Trabajo de Final de Máster - Fase 2: Desarrollo de Trabajo

Alumno: Juan Vigueras Díaz **Tutor**: Jorge Valencia Delgadillo

---

A continuación se exponen las tareas que se han completado durante las semanas correspondientes a la fase 2. Cada modificación realizada respecto a lo expuesto en el plan de trabajo se indica mediante la anteposición de la palabra "**Modificación**" antes de la tarea correspondiente. Las referencias a páginas web se marcan mediante letras y las referencias a bibliografía se marcan mediante números.

Introducción

El bloque de trabajo principal de esta fase ha consistido en acabar de definir las funciones de ajuste/validación para los modelos de inteligencia artificial, realizar entrenamientos sucesivos para evaluar la viabilidad de dichos modelos para predecir bioactividad en función de grafos/descriptores moleculares, analizar los resultados de los experimentos y crear un repositorio donde almacenar tanto el código fuente como los modelos ajustados y el resumen de su rendimiento.

Metodología de Desarrollo de Trabajo

Pese a que no se mencionó en la Fase 1, la metodología general de trabajo que se ha seguido para este proyecto es la basada en el conocido modelo CRISP-DM (del inglés Cross Industry Standard Process for Data Mining), el cual se ilustra a continuación en la Figura 1 y sirve como un estándar para el desarrollo de una solución integral en el marco de un proyecto de ciencia y/o ingeniería de datos [1].

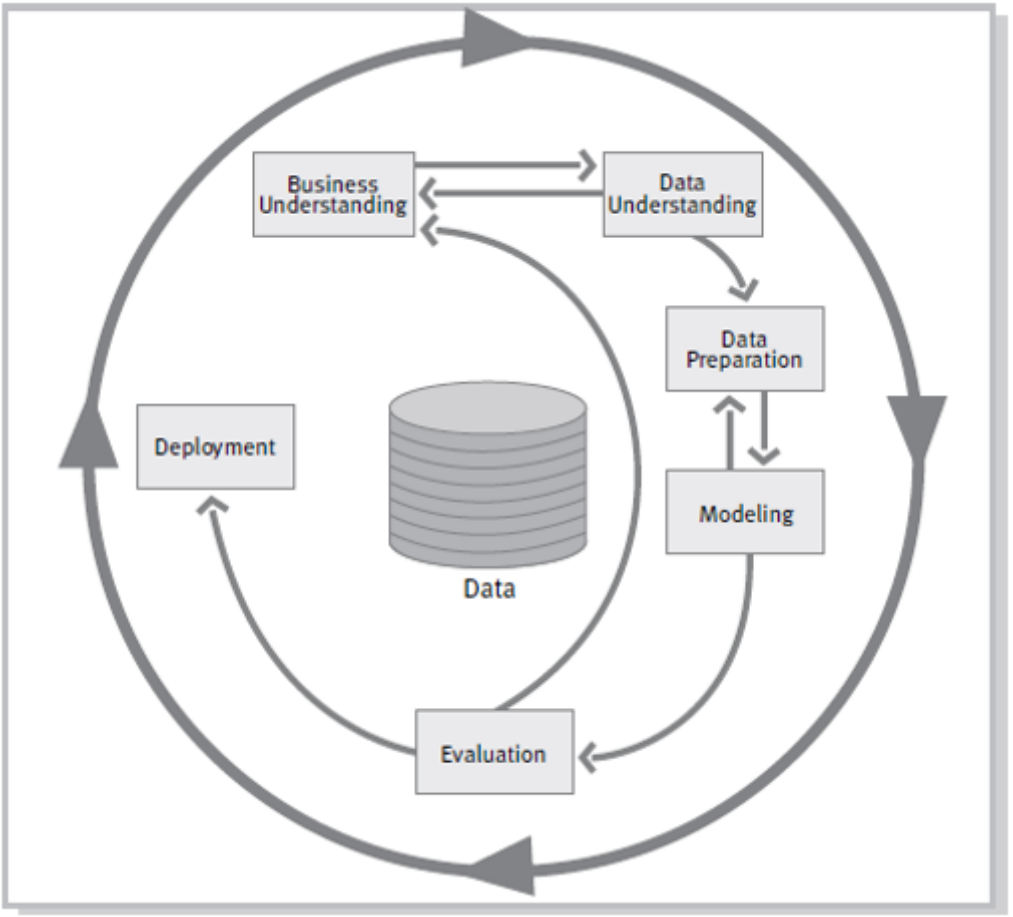


Figura 1. Fases del modelo de referencia CRISP-DM (Fuente: CRISP-DM 1.0 [1])

Si bien en la Fase 1 del proyecto las tareas realizadas giraron en torno a los tres primeros pasos de este modelo (comprensión del problema, comprensión de los datos y preparación de los datos), esta Fase 2 ha puesto foco en

el tercer, cuarto y quinto paso (preparación de los datos, modelación y evaluación). Con intención de llegar a un entendimiento sobre las limitaciones de los modelos en el escenario de estudio planteado y frente a la falta de tiempo para desarrollar el último paso, se ha dejado de lado el despliegue de una aplicación que permitiera evaluar nuevos datos de análisis sobre los modelos que presentaran mejor rendimiento.

Modificación del Cronograma

Así pues y teniendo esto en cuenta, el cronograma de tareas a partir de esta fase de trabajo ha quedado modificado de la siguiente manera:

Fase 2:

Como se puede observar en la Figura 2, se expande la durabilidad de la ejecución de entrenamientos y se elimina el desarrollo de aplicación para el despliegue de los modelos en un hipotético escenario de producción. El motivo principal de la modificación de tareas en esta fase ha sido, tal y como se presentará a continuación en el desarrollo de los experimentos, la necesidad de ajustar los datos de entrenamiento y realizar así diversas iteraciones para analizar la capacidad de aprendizaje de los modelos en función de la varianza de la variable respuesta, lo cual fue un factor que no se tuvo en cuenta en la fase anterior de desarrollo puesto que únicamente se consideró la varianza de las variables regresoras.

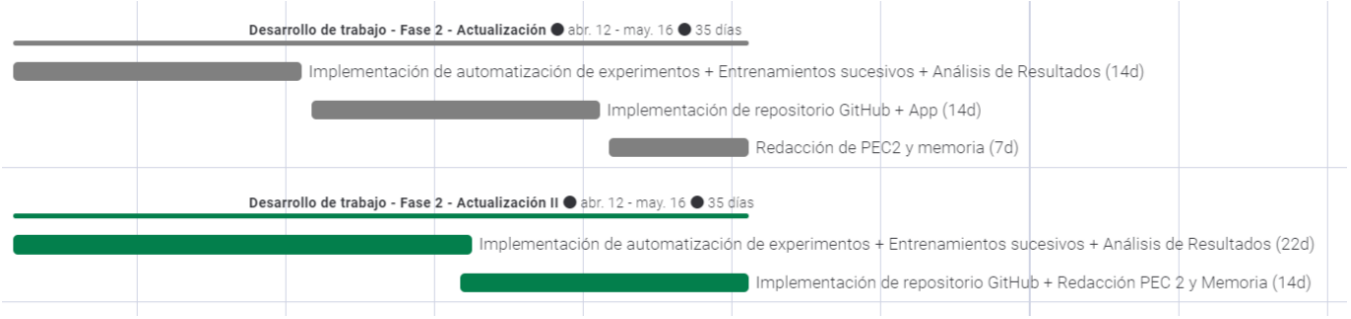


Figura 2. Actualización de plan de trabajo para la Fase 2 del proyecto

Últimas Fases (Quedan inalteradas):

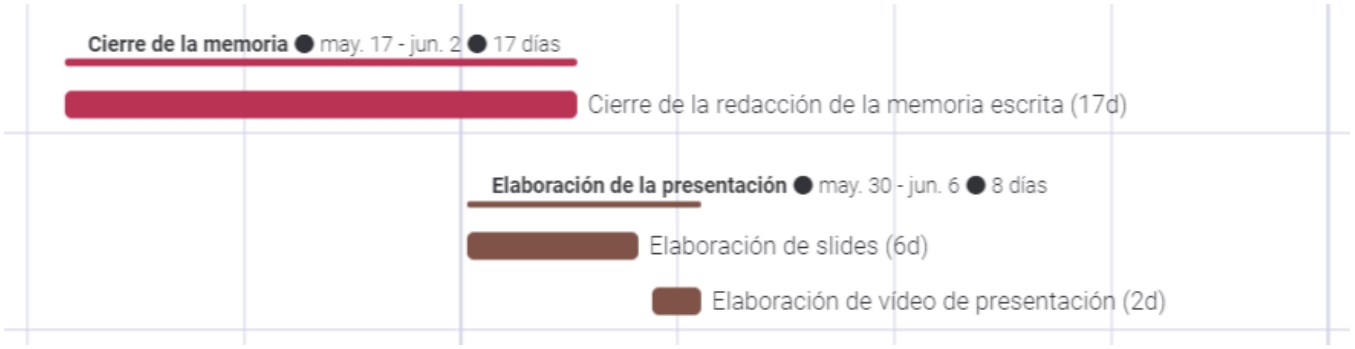


Figura 3. Plan de trabajo para la Fase final del proyecto

Tareas Realizadas y Análisis de Resultados

Funciones de Ajuste de Modelos

Las funciones que se utilizan para entrenar cada tipo de modelo de inteligencia artificial (deep learning / machine learning) se construyen por separado, puesto que el mecanismo de entrenamiento es distinto en cuanto a que para un modelo de deep learning es necesario definir los bucles de entrenamiento y validación (tal y como se ha detallado en la Fase 1 del Desarrollo de Trabajo) mientras que un modelo de machine learning contiene los métodos de ajuste definidos en la propia librería. El detalle de las funciones implementadas en el *notebook* 'MODELS AND TRAIN_TEST LOOPS.ipynb' se muestra a continuación en la Tabla 1.

Función	**Descripción**
fit_DL(...)	Misma función "fit()" descrita en la Fase 1 del Desarrollo del Trabajo para entrenar y validar

	modelos de Deep Learning según la arquitectura propuesta.
fit_ML(...)	Toma como parámetros: - El nombre que se quiere asociar al modelo creado, en formato “cadena de texto”. - Los conjuntos de datos de entrenamiento y *test*. Las variables de entrada y la variable de salida se indican por separado, con lo que son necesarios un total de cuatro parámetros en formato ‘Data Frame’ de la librería Pandas. - Un diccionario que a su vez contenga tres diccionarios con los híper-parámetros de cada tipo de algoritmo de Machine Learning. El nombre de cada híper-parámetro ha de estar indicado según la librería correspondiente a la clase de cada algoritmo. - La ruta donde almacenar los resultados, en formato “cadena de texto”. - Un valor *booleano* que indica si guardar los modelos generados y sus respectivos resultados (i.e., métricas de evaluación). Esta función utiliza los métodos “fit” y “predict” para ajustar tres modelos de distintos algoritmos de Machine Learning: X-Gradient Boost (a), Random Forest (b) y Support Vector Machine (c). En este caso, la métrica de evaluación (RMSE) también se importa directamente como un método desde la librería Scikit-Learn, con lo que, pese a que podría definirse como una función a partir de su expresión matemática, no es necesario implementarla desde cero. En caso de indicarlo por parámetro, la función guarda los modelos ajustados y crea un archivo en formato de texto plano en el que se indican los parámetros de cada algoritmo y los resultados de la métrica de evaluación sobre el conjunto de datos de *test*. Por comodidad según cada librería, los modelos correspondientes al algoritmo X-Gradient Boost se guardan en formato ‘.JSON’ y los correspondientes a Random Forest y Support Vector Machine se guardan en formato ‘.JOBLIB’.

