MAESTRÍA EN INTELIGENCIA ARTIFICIAL APLICADA

Materia:

Proyecto Integrador

Resumen Ejecutivo

EQUIPO 29

Miguel Angel Favela Corella A00818504

Kevin Balderas Sánchez A01795149

José Manuel García Ogarrio A01795147

Sponsor del Proyecto

- **Dr. Juan Arturo Nolazco** – Director del Data Science Hub, Tecnológico de Monterrey.

Colaborador Invitado

- **Dr. Carlos Alberto Brizuela Rodríguez** – Investigador, CICESE (Centro de Investigación

1. Síntesis del problema

El diseño de proteínas bioactivas constituye una de las áreas más críticas dentro de la investigación biomédica moderna, especialmente en el tratamiento de enfermedades crónicas como la diabetes tipo 2 y la obesidad. Entre los principales blancos terapéuticos destaca el receptor GLP-1R (Glucagon-Like Peptide-1 Receptor), una proteína de membrana con un papel esencial en la regulación de la glucosa y el metabolismo energético. Su activación por el péptido GLP-1 estimula la secreción de insulina y reduce el apetito, lo que lo convierte en un objetivo ideal para el desarrollo de nuevas terapias.

No obstante, los métodos tradicionales de diseño y validación de proteínas presentan limitaciones significativas: requieren largos periodos de experimentación, recursos económicos considerables y personal altamente especializado. En promedio, el desarrollo de un nuevo fármaco puede extenderse entre 10 y 15 años, lo que limita la capacidad de respuesta ante problemas de salud pública urgentes. Este panorama resulta especialmente preocupante en México, donde la diabetes afecta al 18.3% de la población adulta y mantiene una tendencia ascendente, generando una carga económica y social de gran magnitud.

Frente a esta situación, surge la necesidad de acelerar los procesos de descubrimiento y diseño molecular mediante herramientas de inteligencia artificial. En este contexto, el presente proyecto propone una solución integral que automatiza el diseño, predicción, validación y evaluación de proteínas candidatas con afinidad estructural hacia GLP-1R, orientadas al tratamiento de la diabetes. La propuesta se desarrolla bajo el patrocinio del Dr. Juan Arturo Nolazco (Tecnológico de Monterrey), con la colaboración del Dr. Carlos Alberto Brizuela Rodríguez (CICESE), integrando conocimiento especializado en data science, modelado computacional y biología estructural.

2. Hallazgos más importantes del análisis exploratorio de datos

Durante el desarrollo del proyecto se construyó un pipeline computacional integrado que permite generar y validar proteínas artificiales de forma automatizada, enfocado en péptidos candidatos con afinidad estructural hacia el receptor GLP-1R. Este flujo de trabajo

se estructuró para optimizar cada etapa del diseño computacional, integrando herramientas de última generación:

- RFDiffusion, utilizada para generar estructuras tridimensionales base (backbones) a partir de restricciones espaciales definidas.
- ProteinMPNN, para diseñar secuencias de aminoácidos compatibles con dichas estructuras.
- AlphaFold2, para predecir el plegamiento estructural y evaluar métricas de calidad como pLDDT e ipTM.

En una primera etapa se generaron cientos de secuencias mediante ProteinMPNN a partir de los backbones diseñados con RFDiffusion. Posteriormente, estas secuencias se validaron con AlphaFold, lo que permitió construir una base de datos con valores reales de ipTM y pLDDT. Sin embargo, el cálculo de estas métricas resultó computacionalmente costoso, lo cual evidenció la necesidad de un modelo predictivo capaz de estimarlas sin depender de simulaciones completas.

3. Modelos generados y razones de la elección del modelo final

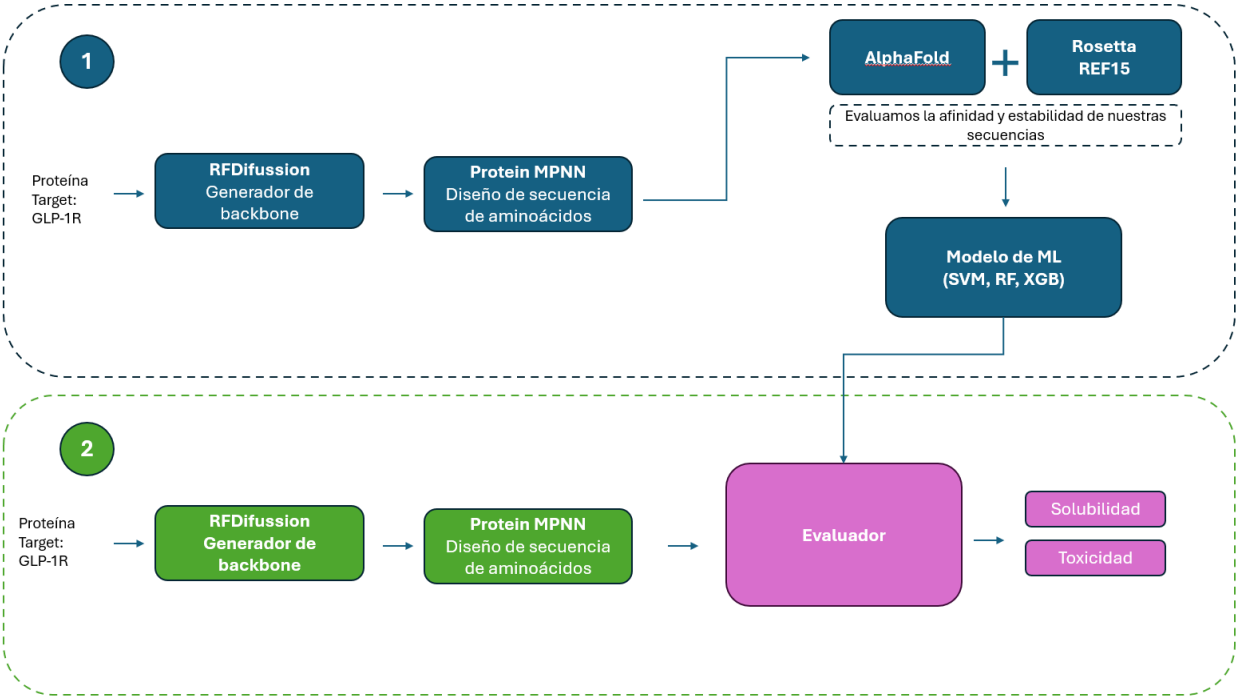
Se utilizaron las siguientes plataformas:

- **RFDiffusion:** para generar estructuras base (backbones) a partir de restricciones espaciales.
- **ProteinMPNN:** para obtener secuencias de aminoácidos compatibles con esas estructuras.
- **AlphaFold2:** para validar la estabilidad y plausibilidad estructural de las secuencias diseñadas.
- **Rosetta (REF15):** para refinar las estructuras y evaluar métricas energéticas avanzadas.

Además, se desarrolló un modelo de aprendizaje automático con el objetivo de predecir la métrica ipTM, empleando embeddings bioquímicos avanzados y algoritmos como MLP y Random Forest.

La arquitectura elegida permite una integración fluida y modular, reduciendo significativamente los tiempos de evaluación sin sacrificar la precisión, gracias al uso de un predictor entrenado con datos reales generados por AlphaFold. A continuación, en el documenta se muestra el flujo de trabajo para diseño computacional y evaluacion de estabilidad proteica.

- **Descripción del Flujo de Trabajo**



El proyecto se estructura en dos etapas principales, representadas en el diagrama correspondiente. Ambas parten de un mismo objetivo: diseñar y evaluar nuevas secuencias peptídicas con potencial afinidad estructural y funcional hacia la proteína GLP-1R (Glucagon-Like Peptide-1 Receptor).

Etapa 1: Generación y evaluación inicial de secuencias

En esta primera fase (color azul), se emplea RFDiffusion para generar el backbone proteico, la estructura tridimensional base sobre la cual se diseñarán las nuevas secuencias. Posteriormente, ProteinMPNN, como ejemplo, generará entre 1,000 y 5,000 secuencias de aminoácidos potencialmente compatibles con dicha estructura.

