



"VIGILADA MINEDUCACIÓN"

---

# Implementación de aprendizaje automático para la optimización de sistemas moleculares en celdas solares

---

Andres Dario Suarez, Cristian E. Susa

Departamento de Física y Electrónica, Universidad de Córdoba, 230002  
Montería, Colombia  
[andres.dario1906@gmail.com](mailto:andres.dario1906@gmail.com)

Jose D. Perea, Luis M. Mejia

Department of Chemistry, University of Toronto, ON, M5S 3H6, Canada  
Carlos A. Echeverry

Departamento de Química, Universidad Industrial de Santander,  
Colombia

October 6, 2020

## Abstract

Compuestos orgánicos, perovskitas y puntos cuánticos, son algunos sistemas físicos que hacen parte de la tercera y cuarta generación de celdas solares. El interés por mejorar las propiedades de estos sistemas con el fin de implementarlos para la conversión eficiente de energía lumínica a eléctrica ha tenido un fuerte incremento gracias a dos aspectos; los avances tecnológicos de control a nivel de sistemas cuánticos individuales, y el reciente interplay entre las áreas de información cuántica e inteligencia artificial. En este trabajo se explora el uso de técnicas de aprendizaje automático (ML) para ayudar a optimizar propiedades de compuestos moleculares como el orbital molecular de alta ocupación (HOMO) y las energías de orbital molecular desocupado más bajo (LUMO), así como el cálculo y calibración de la eficiencia de conversión de potencia (PCE). ) con el ánimo de buscar grandes candidatas a moléculas orgánicas para su uso como sistemas donante(-)receptor en células solares. En particular, probamos una calibración del proceso gaussiano como un modelo ML en un conjunto de moléculas reportadas en la literatura [?] y discutimos algunos aspectos tanto de las propiedades químicas como de la ventaja de usar ML

# Contents

<b>1</b>	<b>Consideraciones técnicas</b>	<b>1</b>
1.1	Notebook de Colab . . . . .	1
1.2	Carga de Repositorio . . . . .	1
1.3	Enlace a Orca . . . . .	1
<b>2</b>	<b>Avances Proceso Principal</b>	<b>1</b>
2.1	Carga de base de datos . . . . .	1
2.2	Generación y búsqueda de energía mínima conformers usando MMFF . .	2
2.3	Eliminación duplicación conformers . . . . .	4
2.4	Valores de HOMO y LUMO con DFT . . . . .	4
2.5	Gaussian Process Calibration . . . . .	4
2.6	Modelo de Scharber . . . . .	6
2.7	Calibración modelo de Scharber . . . . .	6
<b>3</b>	<b>Results</b>	<b>7</b>
<b>4</b>	<b>Hallazgos</b>	<b>8</b>
<b>5</b>	<b>Preguntas</b>	<b>8</b>

# 1 Consideraciones técnicas

En esta sección se expondrá consideraciones técnicas de la ejecución del proyecto que no cuentan como avances, pero que son necesarias para la ejecución.

## 1.1 Notebook de Colab

El código fuente se encuentra montado en un notebook de Colab, esto para poder permitir futuras ediciones por parte de miembros, así como una facilidad en la ejecución de procesos de IA o química cuántica que requieran procesadores o elementos complejos de hardware.

## 1.2 Carga de Repositorio

Para poder hacer uso de archivos externos, y exportar documentación que pudiese agregarse desde el proyecto se optó por utilizar un repositorio como github, el cual es accedido a través de un token, lo cual elimina la necesidad de estar agregando la contraseña cada vez que se ejecute el proyecto.

En el código fuente adicionalmente se encuentran agregadas las líneas de push y pull que permiten respectivamente la carga al repositorio y la descarga, esto con el fin de automatizar el proceso.

## 1.3 Enlace a Orca

Para el enlace a Orca se está en investigación la forma de cargue del programa dentro del servidor en colab, se dispone a descargar el .tar.xz y ejecutar comandos linux que permitan la instalación y comunicación entre el código fuente y Orca, esto para los procesos de DFT que requiere el proyecto.