Tabla 1. Nombre y descripción de las funciones creadas para el entrenamiento de modelos

EVALUACIÓN MODELOS MACHINE LEARNING

Dos aspectos importantes a tener en cuenta en el entrenamiento de modelos de machine learning son:

- Al tratarse de una tarea de regresión, la evaluación de los modelos se ve limitada por la valoración de la métrica RMSE que, aunque tiene la propiedad útil de estar en las mismas unidades que la variable de respuesta y un valor más bajo indica un mejor ajuste, no indica en sí misma si un modelo se puede considerar *bueno* o no. Por este motivo, surge la necesidad de comparar los valores de RMSE obtenidos con la varianza de la variable respuesta en cada uno de los experimentos para conjeturar así si el ajuste obtenido es aceptable o no.
- Antes de aplicar la función de ajuste, se normalizan los datos de todas las variables de entrenamiento. Este paso es comúnmente conocido y aplicado, necesario para mantener las relaciones entre dichas variables al mismo tiempo que se evitan problemas de interpretación del algoritmo derivados de las diferencias entre las escalas de cada columna de datos [2]. Para realizar el escalado, se emplea la clase ‘MinMaxScaler’ de la librería Scikit-Learn (d) que transforma las características escalándolas a un rango dado, por defecto entre 0 y 1. La transformación viene dada por la expresión que se muestra a continuación en la Figura 4.

```
X_std = (X - X.min(axis=0)) / (X.max(axis=0) - X.min(axis=0))
X_scaled = X_std * (max - min) + min
```

Figura 4. Transformación ‘MinMaxScaler’ de la librería Scikit-Learn, fuente: documentación Scikit-Learn (d).

Pruebas de Entrenamiento I

En primer lugar y con el objetivo de comprobar que el código no genere ningún tipo de error, se realiza una prueba de entrenamiento con la configuración que se muestra a continuación, en la Tabla 2.

Objetivo	Indicador	Híper-parámetros
E_COLI	MIC	Configuración por defecto en cada algoritmo. Consultar (a), (b), (c).

Tabla 2. Configuración del experimento para la depuración del proceso de ajuste de los modelos de machine learning

El tiempo de ejecución es de 1 minuto y 6 segundos. El código no genera ningún error y la prueba resulta satisfactoria, los resultados obtenidos se muestran a continuación en la Tabla 3.

Algoritmo	RMSE
X-Gradient Boost	2.2e13
Random Forest	2.9e13
Support Vector Machine	2.7e13

Tabla 3. Resultados del experimento correspondiente a la Tabla 2.

Como se puede observar, los tres algoritmos presentan un RMSE similar y con un orden de magnitud aparentemente alto. Para determinar hasta qué punto estos valores son ciertamente altos o no, es necesario compararlos con los valores que toma la variable respuesta, los cuales se muestran a continuación, en la Tabla 4, en un resumen estadístico.

Objetivo	E_COLI
Indicador	MIC
Número de registros	19 367
Media	2.9e5
Desviación estándar	4.6e6
Coeficiente de Variación	15.9
Cuartil 25%	5.8e-3
Cuartil 50%	5.1e-2
Cuartil 75%	3.1e-1
Valor Mínimo	0
Valor Máximo	1.6e+8

Tabla 4. Resumen estadístico de la variable respuesta del experimento correspondiente a la Tabla 2.

Se puede ver cómo, pese a que la media es del orden de 1e5 y la desviación estándar del orden de 1e6, el 75% de los datos se encuentran incluidos antes del orden de magnitud 1e-1. El coeficiente de variación, definido como la relación entre la desviación estándar y la media, es mucho mayor que uno, lo cual indica que la media no es representativa de la distribución y que esta presenta una dispersión muy fuerte. Este hecho plantea la opción de reducir el rango de trabajo para entrenar los modelos en un entorno de valores pre-determinado, dada la incapacidad de modelar una característica con tanta varianza con el conjunto de variables regresoras de las que se dispone. Dado que en este trabajo no se contempla la opción de averiguar si dicha variabilidad puede ser producida, por ejemplo, por un error experimental, directamente se opta por reducir el conjunto de datos para trabajar con el conjunto contenido en el cuartil del 75% y considerar el resto de valores como valores atípicos.

Objetivo	Indicador	# Registros	Media	Desv. Est.	Coef. Var.	Valor Máx.

E_COLI	MIC	14 525	5.3e-2	7.2e-2	1.35	3.1e-1

Tabla 5. Resumen estadístico de la Tabla 4 sobre el conjunto de datos del cuartil del 75%.

Realizando este corte, tal y como se muestra anteriormente en la Tabla 5, se sacrifican un total de 4.842 registros a cambio de reducir el coeficiente de variación en más de 14 unidades. Tal y como se muestra a continuación en la Figura 4, el rango de la variable respuesta se reduce hasta en ocho órdenes de magnitud. Como un valor alto de MIC está relacionado con un bajo nivel de bioactividad, este resultado permite considerar los valores tan altos como valores atípicos y prescindibles para el modelo de predicción.

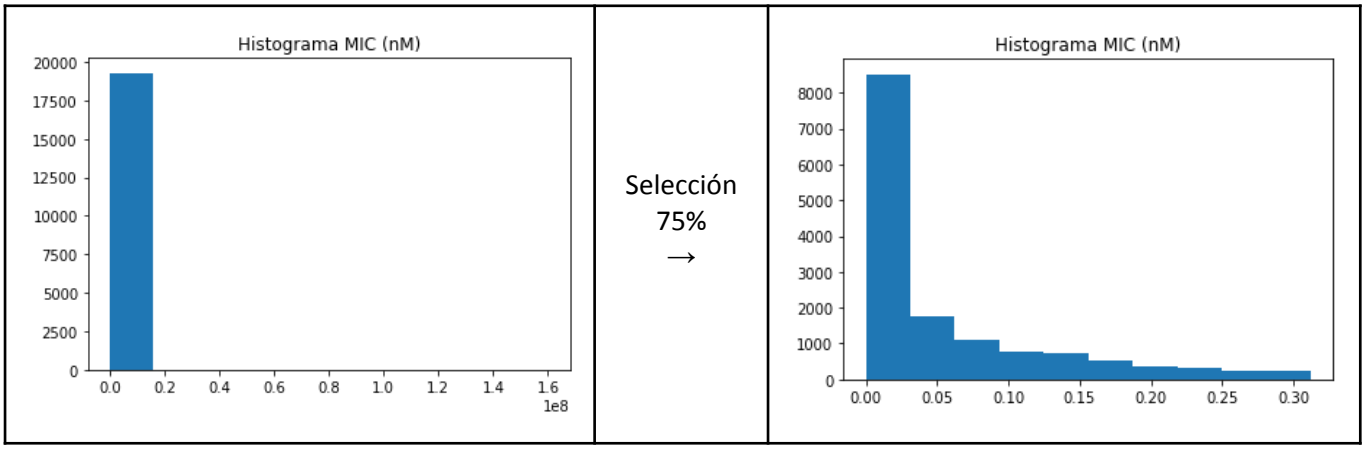


Figura 4. Histogramas del indicador MIC para el objetivo E_COLI con: izquierda, conjunto completo de datos y, derecha, selección del conjunto de datos contenidos en el cuartil 75%

Con esta nueva selección de datos realizada, se realiza una nueva prueba de entrenamiento con la misma configuración de la Tabla 2. El tiempo de ejecución es de 23.5 segundos, y los resultados obtenidos se muestran a continuación en la Tabla 6.

Algoritmo	**RMSE**	**RMSE Norm. Máx**	**RMSE Norm. Media**
X-Gradient Boost	5.1e-3	1.6%	9.6%
Random Forest	7.9e-3	2.5%	14.8%
Support Vector Machine	5.4e-3	1.7%	10.0%

Tabla 6. Resultados del experimento correspondiente a la configuración Tabla 2 con la selección del conjunto de datos contenidos en el cuartil 75%.

Para una mejor interpretación de los resultados de la métrica RMSE en este caso, se muestran en la Tabla 6 los valores de RMSE normalizados según los valores de la media y el máximo de la variable respuesta en el conjunto de datos de *test*. Como se puede observar, en este caso se obtienen resultados plausibles dado que el orden de magnitud de los tres valores de RMSE son dos órdenes de magnitud menor que el rango de la variable respuesta, dando así un % de error según el RMSE normalizado según el valor máximo entre el 1 y el 3% y un RMSE normalizado según la media entre el 9 y el 15%.

Para tener una mejor idea de por qué estos resultados se pueden considerar como buenos modelos base de referencia, es necesario comprender la naturaleza de las expresiones matemáticas de la desviación estándar y la métrica de evaluación. En ausencia de un conjunto de variables predictoras, la media de la variable respuesta puede ser considerada como una estimación simple de todas las predicciones, lo cual hace que la desviación estándar sea una medida del *error estandarizado* que se estaría cometiendo al hacer esta consideración [3]. Cualquier modelo debería tratar de superar su nivel de error a la desviación estándar siempre y cuando su error se mida en las mismas unidades: es por este motivo que el RMSE es una métrica atractiva para realizar esta comparación. Como en este caso se consigue una reducción de un orden de magnitud, el resultado sugiere que los modelos están reduciendo la aleatoriedad de la predicción en contra de considerar la aproximación naíf del promedio de la variable respuesta como resultado de todas las predicciones.

Re-Selección de los Datos de Entrenamiento

El siguiente paso es, pues, aplicar un procedimiento análogo de cribado de registros basado en la variable respuesta para el resto de duplas *target* - indicador y comprobar las nuevas distribuciones en cada caso. Es necesario comprobar si tiene sentido aplicar el cribado del 75% para todos los casos. Se muestra a continuación, en la Tabla 7, una extensión de la Tabla 4 con el resumen estadístico de todas las duplas para la variable respuesta.