Estas secuencias se someten a una evaluación estructural y energética mediante herramientas como AlphaFold y Rosetta, con el fin de estimar propiedades clave:

- Afinidad
- Estabilidad conformacional
- Compatibilidad estructural

El principal desafío de esta etapa radica en el alto costo computacional requerido para analizar miles de secuencias con precisión.

Para abordar esta limitación, se propone entrenar modelos de Machine Learning (basado en algoritmos como SVM, Random Forest o XGBoost) y Deep Learning utilizando un subconjunto representativo de las secuencias evaluadas experimentalmente.

Este modelo funcionará como predictor de afinidad y estabilidad, permitiendo reducir drásticamente el tiempo de evaluación sin comprometer la precisión de los resultados.

Etapa 2: Evaluación masiva mediante modelo predictivo

En esta segunda fase (verde y rosa), se replica el proceso de diseño estructural con RFDiffusion y ProteinMPNN, esta vez a mayor escala, generando un número significativamente mayor de secuencias candidatas.

En lugar de realizar simulaciones intensivas para cada secuencia, estas se analizan mediante el modelo predictivo desarrollado en la etapa anterior. Este modelo, basado en Machine Learning/Deep learning, permite estimar de forma rápida propiedades críticas.

Así, el sistema prioriza automáticamente las secuencias con mayor potencial, optimizando el uso de recursos computacionales y acelerando la identificación de candidatos con mejor perfil biotecnológico.

- ## Infraestructura Computacional: Configuración del Entorno de Simulación Especializado

Como parte del proceso, se implementó una solución basada en contenedores Docker para estandarizar el entorno de ejecución y simplificar la utilización de las herramientas por parte del usuario. Esto permite generar las estructuras proteicas y sus secuencias de aminoácidos de forma local, aprovechando recursos de hardware como unidades de procesamiento gráfico (GPU) para acelerar los cálculos.

Este proceso inicial se estructura en 4 etapas consecutivas:

1. Construcción de la imagen Docker

   A partir del archivo Dockerfile proporcionado en el repositorio de github, se construye una imagen que incluye todas las dependencias y programas necesarios para ejecutar RFDiffusion, ProteinMPNN, AlphaFold y Rosetta. Los comandos y pasos específicos de esta instalación se detallan en el apéndice de este informe.

2. Diseño de la proteína esquelética con RFDiffusion

   Mediante la ejecución del script *1_run_rfdiffusion*, se genera la estructura tridimensional de la proteína objetivo. En esta etapa se definen los parámetros esenciales, como la topología general de la proteína, la ubicación de motivos estructurales específicos (*hotspots*) y cualquier restricción de diseño relevante.

3. Optimización y evaluación de secuencias con ProteinMPNN y AlphaFold

   En esta fase final, se ejecuta el script *2_run_mpnn_af*. Utilizando la estructura esquelética generada en la etapa anterior, ProteinMPNN diseña múltiples secuencias de aminoácidos compatibles con dicha estructura. Posteriormente, AlphaFold modela estas secuencias para predecir su estructura tridimensional y calcular métricas de confianza como pLDDT (que evalúa la confianza local por residuo), pTM (puntuación de similitud global del modelo) e ipTM (puntuación de similitud para complejos proteicos). El pipeline selecciona automáticamente las variantes que presentan los mejores valores en estas métricas, identificando así las secuencias más estables y con un plegado estructural más robusto.

4. (Opcional) Ejecución de Rosetta

   Este código evalúa la estabilidad energética de diseños proteicos usando Rosetta. Calcula la energía de cada estructura, identifica residuos problemáticos, y aplica un proceso de relajación para optimizar la conformación. Clasifica los diseños según métricas energéticas y de RMSD, generando un ranking de las variantes más estables

- **Validación Predictiva: Evaluación de Estabilidad mediante Modelos de Machine Learning y Deep Learning**

Tras completar las etapas previas, se procede a la generación de modelos de evaluación, cuyo objetivo es analizar computacionalmente las secuencias peptídicas diseñadas.

Para ello, las cadenas de aminoácidos (por ejemplo, VSLWETVQKW...) se convierten a un formato numérico interpretable por los modelos de Machine Learning. Este proceso se realiza utilizando ESM-2 (Evolutionary Scale Modeling), un modelo de lenguaje proteico preentrenado desarrollado por Meta, capaz de capturar relaciones evolutivas y estructurales entre secuencias.

El procedimiento comprende las siguientes etapas:

- Tokenización: cada secuencia de aminoácidos se transforma en una serie de tokens (unidades discretas), análogas a las palabras en un modelo de lenguaje natural.
- Normalización de Longitud: se aplica padding para igualar la longitud de las secuencias y truncation para recortar aquellas que superan el límite establecido.
- Generación de Embeddings: el modelo ESM-2 procesa los tokens y produce vectores de 1,280 dimensiones, o embeddings, que codifican información estructural y funcional relevante de cada secuencia.

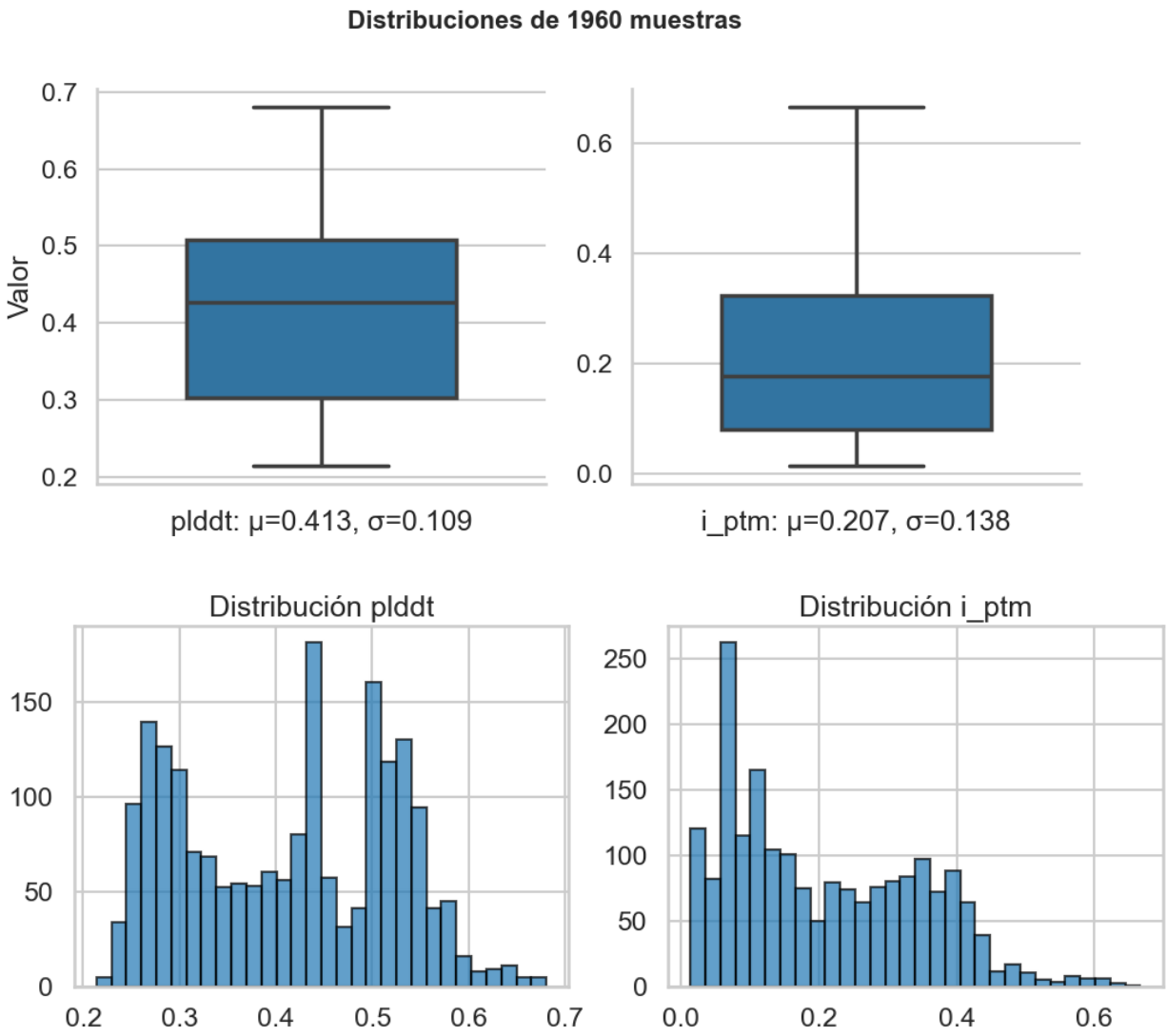
Estos embeddings constituyen la base de entrada para las etapas posteriores de predicción de estabilidad, afinidad y otras propiedades biofísicas, permitiendo un análisis eficiente y escalable del espacio de secuencias generadas.