# 2 Avances Proceso Principal

## 2.1 Carga de base de datos

La carga y la lectura en el código de la base de datos se realizó primero extrayendo la información dada en el artículo científico [1]. Estas bases de datos contenían la siguiente información:

Datafile S1 = dataOfAcceptors in python file	Comma separated value text file containing information on the entire dataset of acceptor molecules, 51280 in total.
Datafile S2 = data3 in python file	Comma separated value text file containing information on the 112 fragments used to build the library of molecules.
Datafile S3 = data4 in python file	Comma separated value text file containing information on fingerprint analysis utilizing 8192 bits with a connectivity radius of 4.
Datafile S4 = data5 in python file	Comma separated value text file containing information on molecules used for HOMO-LUMO GP calibration. Contains experimental values and values calculated with TDDFT.
Datafile S5 = data6 in python file	Comma separated value text file containing information on molecules used for PCE GP calibration. Contains experimental values, values calculated with TDDFT, scharber model values and values obtained after calibration.

La carga de las bases de datos se hicieron utilizando la librería de pandas en python, de la siguiente forma:

```

1 #Comma separated value text file containing information on the
2 #entire dataset of acceptor molecules, 51280 in total
3
4 dataOfAcceptors = pd.read_csv('/content/mmc2.csv')
5 dataOfAcceptors.head()
6

```

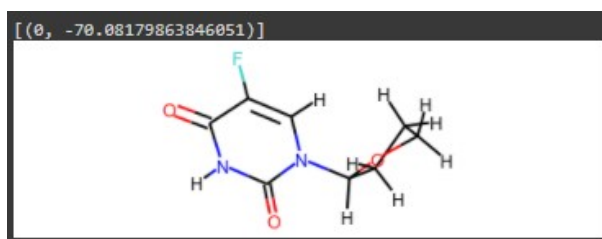
## 2.2 Generación y búsqueda de energía mínima conformers usando MMFF

Una vez realizada la carga de la base de datos se procedió a utilizar la data de las moléculas en formato SMILE para la generación de los conformers, estos conformers se hayaron utilizando la libreria rdkit

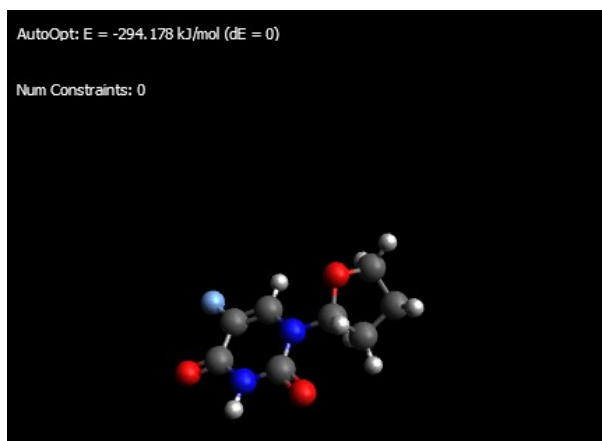


Se utilizó el método AllChem.EmbedMultipleConfs para la generación de los conformers, y se utilizó el método AllChem.MMFFGetMoleculeForceField, para la minimización de la energía a través del método MMFF.

Para constatar el modelo se utilizó el artículo de Hongbo Zhu [2], en el cual tomamos el nombre de la estructura BIPDEJ02 que reporta unos valores de -70.71969 y -70.71739 KCal/mol, para esta estructura se encuentra que su smile es FC1=CN(C2CCCO2)C(=O)NC1=O, el cual al ser ingresado en nuestro modelo nos entrega el siguiente valor.



Y en Avogadro:



NOTA: los valores dados en el modelo son en KCal/mol, mientras los valores dados en avogadro son en KJ/mol.

Como se puede constatar los valores obtenido en todos los sistemas son cercanos entre sí.

## 2.3 Eliminación duplicación conformers

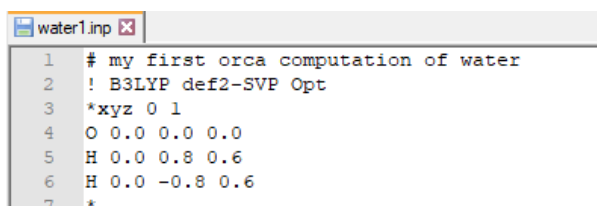
Finalizada la generación y cálculo de la mínima energía de los conformers, se evalúa si existen conformers duplicados, para ellos se busca si existen conformers con energía locales mínimas de igual valor.