Objetivo	E_C	E_C	E_C_P	P_A	P_A	P_A_P	S_A	S_A	S_A_P
Indicador	MIC	IC50	IC50	MIC	IC50	IC50	MIC	IC50	IC50
Núm. de registros	19 367	724	272	11 997	367	311	29 975	843	178
Media	2.9e5	7.1e5	2.9e7	1.2e5	2.6e4	9.8e3	5.9e5	1.2e6	2.0e4
Desv. estándar	4.6e6	1.3e7	1.3e8	2.7e6	6.4e4	1.1e4	4.4e7	1.8e7	9.4e4
Coef. de Variación	15.9	17.8	4.4	22.0	2.5	1.1	74.6	15.4	4.6
Cuartil 25%	5.8e-3	6.4e-1	1.9e1	1.1e-2	9.3e-2	4.0e3	2.0e-3	9.3e-3	3.1e1
Cuartil 50%	5.1e-2	3.7e3	3.2e2	6.3e-2	3.5e3	6.3e3	1.9e-2	4.1e0	2.8e2
Cuartil 75%	3.1e-1	1.9e4	1.0e4	3.4e-1	2.0e4	1.1e4	1.8e-1	6.5e3	6.8e3
Valor Mínimo	0	8.3e-6	0	0	5.5e-5	3.4e1	0	1.6e-6	1.2
Valor Máximo	1.6e8	3.3e8	1.0e9	1.6e8	4.6e5	6.2e4	6.9e9	5.0e8	8.4e5

Tabla 7. Resumen estadístico de la variable respuesta para todas las duplas target - indicador.

De la Tabla 7 se pueden obtener las siguientes dos conclusiones:

- Los coeficientes de variación son, en general, considerablemente más bajos en los casos en los que se estudia una proteína de la bacteria.
- Para el indicador MIC, el valor del cuartil del 75% es del orden de 1e-1 para todos los objetivos, mientras que para el indicador IC50 el cuartil del 75% varía en orden desde 1e3 a 1e4, independientemente de si se analiza la bacteria en sí misma o una proteína de esta.

Resulta evidente deducir que en el caso del indicador MIC es conveniente aplicar el mismo criterio de cribado que en el caso analizado en el apartado anterior, puesto que las características que presenta cada caso son similares.

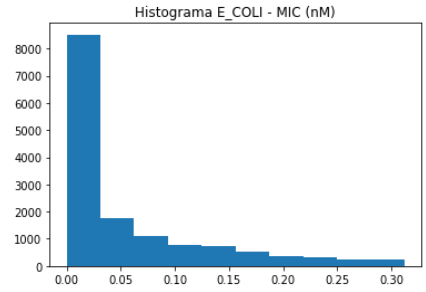
En cambio, en el caso del indicador IC50 no es directa la deducción dado que, por un lado, todos los casos presentan órdenes de magnitud similares para los datos contenidos en el cuartil del 75% (1e3 - 1e4), pero en el caso de los *targets* de proteínas y del *target* P_A el coeficiente de variación es del orden de 1e0, lo cual indica que en esos casos los datos no son tan dispersos como en el resto. Con tal de realizar un procedimiento unificado para todas las duplas, se aplica igualmente el criterio de cribado del 75% en todos los casos, ya que la única desventaja es que, en los casos mencionados en los que el coeficiente de variación es menor, se eliminarán algunos valores que no deberían considerarse atípicos.

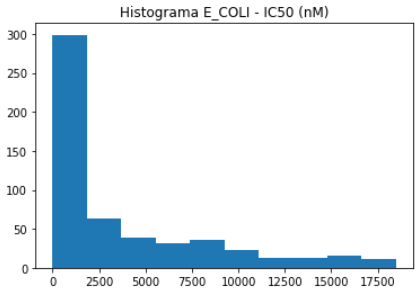
A continuación se muestra, en la Tabla 8, una extensión de la Tabla 5, una versión reducida del resumen estadístico tras aplicar los cribados pertinentes a cada variable respuesta.

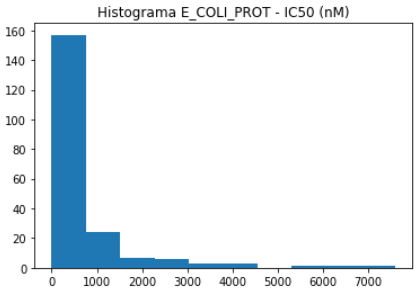
Objetivo	Indicador	# Registros	Media	Desv. Est.	Coef. Var.	Valor Máx.
E_C	MIC	14 525	5.3e-2	7.2e-2	1.35	3.1e-1
E_C	IC50	543	3.5e3	4.7e3	1.33	1.9e4
E_C_P	IC50	283	5.8e2	1.2e3	1.97	7.6e3
P_A	MIC	8 997	6.3e-2	8.1e-2	1.27	3.4e-1
P_A	IC50	275	3.6e3	5.3e3	1.45	2.0e4
P_A_P	IC50	233	5.2e3	2.6e3	0.50	1.1e4
S_A	MIC	22 481	2.6e-2	4.0e-2	1.53	1.8e-1
S_A	IC50	631	4.6e2	1.2e4	2.50	6.4e3
S_A_P	IC50	133	9.2e2	1.8e3	1.88	6.5e3

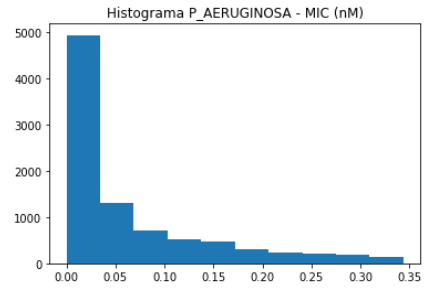
Tabla 8. Resumen estadístico de la Tabla 7 sobre el conjunto de datos del cuartil del 75% para cada variable respuesta.

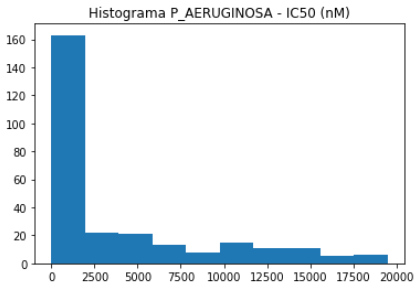
Como se puede apreciar, en todos los casos se produce un ajuste del coeficiente de variación igual que en el caso analizado en el apartado anterior. Se muestran a continuación, en la Figura 5, los histogramas de las distribuciones para cada una de las variables.

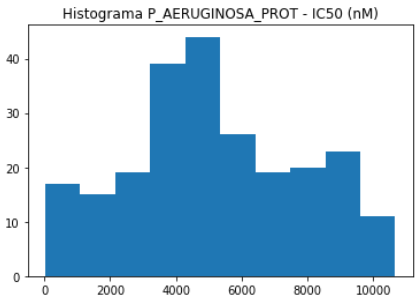
<table>
<tr>
<td>E_COLI MIC
 

SubFig. 1</td>
<td>E_COLI IC50
 

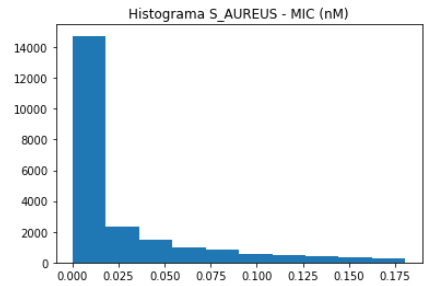
SubFig. 2</td>
<td>E_COLI_PROT IC50
 

SubFig. 3</td>
</tr>
<tr>
<td>P_AEURUGINOSA MIC
 

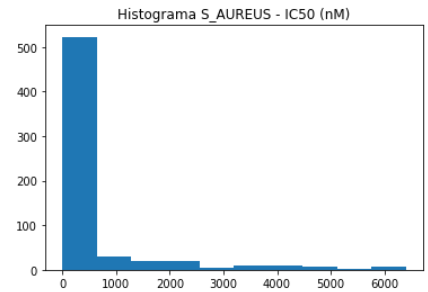
SubFig. 4</td>
<td>P_AERUGINOSA IC50
 

SubFig. 5</td>
<td>P_AERUGINOSA_PROT IC50
 

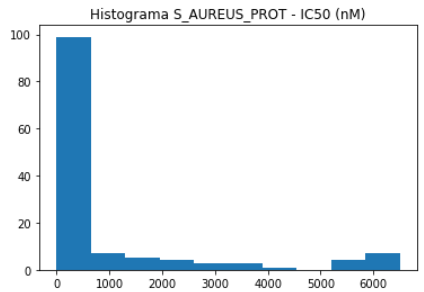
SubFig. 6</td>
</tr>
<tr>
<td>S_AUREUS MIC</td>
<td>S_AUREUS IC50</td>
<td>S_AUREUS_PROT IC50</td>
</tr>
</table>



SubFig. 7



SubFig. 8



SubFig. 9

Figura 5. Histogramas de las duplas target - indicador con la selección del conjunto de datos contenidos en el cuartil 75%

Se puede apreciar cómo únicamente en el caso de P. Aeruginosa Prot - IC50, en el que el coeficiente de variación es menor que 1, la distribución del histograma podría ajustarse a una forma de campana, mientras que en el resto de casos, en los que el coeficiente de variación es mayor que 1, las distribuciones presentan una cola a la derecha. Una comparación estadísticamente más apropiada entre la desviación estándar y la métrica de evaluación RMSE [3] implicaría filtrar todas las distribuciones de datos, eliminando registros hasta conseguir esta forma de campana en cada caso, pero dados los buenos resultados obtenidos en la prueba de entrenamiento tras el cribado del 75% se ha decidido mantener este criterio ya que no existe a priori un método que asegure si existe un corte a partir del cual se obtenga dicha forma en la distribución.

Pruebas de Entrenamiento II

Con los datos obtenidos en la selección del apartado anterior, se vuelve a realizar un ajuste de cada algoritmo con la configuración de híper-parámetros por defecto. y los resultados se muestran a continuación, en la Tabla 9.

Target - Indicador	Algoritmo	RMSE	RMSE Norm. Máx	RMSE Norm. Media
E_C - MIC	XGB	5.1e-3	1.7%	9.5%
	RF	7.9e-3	2.6%	14.7%
	SVM	5.4e-3	1.7%	10.0%
E_C - IC50	XGB	2.4e7	1.3e5%	6.8e5%
	RF	2.5e7	1.4e5%	7.1e5%
	SVM	2.9e7	1.6e5%	8.3e5%
E_C_P - IC50	XGB	3.8e6	5.0e4%	6.5e5%
	RF	4.5e6	5.9e4%	7.8e5%
	SVM	3.7e6	4.9e5%	6.4e5%
P_A - MIC	XGB	6.1e-3	1.8%	9.6%
	RF	7.5e-3	2.2%	11.8%
	SVM	6.5e-3	1.9%	10.3%
P_A - IC50	XGB	3.2e7	1.7e5%	8.9e5%
	RF	5.9e7	3.0e5%	1.6e6%
	SVM	2.2e7	1.1e5%	6.0e5%

P_A_P - IC50	XGB	9.9e6	9.2e4%	1.9e5%
	RF	7.1e6	6.7e4%	1.4e5%
	SVM	7.4e6	7.0e4%	1.4e5%
S_A - MIC	XGB	1.9e-3	1.1%	7.3%
	RF	5.6e-3	3.1%	21.6%
	SVM	2.5e-3	1.4%	9.7%
S_A - IC50	XGB	1.3e6	2.0e4%	2.7e5%
	RF	1.4e6	2.2e4%	3.0e5%
	SVM	1.6e6	2.5e4%	3.5e5%
S_A_P - IC50	XGB	3.7e6	5.7e4%	4.0e5%
	RF	4.4e6	6.8e4%	4.8e5%
	SVM	3.7e6	5.7e4%	4.0e5%

Tabla 9. Resultados del experimento correspondiente a la configuración por defecto de los algoritmos con la selección del conjunto de datos contenidos en el cuartil 75% para cada dupla target - indicador

De la anterior tabla se puede observar que:

- Todos los resultados correspondientes al indicador MIC convergen a unos valores de métrica de evaluación plausibles y equivalentes al de la Prueba de Entrenamiento I.
- Pese a que no existe una diferencia significativa ya que estos valores son próximos entre sí y del mismo orden de magnitud, el algoritmo que presenta mejor resultado para cada bacteria de estudio es X-Gradient Boost, y la bacteria para la que se obtiene un resultado es *S. Aureus*.
- Todos los resultados correspondientes al indicador IC50 divergen, incluso en el caso de la proteína de la bacteria *P. Aeruginosa*, que presentaba una distribución de la variable respuesta ajustada a una campana de Gauss. Esto indica que la aproximación tomada en cuanto al cribado de datos para este indicador no es correcta, pese a tener valores de coeficiente de variación equivalentes en orden de magnitud al caso del indicador MIC.

