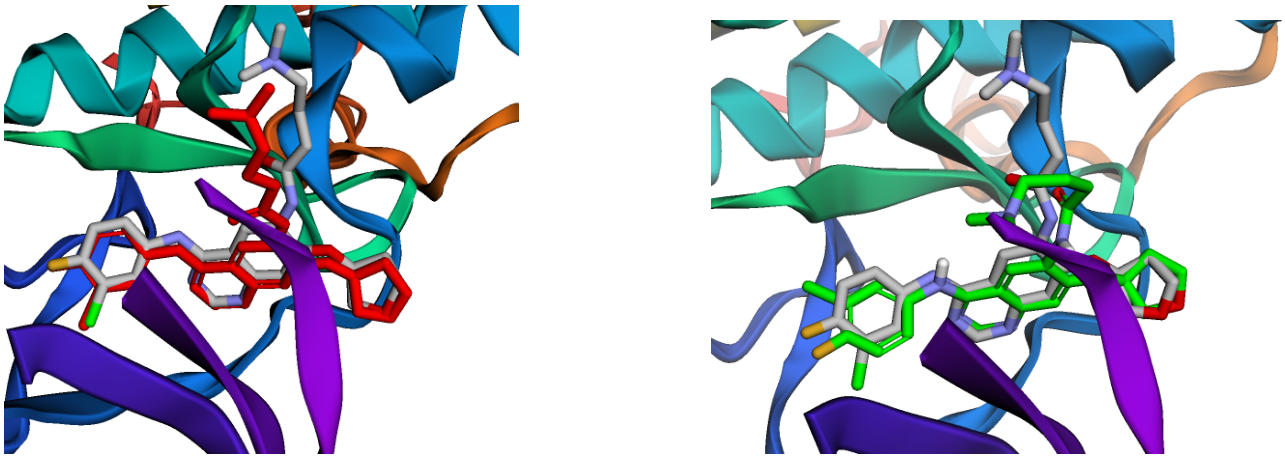
FIGURE 13 – Docking results on many docking algorithm on the PDBBind database [3]

Traditional search-based docking methods utilize physics-based scoring functions and search algorithms to predict ligand-protein interactions. However, these methods are computationally expensive and often inaccurate, especially when faced with the vast search space of blind docking. Despite recent efforts like EquiBind and TANKBind, which leverage machine learning for faster predictions, their performance has yet to match that of traditional search-based methods.

In contrast, diffusion generative models (DGMs) offer a promising alternative. By modeling the score of the diffusing data distribution, DGMs can efficiently generate data. However, existing DGMs are ill-suited for molecular docking due to their distribution learning over the full Euclidean space, which conflicts with the restricted degrees of freedom in docking scenarios. The fact that the space of plausible ligand poses is described by a manifold and that this manifold is easy to find, motivates the development of a diffusion generative model on the manifold rather than the full ambient space $\mathbb{R}^{3n}$. By focusing on this lower-dimensional manifold, we significantly reduce the dimensionality of the state space. This not only simplifies the modeling task but also streamlines the function that the score network needs to approximate. As a result, the approach of Diffdock becomes more computationally efficient and better aligned with the intrinsic constraints and characteristics of ligand-protein interactions.

In terms of inference runtime, Diffdock stands out for its exceptional speed and accuracy. Despite its remarkable precision, operates at a speed that is 3 to 12 times faster than the leading search-based method, GNINA. This rapid processing capability is particularly invaluable for tasks such as high throughput virtual screening to identify potential drug candidates or reverse screening to identify protein targets. In these scenarios, where the exploration of numerous ligand-protein complexes is essential, the efficiency of Diffdock becomes a critical advantage. It's worth noting that, as a diffusion model, Diffdock may exhibit a slightly slower runtime compared to the one-shot deep learning approach employed by equibind. Nonetheless, its balance of speed and accuracy makes Diffdock an appealing choice for a wide range of docking applications.

We will now select a protein and a ligand, run both DiffDock and GNINA, and then compare the results, highlighting the previous findings that suggest DiffDock's superior performance, particularly in prediction accuracy, as evidenced previously.