## 2.4 Valores de HOMO y LUMO con DFT

ORCA es un programa flexible para química computacional que hace especial énfasis en propiedades espectroscópicas de moléculas de capa abierta. Trata una gran variedad de métodos de química cuántica desde métodos semiempíricos, la teoría del funcional de la densidad (Density Functional Theory, DFT), hasta métodos ab initio simples y de multireferencia, así como efectos relativísticos y medioambientales.



Para la ejecución de la solución presentada en el artículo se utilizaron las bases def2-SVP y los métodos BP86 y B3LYP, al consultar la documentación se encuentran todos disponibles, sin embargo una de las bases no se encuentra tal como la muestra el artículo, por lo que mientras se soluciona, se opta por utilizar en ambas la def2-SVP, la otra base era double-z def2-SVP



```
1 # my first orca computation of water
2 ! B3LYP def2-SVP Opt
3 *xyz 0 1
4 O 0.0 0.0 0.0
5 H 0.0 0.8 0.6
6 H 0.0 -0.8 0.6
7 *
```

Como se muestra en la anterior imagen el uso de orca para este sistema es simple en escritura, el paso posterior luego de comprobado que funciona la instalación, fue comunicar con el archivo en colab, sin embargo se encuentra en fase de despliegue del programa de orca para este proposito.

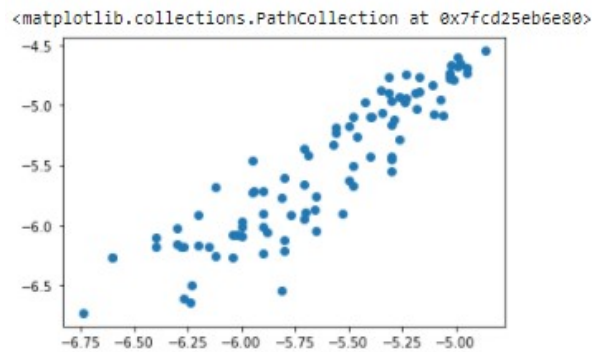
## 2.5 Gaussian Process Calibration

Luego de encontrado los valores de HOMO y LUMO, se debe proceder con la calibración de los mismos utilizando valores experimentales guardados en la base de datos "Datafile S4", para esto primero confirmamos que no existan valores nulos, y se hace un chequeo inicial tanto de distribución de datos como de limpieza de datos erróneos que de ser el caso se encontraran, luego de lo cual se procede a utilizar la librería de scikit Learn.

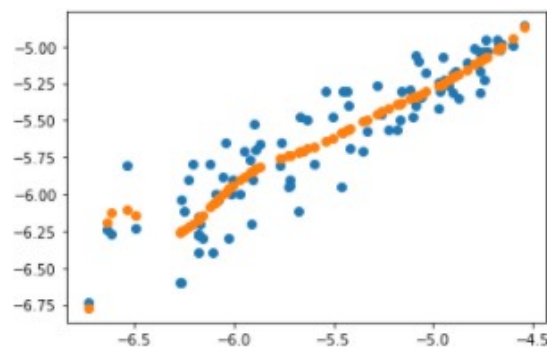
Con ella se implementa un proceso Gaussiano cuya clase es `GaussianProcessRegressor`, el cual en la instancia actual se deja sin agregar parámetros adicionales tales como el kernel.

Adicionalmente se utilizan técnicas como el `KFold`, para poder generar un entrenamiento sobre los datos que genere una salida con una precisión mayor, y se entrena el modelo a través de este sistema.

Al plotear la base de datos suministrada por el artículo obtenemos la siguiente gráfica.