De esto se concluye que el indicador MIC establece un mejor escenario de partida para la tarea de predicción de bioactividad dadas las variables regresoras escogidas en el pre-tratamiento de datos.

EVALUACIÓN MODELOS DEEP LEARNING

Función de pérdida y Representación de las curvas de entrenamiento

Mientras que la evaluación de modelos de machine learning en la tarea de regresión, tal y como se ha visto anteriormente, se ve limitada por únicamente la interpretación que se hace de la métrica de evaluación sobre el conjunto de datos de *test*, durante el ajuste de un modelo de deep learning se puede llevar a cabo un registro de la evaluación del rendimiento del modelo durante todo el proceso de aprendizaje. Para ello, se representan las llamadas *curvas de aprendizaje* representando, para cada época o ciclo de ajuste, la función de pérdida y la métrica de evaluación. En tanto que la segunda es la función RMSE implementada tal y como se detalló en la Fase 1 de Desarrollo de Trabajo, la función de pérdida es la que se pretende minimizar para conseguir un modelo con cierta capacidad de predicción. En este caso, la función escogida es el error cuadrático medio o MSE, que se define como:

$$MSE = \frac{1}{N}\sum_{1}^{N}(Predicted_i - Observed_i)^2 \quad (1)$$

Es decir, es el valor RMSE al cuadrado. Los motivos por los que se ha escogido esta función son:

- Pese a que no se encuentra en las mismas unidades que la variable respuesta, no es necesaria una interpretación cuantitativa de su valor como sí ocurre con la métrica de evaluación, sino que únicamente se analiza el comportamiento de su curva.
- Se encuentra implementada en el módulo *nn* de Pytorch, con lo que se puede aplicar directamente.

En cada entrenamiento se representan dos curvas para cada función, una curva correspondiente a la función de pérdida ó métrica de evaluación aplicada sobre el conjunto de datos de entrenamiento y otra curva correspondiente a las mismas funciones aplicadas sobre el conjunto de datos de validación. En este caso no se evalúa el conjunto de datos de test, que únicamente se pasa por el modelo al final del entrenamiento. A partir de la forma que adopten estas curvas, se puede inferir si el modelo está realizando un sobreajuste de los datos, un ajuste relativamente preciso o un ajuste adecuado con mayor o menor capacidad de generalización del comportamiento de la variable respuesta en función de las variables regresoras. Dado que PyTorch no dispone de un módulo que permita registrar y representar estas curvas, la herramienta que se utiliza para realizar esta tarea es el panel Tensor Board de la biblioteca de código abierto Tensor Flow [e], desarrollada por Google para el desarrollo y entrenamiento de redes neuronales [4].

Pruebas de Entrenamiento I

En primer lugar, se realizan dos entrenamientos con únicamente tres épocas con dos objetivos (uno por experimento): por un lado, comprobar que no se producen errores de código en cuanto a la arquitectura del modelo y su procesamiento de los datos(esta prueba no pretende comprobar la capacidad de aprendizaje del modelo) y, por otro lado, comprobar que un modelo puede ser guardado correctamente y que el experimento correspondiente al modelo generado puede ser correctamente representado mediante la herramienta Tensorboard. La configuración del experimento se muestra a continuación, en la Tabla 10.

Objetivo	E_COLI
Indicador	MIC
Optimizador	Adam (valor de decaimiento = 0)
Planificador	No
Dropout	0
Otras técnicas de regularización	No
Tasa de aprendizaje (learning rate)	0.001
Tamaño del lote (batch size)	32

Tabla 10. Configuración del experimento para la depuración del proceso de ajuste del modelo de deep learning

El valor de la tasa de aprendizaje o *learning rate*, que se puede describir cualitativamente como el tamaño del paso en cada iteración del entrenamiento en el que el algoritmo de optimización trata de alcanzar el mínimo de la función de pérdida, suele variar en problemas típicos de regresión entre 0.001 y 0.01 [6]. Por este motivo, se escoge un valor de 0.001, ya que un paso menor siempre es preferible para no exceder el mínimo de la función de pérdida en un paso demasiado abrupto, y se mantiene invariable durante todo el entrenamiento, descartando el uso de planificador (o *scheduler*) y planteando así un escenario de *learning rate* constante como modelo de referencia para la evaluación comparativa del rendimiento de la red neuronal.

En lo que respecta al optimizador, cuya función es actualizar los valores de los parámetros de la red neuronal (pesos y sesgos) a partir del método del descenso de gradiente para reducir el error dado por la función de pérdida [5], se escoge ‘Adam’ (Adaptive Moment Estimation) por ser comúnmente usado en soluciones Deep Learning y por presentar buenos resultados en problemas de tipo convolucional [7], [8]. Dado que el valor de

decaimiento (o *weight decay*) del optimizador es una penalización aplicada a la función de actualización de los parámetros de la red neuronal y por tanto se considera técnica de regularización, se establece con valor 0.

Al tratarse de una prueba de entrenamiento, evidentemente no se aplican técnicas de regularización de sobreajuste como 'dropout' (que se establece con valor 0) o 'data augmentation' (cuya implementación no se ha considerado en el desarrollo del proyecto) al no tener conocimiento sobre si existe tal sobreajuste o no[9].

Por último, y pese a los resultados obtenidos con los algoritmos de Machine Learning, esta prueba de entrenamiento se realiza con todo el conjunto de datos de entrenamiento del objetivo seleccionado con el fin de comprobar si existe una divergencia en la función de pérdida o en la métrica de evaluación, tal y como ocurría anteriormente con el segundo caso. El valor del tamaño del lote o 'batch size' se escoge como 32 al ser un valor típicamente escogido ya que coincide con el tamaño de la normalización que se aplica en la arquitectura de la red neuronal, teniendo como objetivo así el tratar de optimizar así el uso de los recursos informáticos de *hardware* durante el flujo de datos. Se escoge 32 y no otro valor presente en otras capas normalización (16, 64) de forma arbitraria y sin considerar posibles efectos de la influencia del tamaño de *batch* como, por ejemplo, una peor capacidad de aprendizaje con un tamaño mayor de empaquetamiento. Tampoco se considera un proceso estocástico, i.e., con tamaño del *batch* igual a 1, dada la complejidad del problema y

El primer experimento se ejecuta correctamente, sin ningún tipo de error de compilación. Con esto, se ejecuta una réplica de este experimento, guardando el modelo generado y registrando las curvas de entrenamiento. El tiempo de ejecución de esta segunda prueba es de 56 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 6.

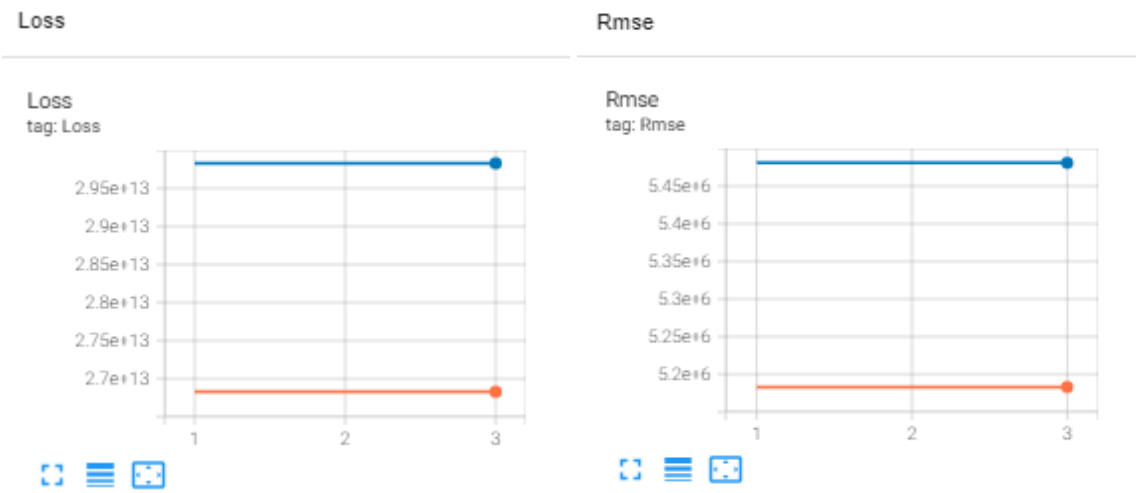


Figura 6. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color naranja, y validación, en color azul, sobre el experimento de prueba con 3 épocas y la configuración correspondiente a la Tabla 10

El resultado de la segunda prueba de control se produce sin generar ningún tipo de error. Aun así, tal y como se puede observar en la Figura 6, el orden de magnitud de la métrica de evaluación es de 1e6. Este valor, pese a ser siete órdenes de magnitud que el de la primera prueba ejecutada con algoritmos de Machine Learning (Tabla 3), es del mismo orden de magnitud que la desviación estándar del conjunto de datos de entrenamiento al completo (Tabla 4). Esto quiere decir que, en la fase inicial de entrenamiento, el modelo ajusta la variable respuesta con una aproximación equivalente a considerar el modelo naíf de todas las predicciones iguales a la media de la distribución de datos de entrenamiento.

Con tal de determinar si el modelo es capaz de realizar un cierto aprendizaje y superar esta aproximación, se ha de realizar un experimento con la misma configuración que en el caso anterior pero con un número de épocas mayor. Se escoge arbitrariamente el valor de 200 épocas, y se repite el experimento con la configuración correspondiente a la Tabla 10. El tiempo de ejecución de esta segunda prueba es de 33 minutos y 57 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 6.