Se analizaron 1,960 secuencias peptídicas, obteniendo sus valores de pLDDT e iPTM a partir de predicciones de AlphaFold:

- pLDDT: mide la confianza local por residuo en la estructura predicha.

- iPTM: estima la calidad del acoplamiento en interfaces proteína–péptido.

Estas métricas se integraron con los embeddings generados por ESM-2, conformando un dataset numérico para el entrenamiento de los modelos de Machine Learning.

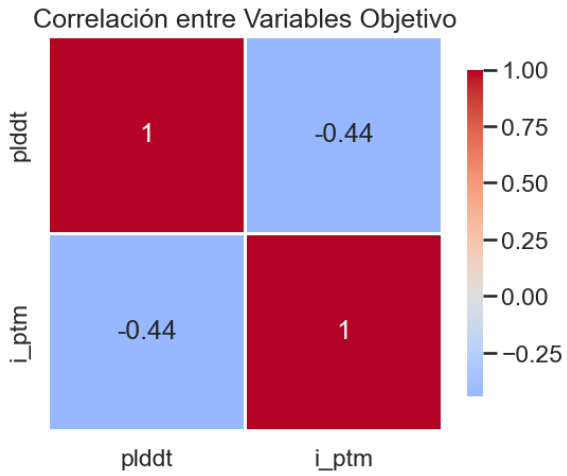


El análisis mostró valores promedio de pLDDT ≈ 0.41 e iPTM ≈ 0.21, lo que sugiere una confianza estructural moderada y baja similitud global entre complejos. Aun así, se identificaron secuencias con picos altos de estabilidad y afinidad, que representan los casos más prometedores.

El objetivo es que, durante el entrenamiento, el modelo predictivo aprenda a reconocer y priorizar estos patrones de alto desempeño, motivo por el cual se definieron múltiples etapas de procesamiento y evaluación en el flujo del proyecto.

Se analizó la correlación entre las métricas estructurales iPTM y pLDDT, obteniéndose un valor de –0.44, lo que indica una correlación negativa moderada. En términos prácticos, esto significa que a mayor confianza local por residuo (pLDDT), tiende a observarse una menor calidad en el acoplamiento global (iPTM).

Aunque, es importante mencionar, a pesar de que existe esta relación inversa entre ambas métricas, esta no es lo suficientemente fuerte como para considerarse determinante, es decir, una alta estabilidad local no garantiza un buen acoplamiento estructural dentro del complejo proteico.



En cuanto a la distribución de las variables, los valores estadísticos obtenidos fueron los siguientes:

Variable	Media (μ)	Desv. Est. (σ)	Asimetría	Curtosis
pLDDT	0.413	0.109	0.015	–1.218
iPTM	0.207	0.138	0.527	–0.669

Las métricas muestran ligeras desviaciones respecto a la normalidad:

- iPTM presenta una asimetría positiva leve, indicando una mayor concentración de valores bajos y una cola extendida hacia valores altos.
- pLDDT muestra una curtosis negativa más pronunciada, lo que sugiere una distribución más plana y dispersa, sin presencia de valores extremos.

En conjunto, estas características reflejan un comportamiento estable, con variaciones leves pero no significativas, por lo que no fue necesario aplicar transformaciones adicionales antes del modelado predictivo.

Previo a la fase de modelado, las secuencias proteicas se procesaron de la siguiente manera:

Dimensión	Significado	Ejemplo
16	Batch de 16 secuencias procesadas al mismo tiempo	16 proteínas distintas
375	Longitud máxima de cada secuencia	Cada letra A, L, G, etc.
1280	Vector numérico (embedding) por residuo	Características latentes aprendidas por ESM-2

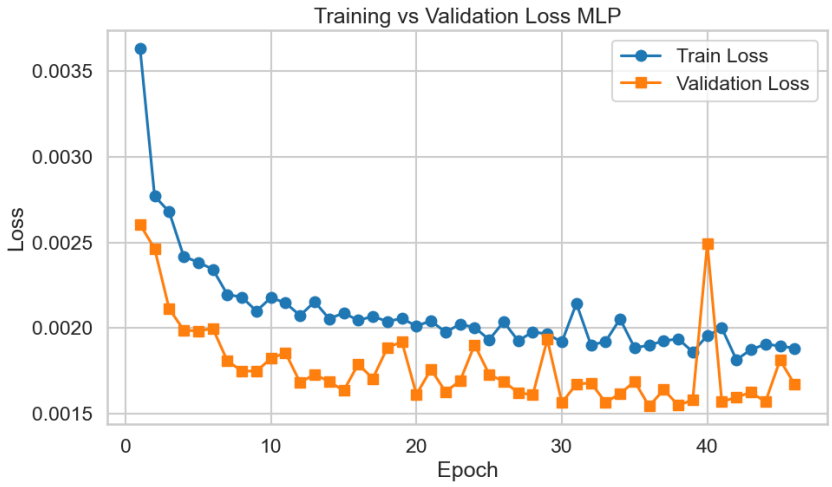
Dependiendo el modelo, se aplicó una operación que calcula el promedio a lo largo de la dimensión de los residuos. Esto generó un vector de 1280 dimensiones por secuencia, representando las características promedio de todos sus aminoácidos

Esta agregación por promedio capturó las propiedades globales de cada secuencia, permitiendo comparar y clasificar las variantes proteicas de manera eficiente.

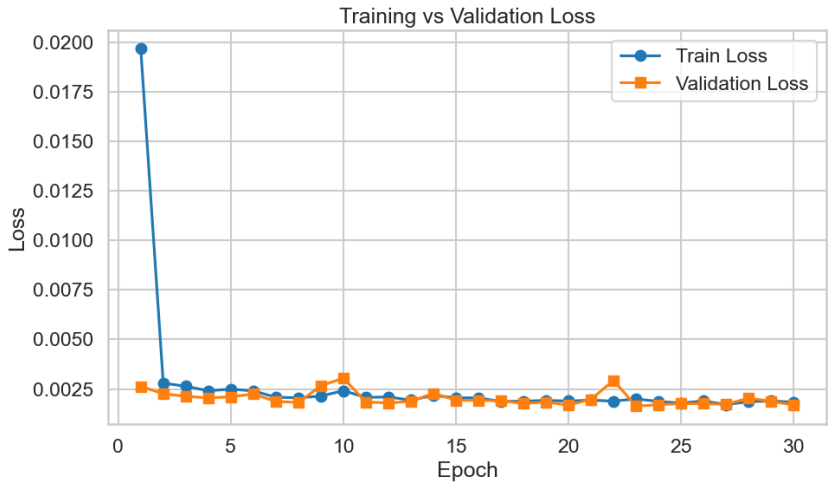
Nota técnica: El modelo ESM-2 tiene un límite máximo de 1024 tokens por secuencia. Dado que nuestras secuencias proteicas tienen longitudes menores a 375 residuos, no se requirió truncamiento adicional durante el procesamiento.

Predicción de pLDDT

Como modelo inicial se probó **Multilayer Perceptron** una red neuronal fully-connected que aprende relaciones no lineales complejas entre los embeddings de ESM-2 y los valores de pLDDT. Configurado con función de pérdida MSE y early stopping (paciencia=10 epocs), el modelo entrena máximo 100 epocs para predecir la estabilidad proteica a partir de las representaciones vectoriales, capturando patrones que modelos lineales no detectarían.