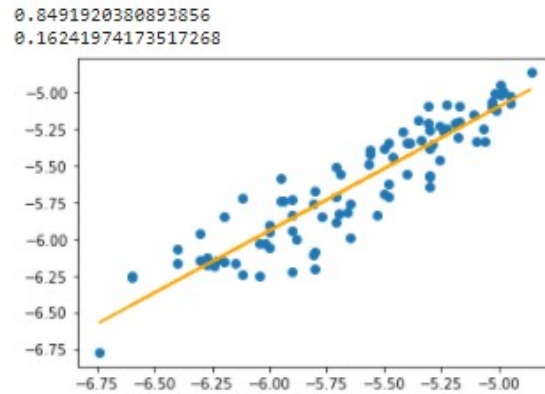


En la cual notamos inmediatamente su tendencia lineal, al agregar el proceso gaussiano descrito anteriormente para su calibración, obtenemos a la salida del proceso.



Al generar la salida tomando un modelo de regresion lineal se obtiene un score en los datos de cerca de 0.85, y un RMSE de 0.16.





Al comparar con la referencia [1], el cual obtuvo valores de score de 0.86 y RMSE de 0.17, podemos notar que el modelo es cercano en precisión al descrito en el artículo.

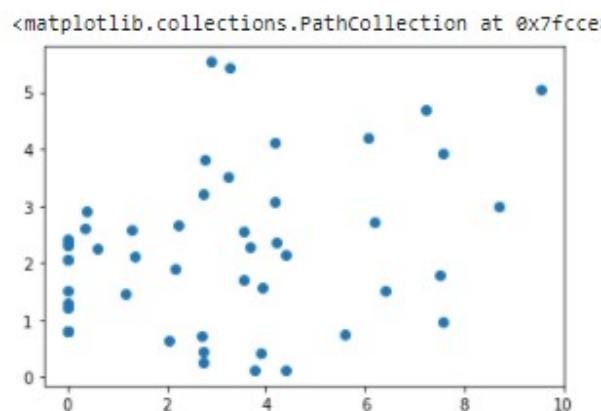
## 2.6 Modelo de Scharber

El modelo de Scharber nos permitirá finalmente con los datos anteriores encontrar la salida deseada, la cual es para qué tipo de configuración existe el menor valor de PCE, en este caso utilizamos lo expuesto en el artículo principal [1], agregando cambios en las ecuaciones de Jsh y Voc obtenidos en literatura adicional [3]

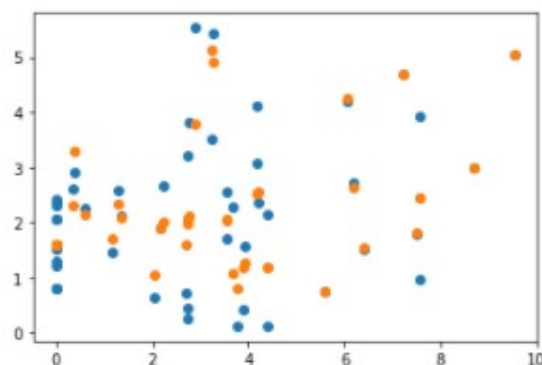
## 2.7 Calibración modelo de Scharber

La calibración del modelo de Scharber se realizó de la misma forma que con la calibración anterior, se utilizó un proceso gaussiano, y un KFold, sin embargo el Kfold fue variado en parámetros.

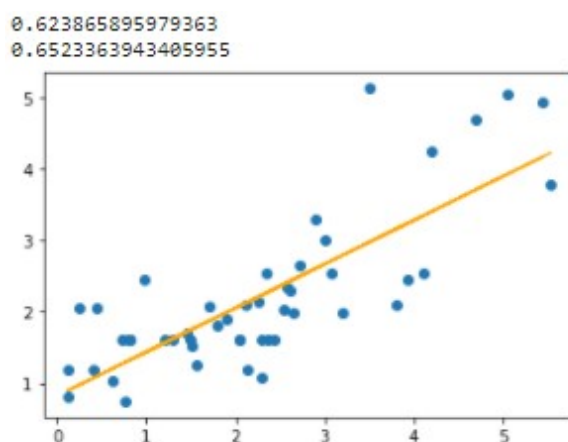
Al plotear los datos en la base de datos en el artículo, vemos que no existe una dependencia clara.



Es por ello que al entrenar el proceso gaussiano, no obtenemos valores tan precisos.



Con estas configuraciones se obtuvo un resultado de 0.62 de score y un RMSE de 0.65.



Al comparar con la referencia [1], el cual obtuvo valores de score de 0.20 y RMSE de 1.22, podemos notar que el modelo presenta mejoría en precisión al descrito en el artículo.