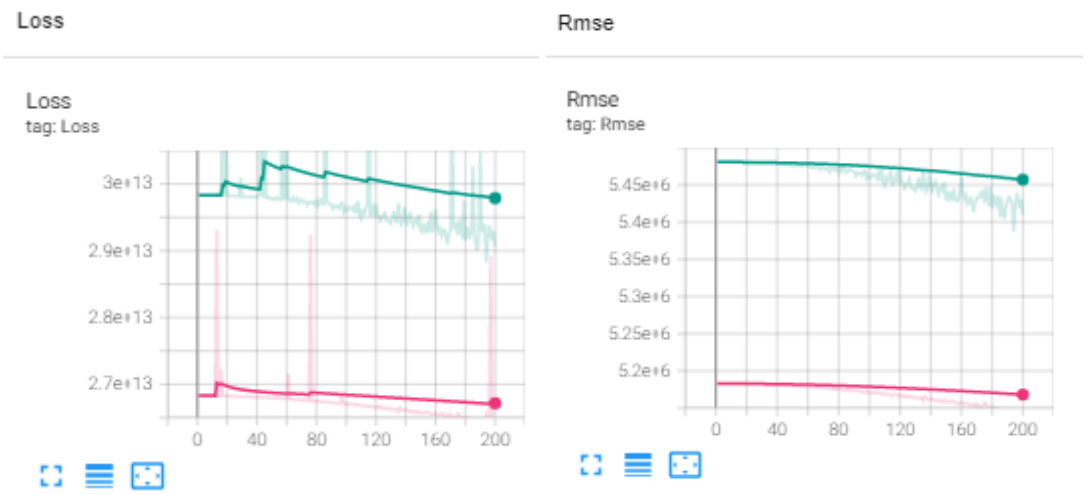


Figura 7. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color rosa, y validación, en color verde, sobre el experimento de prueba con 200 épocas y la configuración correspondiente a la Tabla 10. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	5.38e6
RMSE sobre conjunto de test	4.07e6

Tabla 11. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 7. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Como muestra la Tabla 11, se obtiene el mismo orden de magnitud para la métrica de evaluación que al inicio del entrenamiento.Tal y como se puede observar en la Figura 7, en las curvas sombreadas (correspondientes a las curvas originales de entrenamiento, no confundir con las curvas suavizadas) tanto de la función de pérdida como de la métrica de evaluación hay notables oscilaciones que dificultan el decaimiento general de la curva. Esto se traduce en que:

- El modelo es capaz de adquirir un aprendizaje lento pero progresivo durante tramos relativamente largos del entrenamiento. El motivo por el que las curvas de la métrica de evaluación RMSE son más suaves puede ser debido al efecto de la raíz cuadrada.
- Existen ciertos paquetes o conjuntos de datos que alteran dicho aprendizaje y lo hacen retroceder. Esto puede ser indicativo de la presencia de valores atípicos en el conjunto de datos de entrenamiento.

Con esto, se puede considerar que, en general, existe una tendencia al aprendizaje. Aunque en bibliografía especializada [10] no aparece una justificación de qué número aproximado de épocas utilizar en función de la complejidad del problema (entendido, en parte, como el número de parámetros del modelo a optimizar) y tampoco se han encontrado publicaciones que desarrollen por qué un valor del órden de 100 suele ser la solución comúnmente utilizada en multitud de escenarios, resulta evidente que el proceso de aprendizaje debería resolverse en este número aproximado de épocas. Así pues, con tal de tratar de *acelerar* el ajuste, se repite el experimento anterior con un número menor de épocas, 150, y con las modificaciones en la configuración que se indican a continuación, en la Tabla 12.

Objetivo	E_COLI
Indicador	MIC
Optimizador	Adam (valor de decaimiento = 0)
Planificador	No

Dropout	0
Otras técnicas de regularización	No
Tasa de aprendizaje (learning rate)	0.01
Tamaño del lote (batch size)	32

Tabla 12. Configuración del experimento para la prueba de aprendizaje del modelo de deep learning

El resultado del entrenamiento es:

Mejor RMSE de validación	5.44e6
RMSE sobre conjunto de test	4.13e6

Tabla 13. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 8. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

El tiempo de ejecución de esta segunda prueba es de 18 minutos y 46 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 8.

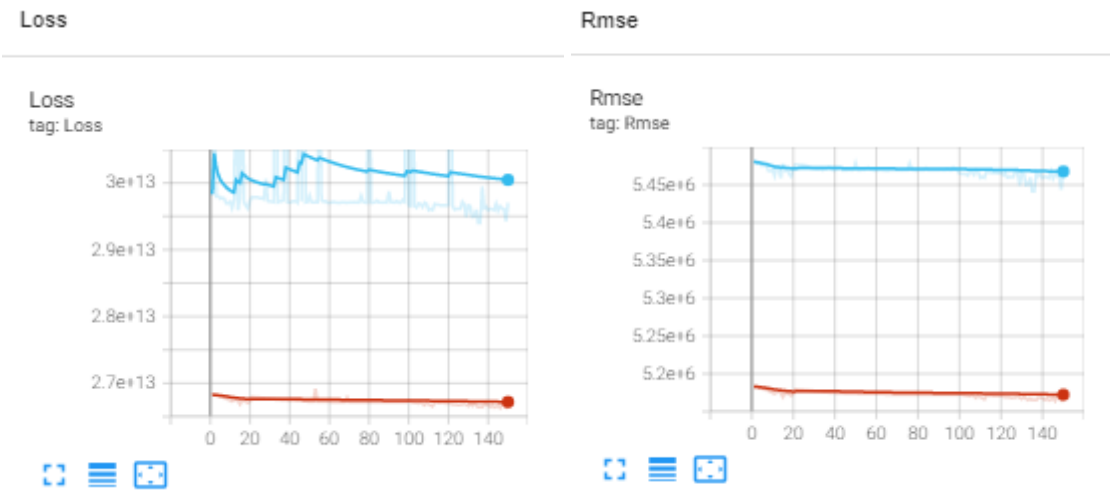


Figura 8. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color marrón, y validación, en color azul, sobre el experimento de prueba con 200 épocas y la configuración correspondiente a la Tabla 10. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

Como se puede observar en la Tabla 13, se obtienen exactamente los mismos resultados que en el experimento anterior. Por otro lado, si bien es cierto que, tal y como se muestra en la Figura 8, las oscilaciones en este caso están menos presentes que en las curvas de la Figura 7, no se aprecia que haya un decaimiento de ninguna de las funciones, lo cual remarca de nuevo la dificultad y/o ausencia de capacidad de aprendizaje.

Con tal de comprobar cuántas épocas se necesitan realmente para detectar dicho decaimiento, se realiza un nuevo experimento con la configuración correspondiente a la Tabla 10, pero aumentando un orden de magnitud el número de épocas del entrenamiento, definiéndolo en 1.000.

El tiempo de ejecución de esta tercera prueba es de 2 horas, 8 minutos y 22 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 9.

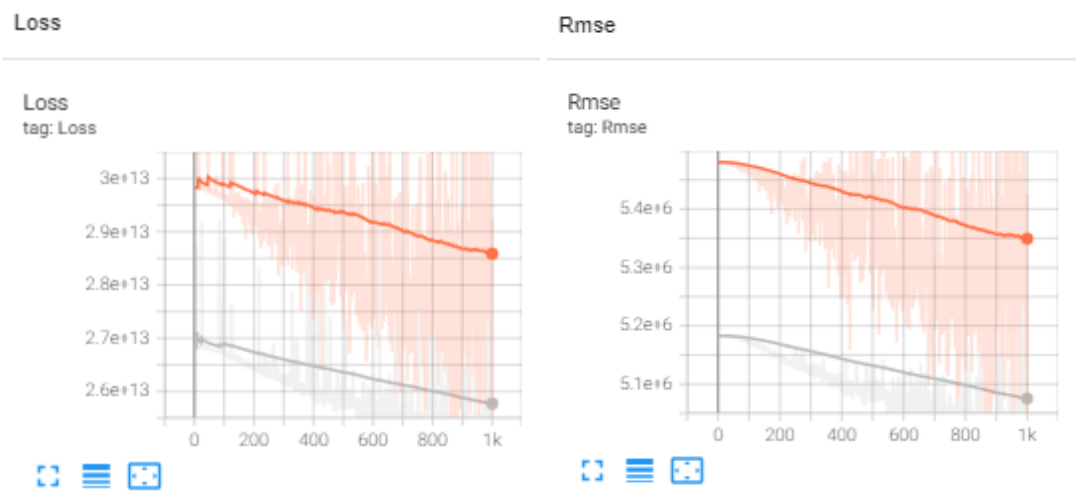


Figura 9. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color gris, y validación, en color naranja, sobre el experimento con 1.000 épocas y la configuración correspondiente a la Tabla 12. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	4.86e6
RMSE sobre conjunto de test	3.84e6

Tabla 14. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 8. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Si bien es cierto que, tal y como se aprecia en la Tabla 14, los resultados de la métrica de evaluación son del mismo orden de magnitud que los del experimento anterior (y, por tanto, muy cercanos también a la desviación estándar de la distribución de datos de entrenamiento), se pueden observar en las curvas suavizadas de la Figura 9 que el comportamiento general de las dos funciones es de decaimiento. Pese a esto, las fuertes oscilaciones que aparecen según avanza el entrenamiento son un claro indicativo, como ya se ha comentado, de la dificultad del modelo para encontrar el mínimo de la función de pérdida debido a varios posibles factores como (i) un *learning rate* demasiado alto, (ii) un tamaño de *batch* demasiado alto que impide abstraer correctamente las particularidades de cada elemento, o (iii) la presencia de un conjunto de valores anómalos en el *dataset*.

Dado que las opciones (i) y (ii) anteriormente planteadas pueden quedar intuitivamente descartadas dadas las justificaciones dadas durante el desarrollo de experimentos, esto plantea la siguiente aproximación al problema para solventar la dificultad de convergencia de la red neuronal: para poder realizar una comparación entre los tipos de modelo de inteligencia artificial expuestos en este trabajo, se hipotetiza con la posible influencia de la opción (iii) y se plantea la de trabajar con un conjunto de datos reducido, analizando el rendimiento de la arquitectura de Deep Learning propuesta para el indicador de bioactividad MIC en un nuevo conjunto de experimentos.

Hardware de entrenamiento

Para realizar los anteriores experimentos, se ha utilizado una unidad de GPU correspondiente a NVIDIA-SMI (NVIDIA System Management Interface) proporcionada por la herramienta Google Colab, cuyas características se detallan a continuación en la Figura 10.