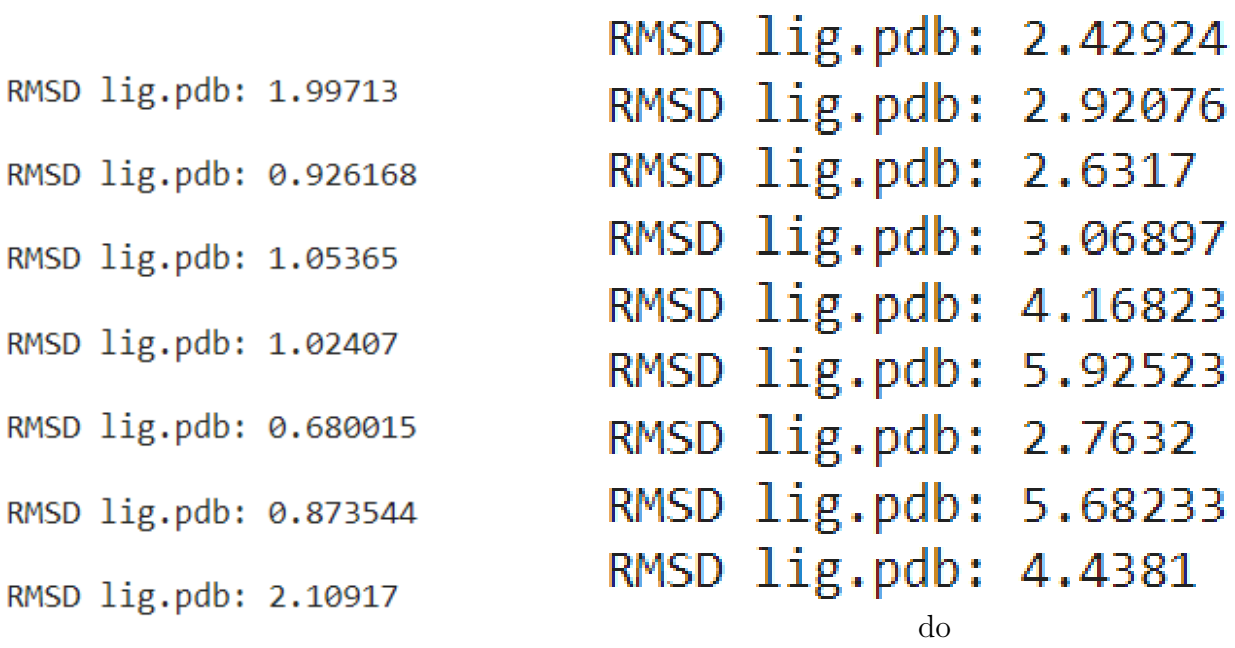


(a) Diffdock (b) GNINA

FIGURE 14 – A protein and the top 1 prediction for both docking algorithm

We easily see, especially in terms of RMSD of the predictions of the predictions15a15b, that Diffdock outperforms GNINA for this particular protein. Moreover, the running time is also much smaller for Diffdock enhancing what was said earlier.

In summary, DiffDock's success underscores the potential of diffusion models in advancing the field of molecular docking. By combining superior accuracy with efficient computation, DiffDock



(a) Diffdock

do

(b) GNINA

FIGURE 15 – RMSD's prediction between the predictions and the groundtruth for both docking algorithm

represents a significant leap forward in ligand-protein interaction prediction.

Conclusion

Molecular docking is pivotal in drug discovery, as it facilitates the prediction of the three-dimensional structure of a protein-ligand complex. This prediction allows for subsequent computational and expert analyses, providing insights into the strength and characteristics of the binding interaction.

We introduced Diffdock, a generative diffusion model tailored for the task of molecular docking, describing the main degrees of freedom of the task through ligand pose transformations spanning the manifold. Empirically, DiffDock outperforms the state-of-the-art by large margins on PDBBind, provides fast inference times, and delivers confidence estimates with high selective accuracy. Thus, DiffDock can offer significant value for many existing real-world pipelines and opens up new avenues of research on the best integration of downstream tasks, such as affinity prediction, within the framework and the application of similar concepts to protein-protein and protein-nucleic acid docking.

However, we are beginning to see the emergence of new technologies for molecular docking, notably with Quantum-Inspired Machine Learning for Molecular Docking [5], which combining quantum-inspired algorithms with deep learning techniques. These methods address challenges in blind docking, where binding sites and conformations are not known in advance, and seek to enhance accuracy and efficiency in predicting binding interactions between drug molecules and target proteins.

To find the several notebooks of this project, visit the GitHub repository : Drug Design Repository.

Références

[1] CNRS. Diffusion model, l'état de l'art du génératif text-to-image ! `https://www.youtube.com/watch?v=V8L766qXmUA`, 2023.

[2] G. Corso. Diffdock : Diffusion steps, twists, and turns for molecular docking. `https://github.com/gcorso/DiffDock`.

[3] G. Corso, H. Stärk, B. Jing, R. Barzilay, and T. Jaakkola. Diffdock : Diffusion steps, twists, and turns for molecular docking. *arXiv*, 2017.

[4] A. McNutt, P. Francoeur, R. Aggarwal, T. Masuda, R. Meli, M. Ragoza, J. Sunseri, and D.R. Koes. Gnina 1.0 : Molecular docking with deep learning. *Journal of Cheminformatics*, 2021. Preprint available on ChemRxiv : https ://chemrxiv.org/engage/chemrxiv/article-details/60c74d2ab32bb8001a5477e1.

[5] Runqiu Shu, Bowen Liu, Zhaoping Xiong, Xiaopeng Cui, Yunting Li, Wei Cui, Man-Hong Yung, and Nan Qiao. Quantum-inspired machine learning for molecular docking.

[6] Yang Song. Score-based generative modeling through stochastic differential equations. `https://yang-song.net/blog/2021/score/`, May 2021. Published on the blog of Yang Song.

[7] Lilian Weng. What are diffusion models ? `https://lilianweng.github.io/posts/2021-07-11-diffusion-models/`, July 2021.