### 3 Results

Los resultados muestran que el modelo con el método MMF94 presenta valores acorde a los documentados en literatura [2]

El modelo de machine Learning presenta una fuerte semejanza en cuanto a los valores presentados así como su precisión en la calibración del HOMO y LUMO.

En las calibraciones del PCE, el proyecto actual muestra una mejora en cuanto a su precisión respecto al artículo principal.

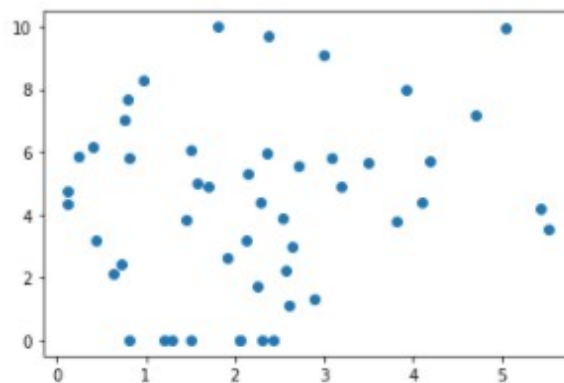
Actualmente el proyecto no se encuentra finalizado, para finalizar la replicación del proyecto faltan las siguientes tareas.

- Configurar Orca para comunicar con Python
- Realizar cálculo de DFT
- Mejorar modelo Scharber para datos obtenido de calibración, y realizar validación con datos en artículo principal.

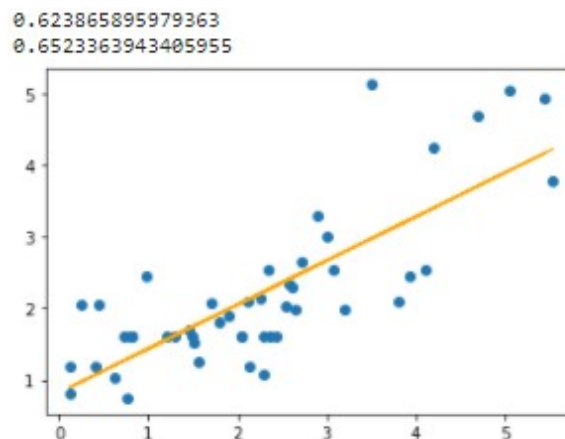
Lo anterior puede modificarse según el curso de la investigación y decisiones de sus participantes.

## 4 Hallazgos

La literatura adicional otorgada en el artículo principal [1], presenta variaciones que serán tema de análisis debido a que no concuerdan en datos y gráficas mostradas, ya que en la base de datos "Datafile S5", al plotear el valor de PCE exp vs PCE calib, obtenemos.



Siendo distinta a la visualizada en el material suplementario, sin embargo, el modelo del proyecto actual, nos muestra la siguiente gráfica.



La cual si concuerda en forma con la reportada en el artículo.

## 5 Preguntas

- Cómo configurar Orca, se planea usarlo desde colab, pero se desconoce si se puede ejecutar directamente desde el servidor de colab a través de comandos, o si se hace

necesario ejecutar el servicio en un sistema cloud, siendo que para el proyecto de ser así, se migraría completo el proyecto a la máquina virtual y se utilizaría jupyter en lugar de colab.

- Se planeaba utilizar docker como contenedor para el desarrollo de la aplicación, pero en consulta con colegas programadores, descartarían la posibilidad debido a que no existe una forma sencilla de utilizar docker y orca como imagen del contenedor, sin embargo se desea saber si alguno conoce una forma de implementar en docker a Orca, o si poseen alguna implementación hecha de ello.

## References

- [1] S. A. Lopez, B. Sanchez-Lengeling, J. de Goes Soares, and A. Aspuru-Guzik, “Design principles and top non-fullerene acceptor candidates for organic photovoltaics,” 2017.
- [2] H. Zhu, “Implementation and application of the mmff94 force field,” 2014.
- [3] F. H. Alharbi, S. N. Rashkeev, F. El-Mellouhi, H. P. Lüthi, N. Tabet, and S. Kais, “An efficient descriptor model for designing materials for solar cells,” 2015.