Siguiente se implementó una **red convolucional 1D (CNN)** que procesa los embeddings completos de ESM-2 sin promediar, preservando la información posicional de cada residuo. El modelo entrena máximo 100 epochs con early stopping de paciencia 7, utilizando convoluciones que detectan patrones locales en la secuencia para predecir pLDDT, optimizando con Adam y MSE loss

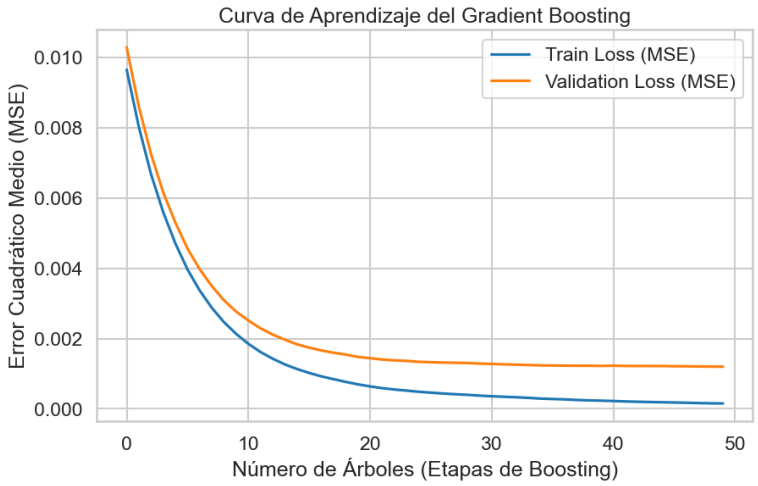


Posteriormente, se implementaron modelos clásicos de machine learning basados en árboles de decisión, que incluyeron técnicas de bagging (Random Forest) y boosting (Gradient Boosting y XGBoost).

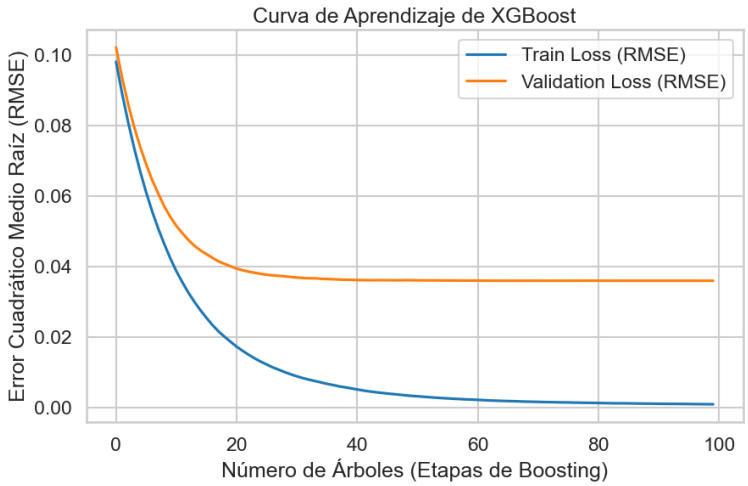
- Random Forest Regressor: se entrenó un ensemble compuesto por 100 árboles de decisión, utilizando los embeddings generados por ESM-2 como variables de entrada para predecir los valores de pLDDT.

```
Evaluando en el conjunto de train

Error Cuadrático Medio (MSE) en train: 0.000179
Evaluando en el conjunto de val

Error Cuadrático Medio (MSE) en val: 0.001246
```

- Gradient Boosting Regressor: se aplicó un modelo de boosting secuencial con 50 árboles y una profundidad máxima de 5. En este esquema, cada árbol se entrena sobre los errores residuales del modelo anterior, permitiendo mejorar progresivamente la capacidad predictiva.



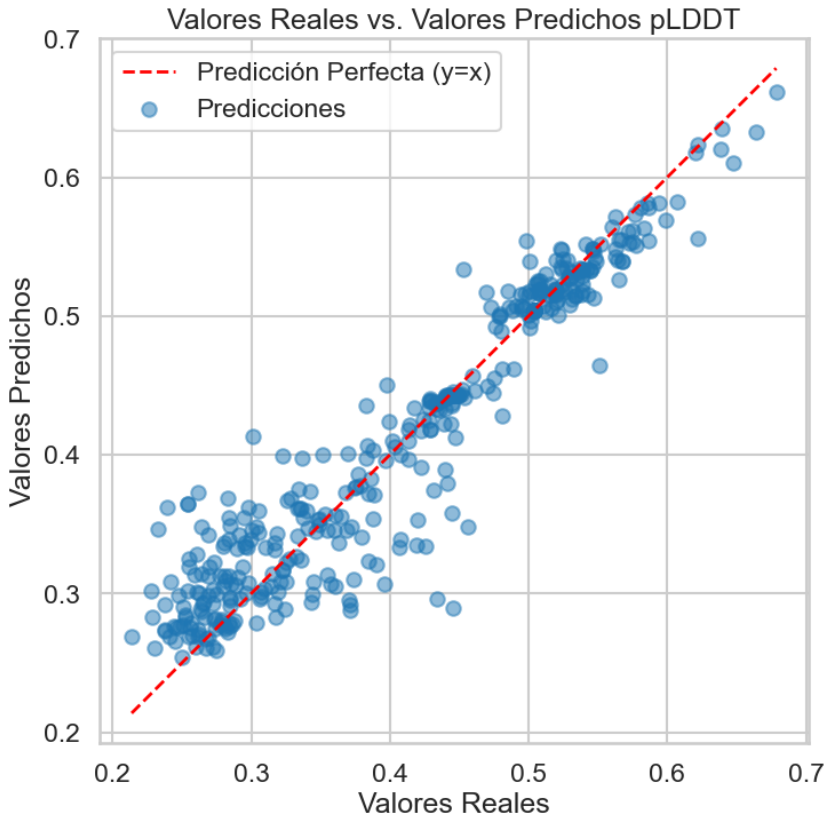
- XGBoost Regressor: se utilizó una implementación optimizada del algoritmo de gradient boosting, configurada con 100 árboles, learning rate de 0.1 y profundidad máxima de 8. Este modelo incorpora regularización L1 y L2 para controlar el sobreajuste, y emplea la métrica RMSE como función de pérdida, permitiendo un equilibrio entre rendimiento y generalización en la predicción basada en los embeddings.



Como parte final, se implemento una modelo lineal, Ridge Regression, con penalización L2 – Alpha = 1 (Mayor regularización) Para prevenir el sobreajuste.

```
MSE del Modelo Ridge Train: 0.00123
MSE del Modelo Ridge Val: 0.00244
```

Por último, se muestra un scatter plot para comparar las predicciones y el valor real para el pLDDT usando el modelo ganador en las muestras de validación. Es interesante observar que igual se obtuvo un R2 = 0.98 para entrenamiento y un R2=0.90 para validación. En puntos alejados a partir de 0.5, el margen de predicción es realtivamente poco. Así que podría usarse para indicar que ese valor es alto en los experimentos que se hagan en el futuro para realmente tenerlos en cuenta.

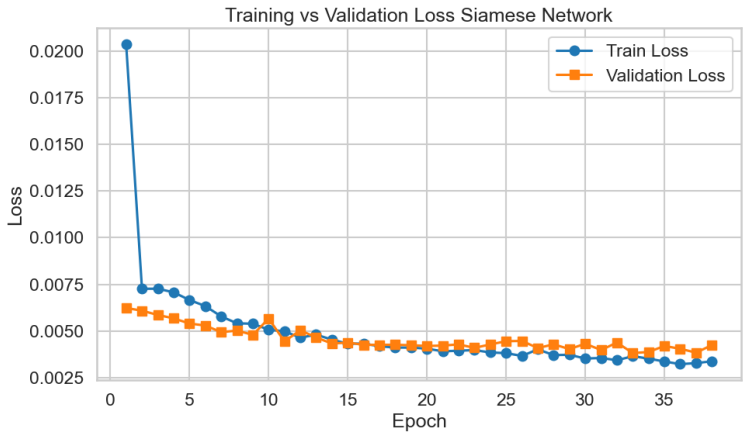


El modelo Gradient Boosting mostró el mejor desempeño predictivo, con los menores errores en validación y tiempos de entrenamiento más eficientes. Su arquitectura secuencial, donde cada árbol corrige los residuales del anterior, demostró mayor capacidad para capturar relaciones complejas en los embeddings que los enfoques de redes neuronales.