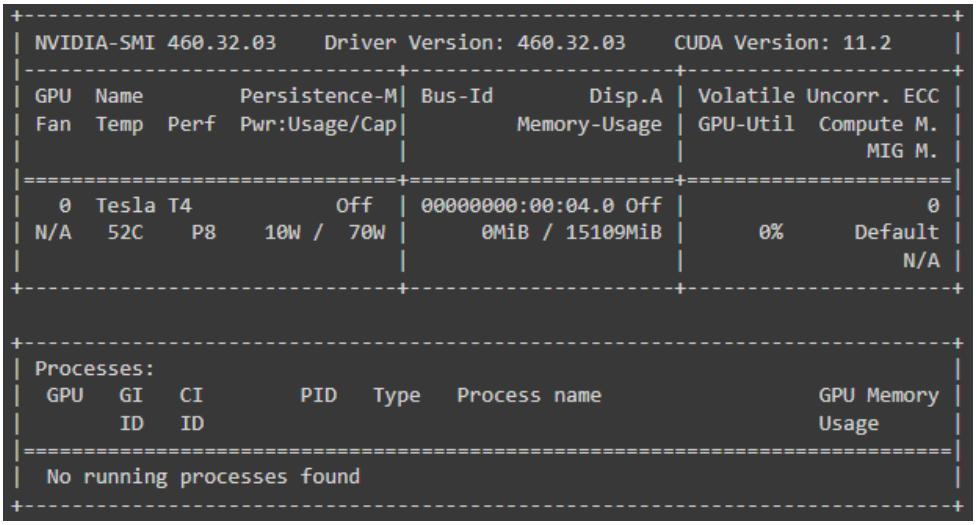


Figura 10. Características de la unidad de de GPU asociada por Google Colab

Tras los experimentos realizados en el apartado anterior, la GPU dejó de estar disponible debido a limitaciones de uso por usuario, y se tuvieron que realizar el resto de experimentos con la CPU estándar que ofrece la herramienta Google Colab. Este factor refuerza el hecho de que, dado que el tiempo resulta un factor limitante en el despliegue de experimentos en este estudio, se decida realizar una acotación del conjunto de datos antes de evaluar el rendimiento de la arquitectura de Deep Learning sobre todas las duplas *target - indicador*.

En los siguientes apartados, los experimentos se ejecutan mediante CPU, con lo que resulta inviable comparar los tiempos de ejecución de todos los entrenamientos.

Re-Ajuste de los Datos de Entrenamiento

Dado que los datos de entrenamiento, validación y test se encuentran almacenados en objetos de tipo 'Data Loader', resulta más sencillo volver a generar un nuevo conjunto de datos que seleccionar al vuelo, durante el entrenamiento, los valores contenidos en el cuartil del 75% de la distribución de la variable respuesta. Para ello, se ha modificado el Notebook 'DATA_PREPROCESSING.ipynb', utilizando parte del código existente para generar nuevos objetos 'Data Loader' únicamente para el indicador de bioactividad MIC y los tres A continuación se muestra, en la Tabla 15, la distribución del número de registros en los nuevos conjuntos de datos de grafos para los tres únicos objetivos disponibles para el indicador de bioactividad MIC.

	Indicador de Bioactividad: MIC		
Objetivos	**TRAIN**	**VALID**	**TEST**
E_COLI	12 944	1 618	1 619
P_AERUGINOSA	7 948	993	994
S_AUREUS	18 239	2 280	2 280

Tabla 15. Cantidad de registros en el conjunto de datos de grafos para cada uno de los split y target, seleccionando los datos contenidos en el cuartil del 75%

Como se puede observar, siguen siendo distribuciones con una cantidad más que asequible [10] de registros para realizar una tarea de regresión con un modelo de red neuronal. Además, el hecho de disponer de tres conjuntos que aproximadamente equidistan en la diferencia de registros que hay entre ellos (18 239 - 12 944 = 5 295 ≈ 4 996 = 12 944 - 7 948) puede resultar de utilidad para analizar la influencia del número del tamaño del conjunto de datos sobre el efecto de aprendizaje, siempre y cuando si se considera que todas las distribuciones contienen conjuntos de datos de entrenamiento y test cualitativamente equivalentes en cuanto a variabilidad (es decir, variedad en cuanto a las moléculas que se pueden encontrar en cada caso). En este caso, se hipotetiza que un mayor número de registros proporciona una mejor calidad de generalización [10] y, por tanto, un mejor resultado

en la métrica de evaluación sobre el conjunto de datos de *test*. Según esta hipótesis, la bacteria objetivo *S. Aureus* debería presentar mejores resultados que el resto.

Pruebas de Entrenamiento II

Con tal de comprobar el efecto de la reducción del conjunto de datos de grafos sobre la capacidad de aprendizaje de la red neuronal, se realizan tres experimentos, uno para cada una de las bacterias objetivo, con una configuración común y análoga a la de los experimentos de base anteriores. La configuración actualizada de los experimentos es equivalente a la de la Tabla 10 a excepción del número de épocas escogido, y se muestra a continuación, en la Tabla 16.

Optimizador	Adam (valor de decaimiento = 0)
Planificador	No
Dropout	0
Otras técnicas de regularización	No
Tasa de aprendizaje (learning rate)	0.001
Tamaño del lote (batch size)	32
Épocas	300

Tabla 16. Configuración de los experimentos sobre las tres bacterias - objetivo, con el conjunto de datos reducido según el cuartil del 75% de la variable respuesta MIC.

Se escoge un número de épocas igual a 300 para poder llegar a observar cualquier tipo de comportamiento de la red neuronal, ya que, por ejemplo, en ocasiones se produce efecto de sobreajuste tras un número lo suficientemente grande de épocas y resulta interesante ver si la arquitectura es capaz de bloquear este fenómeno o no. A continuación se analizan los resultados para cada uno de los experimentos, viendo el caso de la bacteria *E. Coli* por separado y los casos de las otras dos bacterias de forma conjunta.

Para el objetivo *E. Coli*, el tiempo de ejecución de esta prueba es de 40 minutos y 38 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 11.

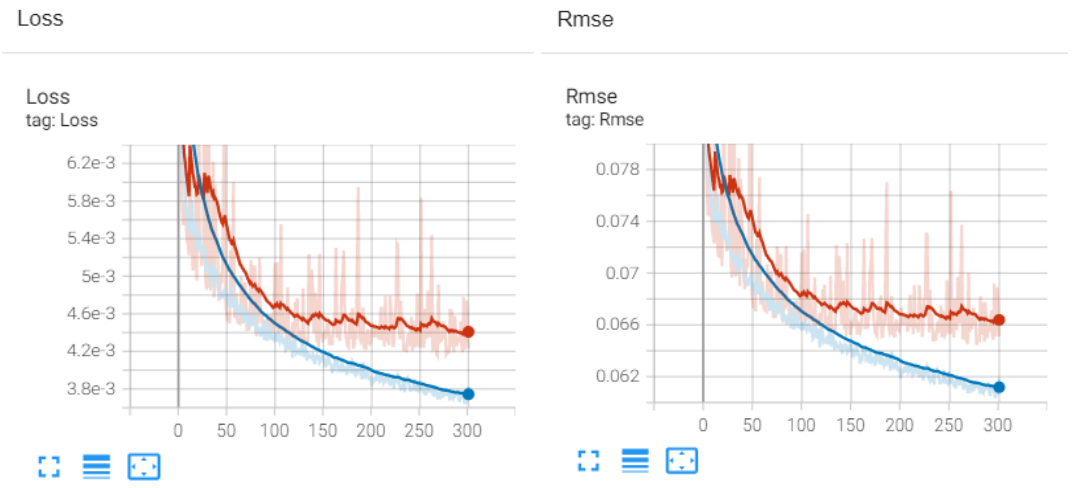


Figura 11. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color azul, y validación, en color marrón, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo E. Coli. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	6.45e-2

RMSE sobre conjunto de test	6.58e-2

Tabla 17. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 11. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Comparando los resultados de la Tabla 17 con los resultados obtenidos para la misma dupla target - indicador de los experimentos de Machine Learning, en este caso se obtiene un peor valor de RMSE, con un orden de magnitud por debajo. Pese a que el resultado obtenido es un orden de magnitud menor que el valor máximo que toma la variable respuesta en la distribución de datos de entrenamiento, el valor de RMSE es mayor que la media y del mismo orden de magnitud que la desviación estándar en dicha distribución. Como ya se ha comentado, esto puede ser indicativo de una incapacidad de generalización en la predicción para superar el modelo naif de considerar todas las predicciones como la media de la distribución de la variable respuesta.

Por otro lado, tal y como se observa en la Figura 11, en este caso se registran unas curvas de entrenamiento que decaen con el número de épocas, llegando incluso a alcanzar cierta estabilidad en el caso de la curva de validación. No obstante, no es un buen indicativo que la curva de validación se estabilice y la de entrenamiento no lo haga, pues esto es un indicativo de la incapacidad de generalización del modelo. Además, siguen apareciendo fuertes oscilaciones, pese a que de forma menos abrupta en la curva de entrenamiento y, este efecto, conjuntamente con la separación que existe entre las curvas al final del entrenamiento, indican un fenómeno de infra-ajuste y una capacidad insuficiente de generalización sobre la variabilidad en los datos de validación.

Con esto, pese a que la curva de entrenamiento no ha alcanzado la estabilidad, todos los factores anteriormente comentados indican que el límite de aprendizaje en este modelo no es capaz de superar el modelo estándar de aproximar todas las predicciones como la media de la variable respuesta. Aun así, como se verá más adelante, se puede tratar de comprobar si el origen de las oscilaciones es un *learning rate* demasiado alto, impidiendo a la función de pérdida llegar a su mínimo absoluto.

Para el objetivo *P. Aeruginosa*, el tiempo de ejecución de esta prueba es de 24 minutos y 51 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 12.

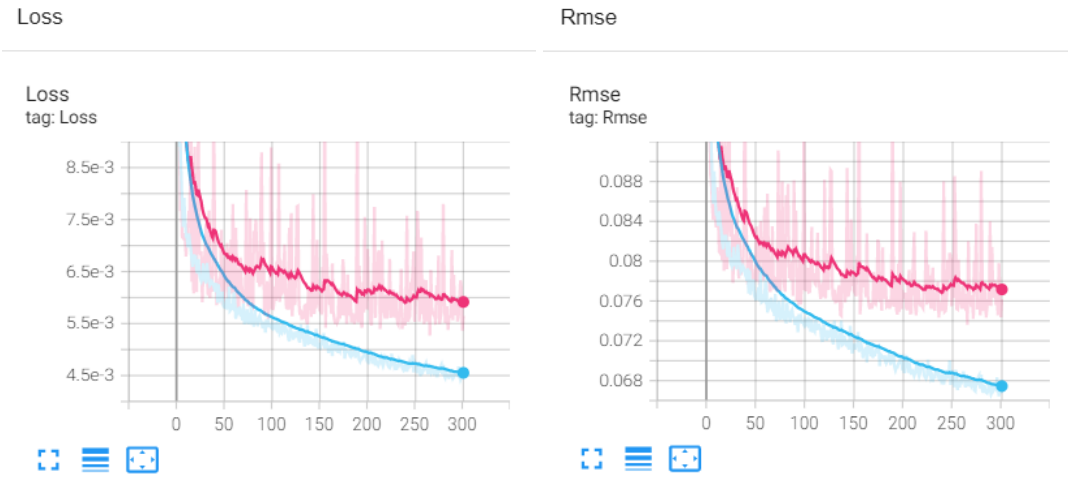


Figura 12. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color azul, y validación, en color rosa, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo P. Aeruginosa. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	7.36e-2
RMSE sobre conjunto de test	7.49e-2

Tabla 18. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 12. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Para el objetivo *S. Aureus*, el tiempo de ejecución de esta prueba es de 57 minutos y 47 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 13.

