	GUÍA PARA PRESENTACIÓN DE INFORMES FINALES UAM	**CÓDIGO: GIN-GUI-001**
		VERSIÓN: 01
		FECHA ELABORACIÓN DEL DOCUMENTO: 23/ENE/2015

UNIVERSIDAD AUTÓNOMA DE MANIZALES

VICERRECTORÍA ACADÉMICA

UNIDAD DE INVESTIGACIÓN

UNIDAD DE POSGRADOS

TÓPICOS PARA LA PRESENTACIÓN DE INFORMES FINALES[1]

UNIVERSIDAD AUTÓNOMA DE MANIZALES

PROYECTO: Aplicación de redes neuronales profundas para la clasificación de secuencias de ADN.

GRUPO DE INVESTIGACIÓN: Ingeniería de software y Automática

ESTUDIANTE: Álvaro Andrés Pérez Sánchez

TUTOR DE TESIS: Simón Orozco Arias

CO TUTOR DE TESIS: Reinel Tabares Soto

DATOS DE IDENTIFICACIÓN: 10246864854

AÑO: 2020

1 Formato adaptado de COLCIENCIAS. 2006

1. RESUMEN

En el presente proyecto se usó una red neuronal completamente conectada, que es un sistema de nodos que están completamente conectados por los cuales pasa una información de entrada recorre dichos nodos hasta generar una salida. La red neuronal que se usó está basada en la publicada por Nakano et al., en el 2018, la cual fue utilizada en la clasificación de elementos transponibles (TEs). Los TEs son unidades genómicas que tienen la habilidad de replicarse o moverse a través de los cromosomas de prácticamente todos los organismos vivos. Estos elementos constituyen la mayor parte del contenido de ADN nuclear de los genomas de las plantas (Choulet et al., 2014).

la clasificación de Nakano se hace siguiendo un enfoque jerárquico y hasta el nivel de súper-familias. Partiendo de esta arquitectura, se afinaron los híper-parámetros más comunes como cantidad de capas ocultas, cantidad de neuronas por capa, funciones de activación, algoritmos de optimización, métodos de regularización, entre otros. Esta arquitectura fue entrenada y afinada usando una base de datos creada por el equipo y que consta de más de 60 mil muestras, cada una con más de 5 mil características. Para la ejecución de este proyecto, se usó arquitecturas GPUs y CPUs en ambientes de nube como Google Colabaratory y el lenguaje de programación Python con las librerías de Keras y Tensorflow.

PALABRAS CLAVES: Elementos transponibles, LTR retrotrasposones, bioinformática, redes neuronales profundas.

Abstract

In the present project, a fully connected neural network was used, which is a system of nodes that are completely connected through which an input information passes through the nodes until it generates an output. The neural network used is based on the one published by Nakano et al. in 2018, which was used in the classification of transposable elements (TEs). TE's are genomic units that have the ability to replicate or move through the chromosomes of virtually all living organisms. These elements constitute the majority of the nuclear DNA content of plant genomes (Choulet et al., 2014).

Nakano's classification is done following a hierarchical approach and up to the level of superfamilies. Based on this architecture, the most common hyper-parameters were refined, such as number of hidden

layers, number of neurons per layer, activation functions, optimization algorithms, regularization methods, among others. This architecture was trained and tuned using a database created by the team and which consists of more than 60 thousand samples, each with more than 5 thousand characteristics. For the execution of this project, we used GPU and CPU architectures in cloud environments such as Google Colabaratory and the Python programming language with the Keras and Tensorflow libraries.

Key works: Transposable elements, LTR retrotrasposons, bioinformatics, deep Leaning

2. ÁREA PROBLEMÁTICA Y JUSTIFICACIÓN

Los elementos transponibles (o TEs por sus siglas en inglés) evolucionan más rápidamente que los genes codificadores, estos elementos muestran una evolución dinámica debido a la inserción de otros TEs en sus secuencias (inserción anidada), la recombinación ilegítima y desigual, la captura de genes celulares y las duplicaciones cromosómicas (Garbus et al., 2015). Por esta razón, su clasificación y anotación es una tarea muy compleja (Bousios et al., 2012). Se han hecho muchos intentos para crear un sistema unificado de clasificación que combine los aspectos filogenéticos y enzimáticos, sin embargo, la clasificación se vuelve más difícil en los niveles más bajos como las súper-familias y los linajes (Negi et al., 2016). En algunos casos, se requiere de una investigación compleja y manual hecha por especialistas.