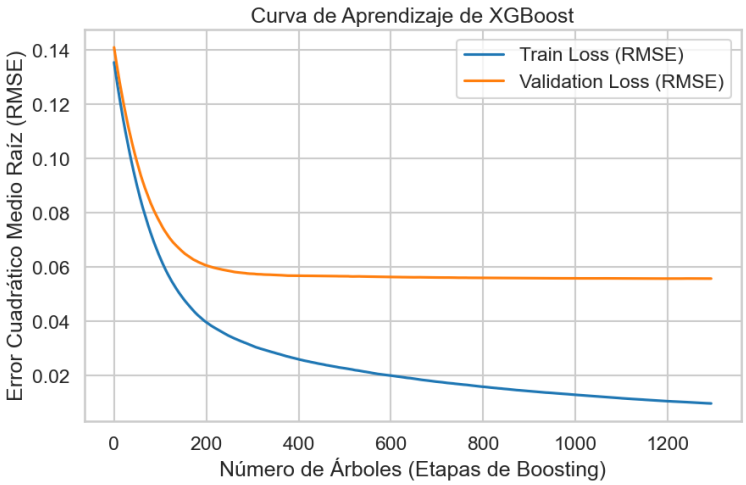
Predicción de iPTM

Como etapa final del pipeline predictivo, se implementó la predicción de la métrica iPTM para evaluar la calidad de interfaces en complejos proteicos. Para esta tarea se utilizaron los siguientes modelos.

Modelo A – Se implementó una red siamesa con dos ramas idénticas que procesan pares de secuencias (Secuencia A y B), utilizando embeddings de ESM-2 promediados por residuo y capas fully-connected (256 → 64 neuronas) para predecir iPTM. El modelo entrenó 100 epocs máximo con early stopping (paciencia=5), optimizando con Adam (lr=0.0001) y MSE Loss.

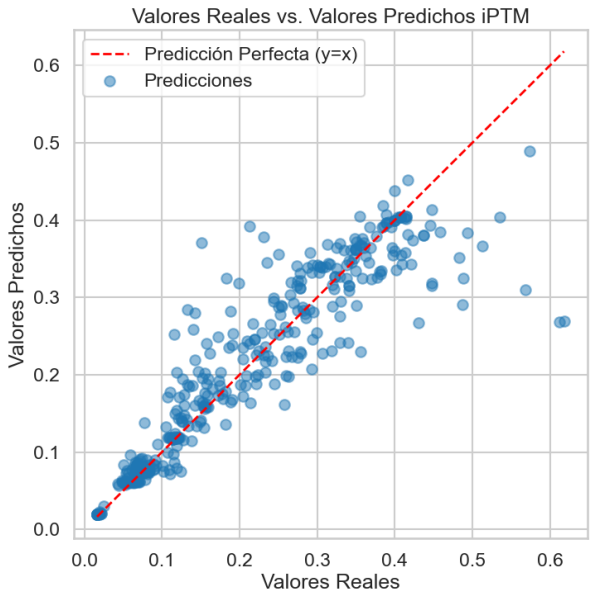


Modelo B – Red Siamesa + XGBoost: Se implementó un modelo híbrido que utiliza la torre siamesa pre-entrenada para convertir cada par de secuencias (A y B) en vectores de 64 dimensiones, calculando posteriormente la diferencia absoluta entre ambos vectores. Los 192 features resultantes (64 de A + 64 de B + 64 de diferencia) se alimentaron a un XGBoost configurado con 2000 árboles, learning rate de 0.01, profundidad máxima de 6 y early stopping de 100 rondas para predecir iPTM.



Modelo C – Red Siamesa + Random Forest: Se implementó un modelo híbrido que utiliza las mismas características de la torre siamesa (192 features: 64 de secuencia A + 64 de secuencia B + 64 de diferencia) para alimentar un Random Forest configurado con 300 árboles y máxima profundidad de 15 para predecir iPTM.

Por último, también observamos un scatter plot para el set de validación y observar que tan alejados estamos, notando que aquí el R2=0.84. Haciendo notar que para valores de 0.1 a 0.3 el margen es muy poco, lo que permite también descartar experimentos de manera rápida ahorrando tiempo y recursos al sólo enfocarse en las demás para investigación.



Para comparar el desempeño predictivo de los modelos desarrollados, se generó la siguiente tabla resumen:

Modelo	Arquitectura	Mejor MSE (Validación)	Mejor RMSE (Validación)
Modelo B	SNN Features + **XGBoost**	~0.00311 (calculado de RMSE)	**0.0558**
Modelo C	SNN Features + **Random Forest**	0.00329	**0.0573**
Modelo A	**Red Siamesa Pura** (End-to-End)	0.0042	**0.0648**

El modelo B (Red Siamesa + XGBoost) mostró el mejor desempeño predictivo para iPTM, donde la red siamesa extrajo características estructurales relevantes y

XGBoost las utilizó para hacer regresión mediante el ensamblado secuencial de 2000 árboles que corrigen residuales iterativamente. Esta combinación de representaciones aprendidas + Gradient Boosting demostró ser más efectiva que el uso único de un modelo de Deep Learning o solo tradicional.

4. Recomendaciones clave para implementar la solución

A partir de los resultados obtenidos durante el análisis exploratorio y la fase de modelado, se formulan las siguientes recomendaciones para la implementación del pipeline computacional. Estas recomendaciones buscan consolidar una solución eficiente, reproducible y sostenible, asegurando la continuidad del proyecto y su potencial de escalamiento.

- **Contenerización del pipeline.**
  Dado que el modelado involucra múltiples dependencias (RFDiffusion, ProteinMPNN, AlphaFold), se recomienda mantener la ejecución bajo un entorno Docker. Esta práctica garantiza la reproducibilidad del sistema, facilita la colaboración entre instituciones y minimiza errores derivados de configuraciones locales. La contenerización es especialmente importante para la futura actualización del modelo predictor de *ipTM* y la integración de nuevas arquitecturas de aprendizaje.

- **Automatización del flujo de trabajo**.
  Los hallazgos del EDA y los resultados del modelado demostraron que el procesamiento manual de miles de secuencias demanda recursos significativos. Por ello, se sugiere la automatización del flujo RFdiffusion → ProteinMPNN → AlphaFold, incorporando mecanismos de ejecución secuencial y validación automática. Esto permitirá optimizar la asignación de recursos, reducir el tiempo de ejecución y mantener un control de calidad uniforme.

- **Modularidad e integración flexible.** Cada componente del pipeline debe conservar un diseño modular, de forma que pueda ejecutarse y actualizarse de manera independiente. Esta arquitectura, ya empleada durante el modelado, facilita la incorporación de nuevas métricas o modelos sin necesidad de reconstruir el sistema completo. Además, permite adaptar el flujo a otros receptores o proteínas, ampliando la aplicabilidad del proyecto en contextos biomédicos diversos.

5. Análisis costo – beneficio

A lo largo del desarrollo de este proyecto, se buscó mantener un equilibrio entre la eficiencia computacional y el uso racional de recursos. Dado que el objetivo principal era implementar un pipeline para el diseño automatizado de proteínas candidatas dirigidas al receptor GLP-1R, se decidió realizar el despliegue en infraestructura local. Esta elección permitió controlar los costos operativos y garantizar un dominio completo del entorno técnico, aunque implicó inversiones en tiempo de implementación y uso intensivo de recursos físicos.

Análisis de costos

El análisis de costos se estructura conforme a las fases de la metodología CRISP-ML. Durante la etapa de comprensión del negocio y de los datos, no se incurrieron gastos económicos directos, ya que se utilizó información pública y bibliografía científica especializada. Esta fase se basó en la revisión de literatura y la exploración de bases de datos estructurales, como el Protein Data Bank (PDB) y repositorios de grupos de investigación en diseño de proteínas.

En la fase de preparación de datos, se realizaron tareas de conversión y estandarización de archivos estructurales para su uso en herramientas como ProteinMPNN y AlphaFold.