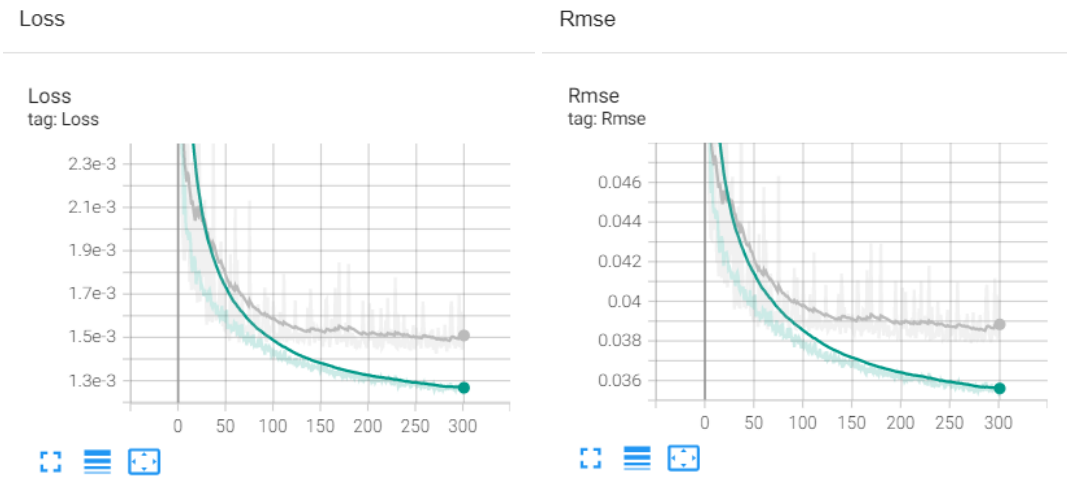


Figura 13. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color verde, y validación, en color gris, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo S. Aureus. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	3.79e-2
RMSE sobre conjunto de test	3.71e-2

Tabla 19. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 13. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Como se puede observar en las Figuras 12 y 13, el comportamiento de las curvas de entrenamiento en el caso de las bacterias *P. Aeruginosa* y *S. Aureus* es análogo al de la bacteria *E. Coli*. Aparece una separación entre las curvas de entrenamiento y validación y fuertes oscilaciones en la segunda, lo cual es indicativo de infra-ajuste e incapacidad de generalización de la relación entre los datos de entrada y la variable respuesta. No obstante, en los gráficos de la bacteria *S. Aureus* se aprecia cómo se alcanza más estabilidad en las 300 épocas, lo cual puede haber sido consecuencia de disponer de un conjunto de datos mayor, induciendo así que la influencia de aumentar el conjunto de datos únicamente refuerza más el comportamiento que ya presentan los otros casos, planteando así la veracidad hipótesis planteada previamente sobre el tamaño del *dataset*.

Por otro lado, en lo que respecta a las métricas de evaluación, se obtienen para las bacterias *P. Aeruginosa* y *S. Aureus* unos valores de RMSE del mismo orden de magnitud que en el caso de la bacteria *E. Coli*, con lo que se infieren para las mismas dos bacterias las mismas conclusiones que para la tercera. No obstante, resulta interesante ver cómo la bacteria *S. Aureus* presenta un mejor resultado, siendo el caso que contiene un conjunto de datos de entrenamiento mayor. Las diferencias relativas con las bacterias *P. Aeruginosa* y *E. Coli* para los conjuntos de datos de *test* son del 50% y del 44%, respectivamente, lo cual parece confirmar la hipótesis de que un mayor conjunto de datos de entrenamiento consigue, en este caso, un mejor resultado de RMSE. Esto no indica, sin embargo, que aumentar indefinidamente el conjunto de datos aumente indefinidamente el rendimiento del modelo - como se ha comentado, con esta configuración de experimentos parece haberse llegado a cierto límite en la capacidad de aprendizaje debido a la casi estabilidad de las curvas de entrenamiento.

Por último, para estudiar el origen de las oscilaciones y ver su posible relación con el parámetro de *learning rate*, se vuelven a ejecutar los tres experimentos anteriores con la misma configuración de la Tabla 16, a excepción del *learning rate*, que se reduce un orden de magnitud y se establece en 0.0001. La cantidad en la reducción es puramente arbitraria, puesto que en un estudio más elaborado se podría utilizar un optimizador de *learning rate*

como PyTorch Lightning [f] para proponer un valor óptimo a utilizar previo al propio entrenamiento de la red neuronal. A continuación se analizan los resultados para cada uno de los experimentos.

Para el objetivo *E. Coli*, el tiempo de ejecución de esta prueba es de 42 minutos y 19 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 13.

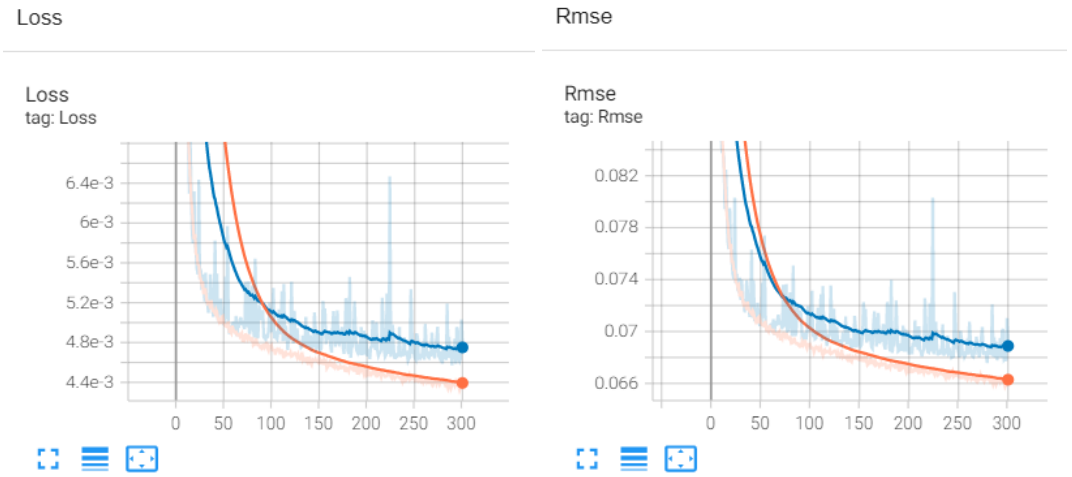


Figura 14. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color naranja, y validación, en color azul, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo E. Coli, con una reducción de LR a 0.0001. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	6.76e-2
RMSE sobre conjunto de test	6.78e-2

Tabla 20. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 14. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Para el objetivo *P. Aeruginosa*, el tiempo de ejecución de esta prueba es de 26 minutos y 29 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 13.

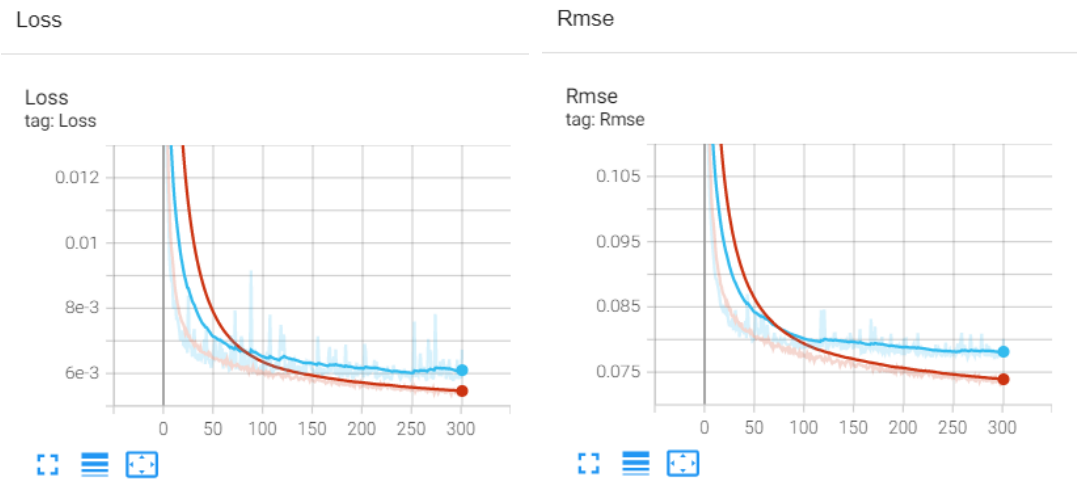


Figura 15. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color marrón, y validación, en color azul, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo P. Aeruginosa, con una reducción de LR a 0.0001. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	7.70e-2
RMSE sobre conjunto de test	7.73e-2

Tabla 21. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 15. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Para el objetivo *S. Aureus*, el tiempo de ejecución de esta prueba es de 1 hora, 4 minutos y 2 segundos, y los resultados de las curvas de entrenamiento se muestran a continuación en la Figura 13.

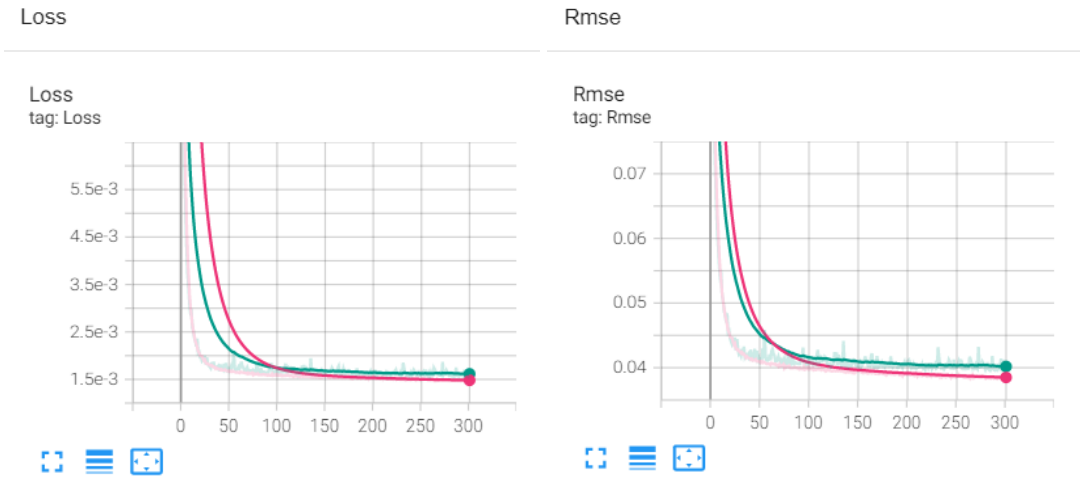


Figura 16. Curvas de la función de pérdida (izquierda) y métrica de evaluación RMSE (derecha) sobre los conjuntos de datos de entrenamiento, en color rosa, y validación, en color verde, sobre el experimento con la configuración correspondiente a la Tabla 16 y el objetivo S. Aureus, con una reducción de LR a 0.0001. Se muestran, por un lado, las curvas originales (curvas sombreadas) y las mismas curvas superpuestas con el máximo grado de suavización que permite la herramienta Tensor Board

El resultado del entrenamiento es:

Mejor RMSE de validación	3.94e-2
RMSE sobre conjunto de test	3.88e-2

Tabla 22. Resultados de la métrica de evaluación del experimento correspondiente a la Figura 16. Se muestra el mejor resultado obtenido sobre el conjunto de datos de validación, y el resultado obtenido sobre el conjunto de datos de test

Como se puede observar en las Figuras 14, 15 y 16 de los tres experimentos, el hecho de reducir el *learning rate* suaviza las oscilaciones de las curvas de entrenamiento tanto para la función de pérdida como para la métrica de evaluación. Además, en todas las gráficas y especialmente en el caso de la bacteria *S. Aureus*, el espacio entre las curvas de entrenamiento y validación se reduce, con lo que resuelve el problema de generalización y confirma que el tamaño de *batch* escogido es cualitativamente adecuado. Este resultado remarca la importancia de optimizar cada uno de los híper - parámetros que conforman un modelo, con tal de encontrar una solución óptima para así descartar todas las hipótesis que se plantean a partir de las distintas configuraciones posibles.