La fase de modelado computacional fue la más intensiva en recursos. Se generaron secuencias mediante RFdiffusion y ProteinMPNN, las cuales posteriormente se evaluaron con AlphaFold para obtener métricas estructurales (pLDDT e ipTM). Este proceso se ejecutó en estaciones de trabajo personales equipadas con GPUs, incluyendo una NVIDIA RTX 3070, además de GPUs móviles como la RTX 3050 y la GTX 1650. El uso de GPU permitió reducir los tiempos de cómputo hasta en un 5000 % frente al uso de CPU, justificando plenamente la inversión en aceleradores gráficos.

Componente	Tiempo procesamiento (min) ProteinMPNN-Alphafold “Valor Contig” 20-20/0 R30-127/R138-336/R345-400
CPU **AMD Ryzen 5 3500X 6-Core Processor**	48 mins – 8 secuencias analizadas **~ 10 secuencias por hora**
GPU **NVIDIA GeForce RTX 3070** **Dedicated GPU memory 8.0 GB**	57 mins – 520 secuencias analizadas **~ 547 secuencias por hora**

Para estimar los costos operativos, se consideró el consumo energético de la RTX 3070 en un entorno local de Monterrey, Nuevo León, con un tiempo de actividad del 70 % mensual. El consumo estimado fue de 201.6 kWh/mes, equivalente a $352.80 MXN/mes, considerando una tarifa promedio de $1.75 MXN/kWh. Este valor incluye el gasto energético de todo el sistema (GPU, procesador, ventilación y memoria).

Finalmente, la fase de implementación se centró en la contenerización del pipeline completo mediante Docker, lo que garantiza su reproducibilidad y escalabilidad.

Análisis de beneficios

Los beneficios derivados del proyecto se manifiestan en distintos niveles. Desde el punto de vista operativo, el pipeline desarrollado permite optimizar los tiempos de diseño y validación de proteínas, lo que se traduce en un proceso más ágil y eficiente de experimentación. Esta mejora en la productividad permite dedicar mayores esfuerzos al análisis científico y a la interpretación de resultados, reduciendo la carga manual de tareas técnicas repetitivas.

En el ámbito técnico, el modelo predictor de ipTM ofrece la posibilidad de realizar evaluaciones preliminares sin necesidad de ejecutar simulaciones completas con AlphaFold. Esto implica un uso más racional de los recursos computacionales y una mayor capacidad para explorar alternativas moleculares dentro de un mismo periodo de trabajo.

De acuerdo con Deloitte Access Economics (2017), los beneficios del aprendizaje automático no se limitan al ahorro financiero, sino que impulsan la innovación, la eficiencia y la creación de valor organizacional. En coherencia con esta visión, los beneficios intangibles del presente proyecto incluyen el fortalecimiento de capacidades científicas en biología computacional, la colaboración interinstitucional entre el

Tecnológico de Monterrey y CICESE, y la formación de talento especializado en la intersección entre inteligencia artificial y biomedicina.

En conjunto, los costos operativos del proyecto se mantienen moderados frente al valor generado. A cambio, se obtiene una solución reproducible y escalable, con un potencial tangible de aplicación en el diseño de fármacos y un impacto significativo en la investigación biomédica.

6. Riesgos y desafíos de la solución

El desarrollo de proteínas y péptidos mediante inteligencia artificial representa una frontera prometedora en la investigación biomédica, pero también conlleva una serie de riesgos y desafíos que deben abordarse con prudencia. En este proyecto, la aplicación de modelos generativos para el diseño de péptidos dirigidos al tratamiento de la diabetes introduce interrogantes que van más allá del ámbito técnico, alcanzando dimensiones biológicas, éticas y legales.

En primer lugar, desde una perspectiva biológica, el principal desafío radica en la incertidumbre inherente a las moléculas generadas artificialmente. Aunque los algoritmos pueden optimizar la estructura y prever interacciones moleculares con gran precisión, el comportamiento real de los péptidos en sistemas biológicos es complejo y, en ocasiones, impredecible. Existen riesgos potenciales de reacciones no deseadas, baja estabilidad o interacción con otras rutas metabólicas. Por esta razón, la validación experimental sigue siendo indispensable y constituye un punto crítico dentro del proceso.

Tambien, es necesario reflexionar sobre las implicaciones comparativas entre el diseño artificial y el convencional. Mientras el método tradicional se caracteriza por su rigor experimental y la acumulación de conocimiento empírico, el diseño asistido por IA se distingue por su velocidad, automatización y capacidad de exploración masiva del espacio químico. Sin embargo, esta misma rapidez puede generar un riesgo de desconexión entre el proceso de descubrimiento y la comprensión profunda de los mecanismos biológicos involucrados. Por ello, el verdadero valor del enfoque no radica en sustituir el método clásico, sino en complementarlo: aprovechar la eficiencia de la inteligencia artificial sin renunciar a la verificación científica que caracteriza al desarrollo farmacéutico responsable.

En grandes rasgos, los riesgos identificados se agrupan en tres categorías:

- **Biológicos:** inestabilidad o reactividad imprevista de los péptidos.
- **Éticos:** sesgos en los datos, transparencia y rendición de cuentas.

- **Técnicos:** dependencia de datos de entrenamiento y actualización de librerías.

Para mitigarlos, se propone un esquema de validación continua, documentación exhaustiva del pipeline y supervisión ética institucional. Estos mecanismos aseguran la confiabilidad de los resultados y garantizan el cumplimiento de las buenas prácticas de investigación biomédica.

8. Conclusión

A pesar de los riesgos y desafíos que conlleva la incorporación de la inteligencia artificial en el campo biológico, su implementación marca un punto de inflexión en la forma en que concebimos la investigación y el desarrollo farmacéutico. El diseño de péptidos mediante IA no busca sustituir la experimentación tradicional, sino amplificar su alcance a través de la automatización, la precisión y la velocidad de análisis. Esta convergencia entre biología e inteligencia artificial abre nuevas fronteras en la búsqueda de soluciones terapéuticas, permitiendo reducir los tiempos de diseño y validación de moléculas con potencial clínico.

Como señala Deloitte (2017), la adopción de machine learning genera beneficios que trascienden el aspecto económico, impulsando la innovación y el bienestar social. En este contexto, el presente proyecto no solo representa un avance técnico, sino también una oportunidad para mejorar la calidad de vida de millones de personas afectadas por enfermedades crónicas como la diabetes. Si se gestiona con ética, rigor y responsabilidad, la inteligencia artificial puede consolidarse como una herramienta transformadora, capaz de redefinir los límites de la biomedicina.

9. Bibliografía

- Jumper, J., Evans, R., Pritzel, A., et al. (2021). *Highly accurate protein structure prediction with AlphaFold*. Nature, 596(7873), 583–589.
- Watson, J. L., Juergens, D., Bennett, N. R., et al. (2023). *De novo design of protein structure and function with RFdiffusion*. bioRxiv. https://doi.org/10.1101/2023.03.20.533421
- Anand, N., & Achim, T. (2022). *ProteinMPNN: A deep learning model for protein sequence design*. bioRxiv. https://doi.org/10.1101/2022.07.20.500902

- Baccarin, F. (2020, diciembre 28). *Machine learning that pays the bills: Choosing models in business contexts.* Medium. https://towardsdatascience.com/machine-learning-that-pays-the-bills-choosing-models-in-business-contexts-e9003fd434a1
- Babic, B., Cohen, I. G., Evgeniou, T., & Gerke, S. (2021). *When Machine Learning Goes Off the Rails.* Harvard Business Review. https://hbr.org/2021/01/when-machine-learning-goes-off-the-rails
- Deloitte Access Economics. (2017). *Business impacts of machine learning.* Deloitte. https://www2.deloitte.com/content/dam/Deloitte/tr/Documents/process-and-operations/TG_Google%20Machine%20Learning%20report_Digital%20Final.pdf

10. Apéndices

Instalación contenedor Docker en equipo de computo local. (Se asume que el equipo ya cuenta con Docker instalado)

1.- Estructura de carpetas

Crea una carpeta de trabajo principal, por ejemplo:

```
RFDiffusion_Project/
|
├── Dockerfile
├── PyRosetta4.Release.python310.linux.release-387.tar.bz2
├── requirements.txt
|
├── outputs/                # Carpeta donde se guardarán los resultados
├── 1_run_rfdiffusion.py
├── 2_run_mpnn_af.py
└── 3_run_rosetta.py
```

Ingrese al siguiente enlace:

https://graylab.jhu.edu/download/PyRosetta4/archive/release/PyRosetta4.Release.python310.linux/

y descargue el archivo: ***PyRosetta4.Release.python310.linux.release-387***

2. Creación del contenedor Docker

Construir la imagen

Abre CMD o PowerShell en la carpeta del proyecto y ejecuta:

```
C:\Users\Jose Manuel Garcia\Documents\RFDifussion>docker build -t rfdiffusion .
```

Esto creará una imagen llamada ***rfdiffusion*** basada en tu Dockerfile. Incluye dependencias de RFdiffusion, PyRosetta y las librerías necesarias.

3. Configuración del entorno virtual

Aunque Docker ya aísla el entorno, se crea un entorno virtual para posteriormente correr nuestros modelos de machine learning en el proceso predictor.

python3 -m venv protein_env

protein_env\Scripts\activate

Instalar dependencias:

pip install -r requirements.txt

4. Ejecución de los scripts

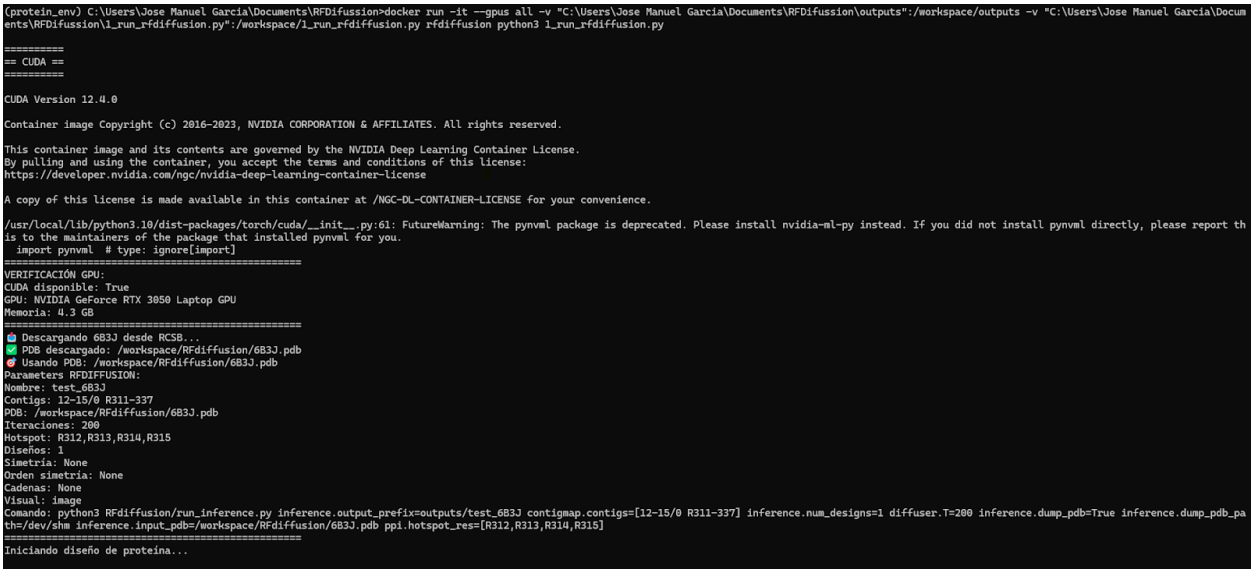
Cada script de Python se ejecutará dentro del contenedor. La idea es montar las carpetas locales de tu PC en /workspace dentro de Docker

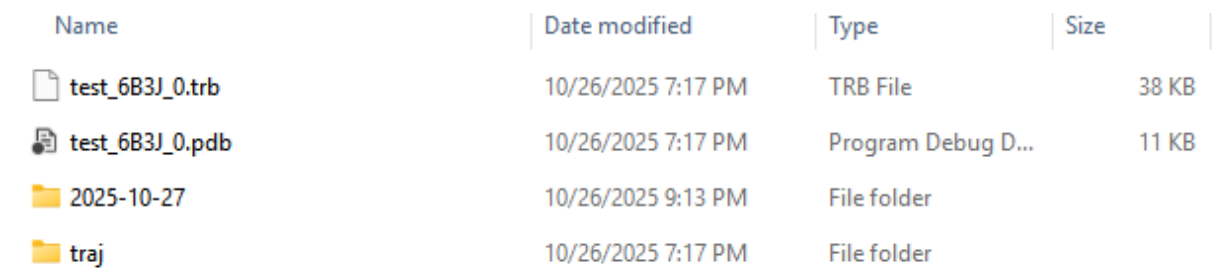
4.1 RFdiffusion — Generación de estructuras

docker run -it --gpus all -v "C:\Users\Jose Manuel Garcia\Documents\RFDifussion\outputs":/workspace/outputs -v "C:\Users\Jose Manuel Garcia\Documents\RFDifussion\1_run_rfdiffusion.py":/workspace/1_run_rfdiffusion.py rfdiffusion python3 1_run_rfdiffusion.py

Esto:

- Usa tu GPU (--gpus all)
- Monta tu carpeta local de resultados (outputs)
- Ejecuta el script 1_run_rfdiffusion.py dentro del contenedor.