No obstante, en lo que respecta a la métrica de evaluación RMSE, no se ha superado el umbral de 1e-3. Dada la estabilidad que presentan las curvas tras las 300 épocas de entrenamiento, se concluye que se ha alcanzado el límite máximo de aprendizaje de esta arquitectura con la configuración dada, imposibilitando dar así con un modelo de predicción entrenado con las estructuras bidimensionales de las moléculas en formato tensor.

Por último, aunque no se haya obtenido un modelo predictivo válido, el haber hecho un cribaje de datos según el cuartil del 75% de la variable respuesta ha demostrado la capacidad del algoritmo de realizar un entrenamiento válido pese a limitado y ha permitido, como se verá a continuación, comparar el rendimiento de modelos de Deep Learning y modelos de Machine Learning en la tarea de regresión de bioactividad para el indicador MIC.

RESUMEN DEL ANÁLISIS DE RENDIMIENTO DE MODELOS

Se han realizado un total de 11 experimentos de Machine Learning y 10 experimentos de Deep Learning. Los mejores resultados obtenidos en cada caso se muestran a continuación en la Tabla 23.

	RMSE Conjunto de Test (Indicador de bioactividad MIC)		
Modelo \ Objetivo	**E. Coli**	**P. Aeruginosa**	**S. Aureus**
X-Gradient Boost	5.1e-3	6.1e-3	1.9e-3
Random Forest	7.9e-3	7.5e-3	5.6e-3
Support Vector Machine	5.4e-3	6.5e-3	2.5e-3
Graph Convolutional Net.	3.9e-2	7.7e-2	6.8e-2

Tabla 23. Resultados de la métrica de evaluación RMSE sobre los experimentos de machine learning y deep learning con mejores resultados para cada dupla target - indicador

La conclusión general del trabajo es que un modelo base de Deep Learning entrenado únicamente con datos relativos a estructuras moleculares 2D no supera un modelo de Machine Learning entrenado con indicadores moleculares calculados a partir de dichas estructuras en la tarea de predicción del indicador de bioactividad MIC; y no solo eso, sino que el modelo de Deep Learning es incapaz de generar un error inferior a la desviación estándar de la variable respuesta MIC, mientras que el modelo de Machine Learning sí podría ser considerado como un modelo predictivo a optimizar para su posterior puesta en producción.

Otras conclusiones que se extraen del análisis de experimentos son:

- La imposibilidad de trabajar con el indicador de bioactividad IC50. No obstante, cabe hacer hincapié en que no se ha puesto foco en el origen o causa de sus resultados en cuanto a incapacidad predictiva de los modelos con este indicador; tal vez con un análisis profundo del entendimiento de la variable podría generar modelos válidos.
- El algoritmo de Machine Learning que presenta mejores resultados en todas las bacterias de estudio es X-Gradient Boost. Si bien este resultado no es cualitativamente sorprendente dado el estado del arte que representa este tipo de algoritmo [11] y los buenos resultados obtenidos en otros estudios de predicción de bioactividad abordados como tareas de regresión [12], se ha obtenido un impresionante valor de precisión, entendida como el valor porcentual de la métrica de error respecto al rango de la variable respuesta, del 98.3%.

Tal y como se ha mencionado, en este desarrollo únicamente se han considerado modelos base con configuraciones de híper - parámetros que se consideran *por defecto*. Esto abre un extenso abanico de posibilidades que se podrían considerar de cara a extender el proyecto hacia próximos pasos, como podría ser, por ejemplo, utilizar un optimizador para ajuste de híper - parámetros en los algoritmos de Machine Learning o estudiar el efecto de reducir el tamaño del *batch* en los conjuntos de datos de entrenamiento para Deep Learning.

En la Memoria del trabajo se expondrán de manera más detallada, por un lado, posibles interpretaciones biológicas de los resultados obtenidos más allá de la interpretación técnica y, por otro lado, las posibles vías que se podrían tomar en próximos pasos del proyecto.

CREACIÓN REPOSITORIO GITHUB

Con objetivo de reunir y hacer accesible de forma pública el código, los datos, los modelos y los resultados generados a partir de este proyecto, se crea repositorio en la plataforma GitHub. Por falta de tiempo de desarrollo, no se generan los archivos necesarios para ejecutar experimentos a través de línea de comandos.

El enlace del repositorio es:

https://github.com/jvdiaz21/uoc-mbb-tfm-22-ia_bioactivity

Se recomienda consultar el repositorio y el archivo “README” correspondiente para consultar acerca de la estructura del proyecto en ese entorno.

Se añade una modificación del sistema de carpetas y archivos que finalmente ha resultado del trabajo en la plataforma Google Drive, el cual se muestra a continuación en la Figura 17.

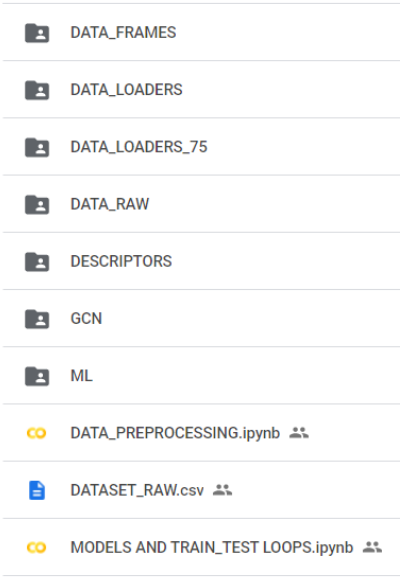


Figura 17. Directorios de Google Drive para el desarrollo del proyecto en línea con las rutas y archivos generados

El enlace del sistema de desarrollo en Google Drive es:

https://drive.google.com/drive/folders/16F1U1JSQMhgVE7PfZcTgpMNqPCdap0GH?usp=sharing

REFERENCIAS Y BIBLIOGRAFÍA

[a] XGBoost Regression algorithm for Python. (Last Visit: May 2022):

https://xgboost.readthedocs.io/en/stable/python/python_api.html

[b] Random Forest Regression algorithm for Python. (Last Visit: May 2022):

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

[c] Support Vector Machine Regression algorithm for Python. (Last Visit: May 2022):

https://scikit-learn.org/stable/modules/svm.html

[d] Escalador de preprocesado ‘MinMaxScaler’ de la librería Scikit-Learn. (Last Visit: May 2022):

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

[e] Herramienta Tensor Board del entorno Tensor Flow. (Last Visit: May 2022):

https://www.tensorflow.org/tensorboard?hl=es-419

[f] Módulo PyTorch Lightning de la librería PyTorch. (Last Visit: May 2022):

https://pytorch-lightning.readthedocs.io/en/latest/common/optimization.html

[1] Shearer, C. (2000) The CRISP-DM Model: The New Blueprint for Data Mining. Journal of Data Warehousing, 5, 13-22.

[2] Jo, J. M. (2019). Effectiveness of normalization pre-processing of big data to the machine learning performance. The Journal of the Korea institute of electronic communication sciences, 14(3), 547-552. https://doi.org/10.13067/JKIECS.2019.14.3.547

[3] Shmueli, G., Bruce, P. C., Stephens, M., & Patel, N. R. (2016). Data Mining for Business Analytics: Concepts, Techniques, and Applications with JMP Pro (3rd Edition). Wiley.

[4] Pang, B., Nijkamp, E., & Wu, Y. N. (2020). Deep Learning With TensorFlow: A Review. Journal of Educational and Behavioral Statistics, 45(2), 227–248. https://doi.org/10.3102/1076998619872761

[5] S. Bock and M. Weiß, (2019). A Proof of Local Convergence for the Adam Optimizer, 2019 International Joint Conference on Neural Networks (IJCNN), pp. 1-8, https://doi.org/10.1109/IJCNN.2019.8852239

[6] Wilson, D. & Martinez, Tony. (2001). The need for small learning rates on large problems. 1. 115 - 119 vol.1. https://doi.org/10.1109/IJCNN.2001.939002

[7] Cuevas, J. J., Martínez, E. P., Cortés, J. d. J. G., Pérez, S. S., & Campos, J. A. C. (2020). COMPARATIVA DE DESEMPEÑO DE LOS OPTIMIZADORES ADAM VS SGD EN EL ENTRENAMIENTO DE REDES NEURONALES CONVOLUCIONALES PARA LA CLASIFICACIÓN DE IMÁGENES ECG (COMPARATIVE PERFORMANCE OF ADAM VS. SGD OPTIMIZERS IN CONVOLUTIONAL NEURAL NETWORK TRAINING FOR THE CLASSIFICATION OF ECG IMAGES). Pistas educativas, 42(137 Especial). http://www.itc.mx/ojs/index.php/pistas/article/view/2300/1846

[8] Jais, I., Ismail, A., & Nisa, S. (2019). Adam Optimization Algorithm for Wide and Deep Neural Network. Knowledge Engineering and Data Science, 2(1), 41-46. doi:http://dx.doi.org/10.17977/um018v2i12019p41-46

[9] Kukacka, J., Golkov, V., & Cremers, D. (2017). Regularization for Deep Learning: A Taxonomy. doi: https://doi.org/10.48550/arXiv.1710.10686

[10] Chollet, F. (2017). Deep learning with python. Manning Publications.

[11] Bentéjac, C., Csörgő, A. & Martínez-Muñoz, G. (2021) A comparative analysis of gradient boosting algorithms. Artif Intell Rev 54, 1937–1967. https://doi.org/10.1007/s10462-020-09896-5

[12] Babajide Mustapha I, Saeed F. (2016) Bioactive Molecule Prediction Using Extreme Gradient Boosting. Molecules, 21(8):983. https://doi.org/10.3390/molecules21080983