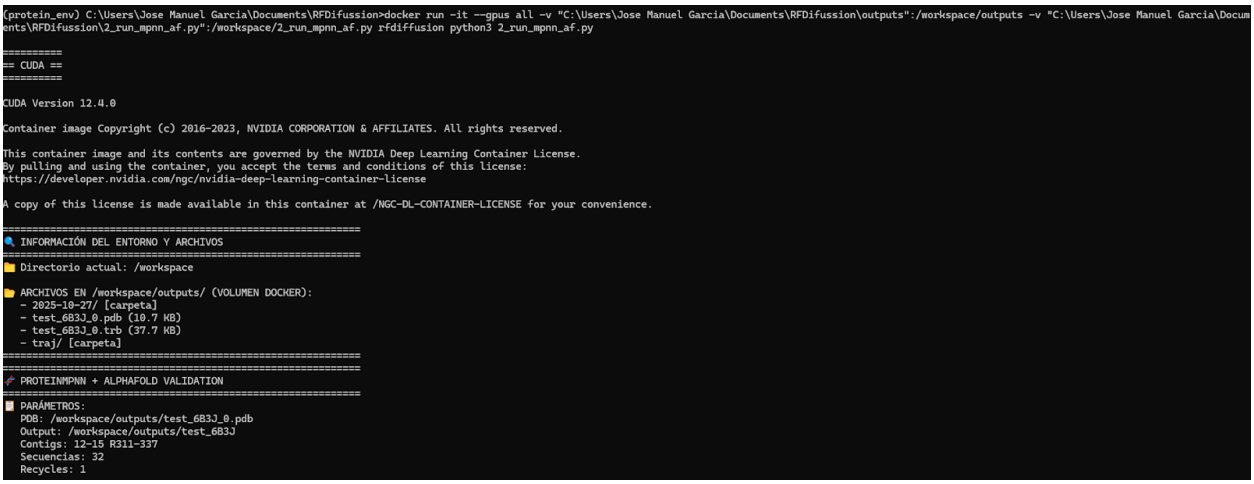


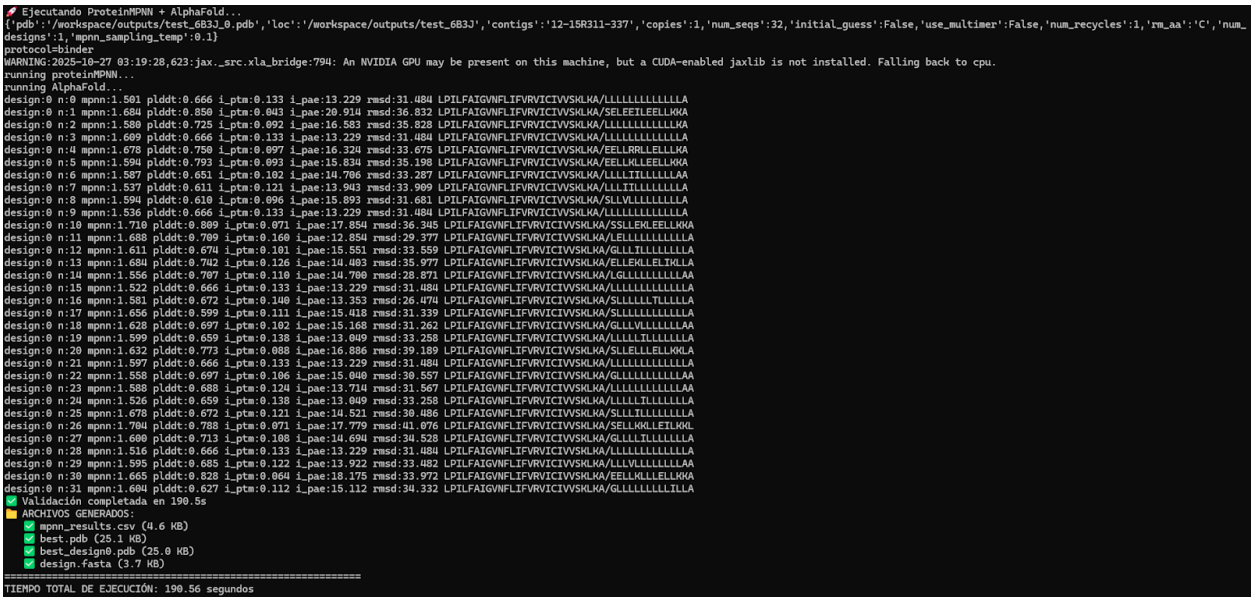
4.2 ProteinMPNN — Generación de secuencias

Luego, usa el modelo 3D generado por RFdiffusion:

docker run -it --gpus all -v "C:\Users\Jose Manuel Garcia\Documents\RFDifussion\outputs":/workspace/outputs -v "C:\Users\Jose Manuel

Garcia\Documents\RFDifussion\2_run_mpnn_af.py":/workspace/2_run_mpnn_af.py rfdiffusion python3 2_run_mpnn_af.py





ProteinMPNN: genera secuencias de aminoácidos que se ajustan a la estructura 3D dada.

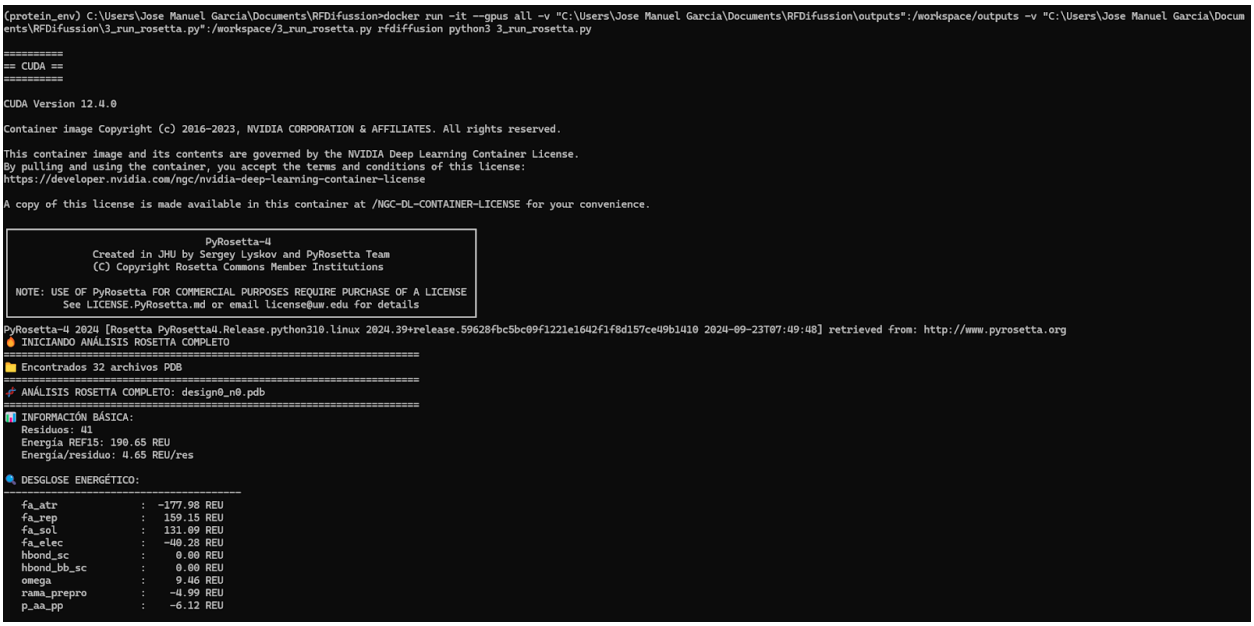
AlphaFold: predice cómo se doblaría esa secuencia, y sirve para verificar si la secuencia propuesta realmente adopta la forma deseada.

4.3 Rosetta — Refinamiento estructural

Finalmente, ejecuta el tercer script dentro del mismo contenedor:

docker run -it --gpus all -v "C:\Users\Jose Manuel Garcia\Documents\RFDifussion\outputs":/workspace/outputs -v "C:\Users\Jose

Manuel Garcia\Documents\RFDifussion\3_run_rosetta.py":/workspace/3_run_rosetta.py rfdiffusion python3 3_run_rosetta.py



```
================================================================================


================================================================================
🏆 REPORTE COMPARATIVO - MEJORES DISEÑOS
================================================================================
Archivo                   Residues Energy/Res RMSD   Estado
--------------------------------------------------------------------------------
 1. design0_n10.pdb         41       3.04       8.74   PROBLEMA
 2. design0_n2.pdb          41       3.42       3.14   PROBLEMA
 3. design0_n1.pdb          41       3.76       4.50   PROBLEMA
 4. design0_n29.pdb         41       3.82       2.41   PROBLEMA
 5. design0_n5.pdb          41       4.38       2.58   PROBLEMA
 6. design0_n0.pdb          41       4.65       1.86   PROBLEMA
 7. design0_n15.pdb         41       4.65       1.95   PROBLEMA
 8. design0_n21.pdb         41       4.65       1.96   PROBLEMA
 9. design0_n28.pdb         41       4.65       2.03   PROBLEMA
10. design0_n3.pdb          41       4.65       1.96   PROBLEMA
11. design0_n9.pdb          41       4.65       2.03   PROBLEMA
12. design0_n23.pdb         41       4.69       2.01   PROBLEMA
13. design0_n18.pdb         41       5.15       2.85   PROBLEMA
14. design0_n27.pdb         41       5.20       2.04   PROBLEMA
15. design0_n14.pdb         41       5.38       3.67   PROBLEMA
16. design0_n22.pdb         41       5.51       3.67   PROBLEMA
17. design0_n4.pdb          41       5.63       2.58   PROBLEMA
18. design0_n20.pdb         41       5.81       3.73   PROBLEMA
19. design0_n30.pdb         41       5.98       6.42   PROBLEMA
20. design0_n11.pdb         41       6.04       2.56   PROBLEMA
21. design0_n13.pdb         41       6.17       2.48   PROBLEMA
22. design0_n26.pdb         41       6.45       3.84   PROBLEMA
23. design0_n25.pdb         41       6.53       1.85   PROBLEMA
24. design0_n12.pdb         41       7.48       2.52   PROBLEMA
25. design0_n31.pdb         41       8.69       4.30   PROBLEMA
26. design0_n8.pdb          41       11.24      1.41   PROBLEMA
27. design0_n6.pdb          41       14.78      1.77   PROBLEMA
28. design0_n17.pdb         41       21.29      2.16   PROBLEMA
29. design0_n19.pdb         41       23.44      1.43   PROBLEMA
30. design0_n24.pdb         41       23.44      1.43   PROBLEMA
31. design0_n7.pdb          41       38.33      1.72   PROBLEMA
32. design0_n16.pdb         41       48.11      2.24   PROBLEMA
```