La agrupación de elementos por sus características estructurales comunes, funciona bien para grupos que tienen estructuras uniformes y mecanismos de integración bien definidos (Eickbush y Jamburuthugoda, 2008), pero hay pocas características compartidas a la hora de analizar otros tipos. Además, la anotación de genes, el uso de bases de librerías o secuencias de referencia de TEs para la identificación o clasificación es un reto importante, porque estos son específicos de cada especie. En consecuencia, se desconocen los TEs de las especies que se han secuenciado más recientemente (Girgis, 2015).

Aunque existen técnicas y herramientas bioinformáticas para la detección y clasificación de los elementos transponibles, aún no es posible obtener resultados confiables, debido a la gran diversidad en sus estructuras, formas de replicación y ciclos de vida. Además, estos componentes genómicos tienen características que hacen muy complejo su estudio, como la especificidad de cada especie, la gran diversidad a nivel de nucleótidos (Orozco S., Arango J. (2016)).

Actualmente se encuentra publicada una arquitectura basada en redes neuronales profundas para la clasificación de elementos transponibles (Nakano, et al., 2018). Sin embargo, esta investigación no tiene en cuenta el nivel de los linajes, sino que llega hasta súper-familias. Por otro lado, los autores utilizan una metodología jerárquica desde dos enfoques principales uno global y otro local. El enfoque local utiliza reducciones de machine learning para reducir el problema principal a otros más pequeños. después, se entrena a los clasificadores locales para cada sub-problema y se ensamblan

sus soluciones. El enfoque global entrena un solo clasificador capaz de manejar la jerarquía. Especialmente el enfoque local, demostró ser mejor a la clasificación basada en homología (Nakano, et al., 2018).

Gracias a la enorme cantidad de datos que se encuentran disponibles en la actualidad y la publicación de bases de datos de TEs, es posible la aplicación de técnicas de aprendizaje profundo. Además, las redes neuronales profundas han mostrado muy buenos resultados en otros problemas de la genómica y la bioinformática. Por esta razón, se hace factible el planteamiento de una arquitectura de red neuronal profunda para clasificar de forma automática los LTR retrotransposones de genomas de plantas.

3. REFERENTE TEÓRICO

En el ADN, hay elementos que son rodeados por medio de repeticiones terminales largas (LTR) y se denominan LTR retrotransposones (LTR-RTs). Otros retrotransposones carecen de LTR (no-LTR retrotransposones) y están separados en dos grupos, elementos intercalados largos (LINEs) y elementos intercalados cortos (SINEs) (Huang et al., 2012).

Además de estos grandes órdenes, hay otros dos, llamados DIRS y PLEs (Hermann et al., 2014). Los retrotransposones son la clase de elemento repetitivo más abundante ya que proliferan a través de un mecanismo de copiar y pegar mediado por el ARN, aumentando rápidamente su número de copias (Q. Li et al., 2018; Schulman, 2013). La organización estructural de los LTR retrotransposones (LTR-RTs) es similar a la de los retrovirus (Grandbastien, 2015; Q.-J. Zhang y Gao, 2017) excepto por la ausencia o presencia no funcional del gen envelope (Eickbush y Jamburuthugoda, 2008; Godinho et al., 2012). Los LTR-RTs son extremadamente variables en tamaño, oscilando en plantas desde 4 kb hasta más de 18-23 kb (Cossu et al., 2012; Flavia Mascagni et al., 2017).

Las secuencias de repetición de terminal largas (LTR) son regiones no codificantes que evolucionan más rápidamente que otros componentes de los LTR-RT (Grover y Sharma, 2017). Contienen señales de inicio y parada de transcripción (Alzohairy et al., 2014; Paz et al., 2017; Zhou et al., 2018), señales de poliadenilación y potenciadores (Giordani et al., 2016) que son críticos para el proceso de replicación (Gao et al., 2012).

El aprendizaje automático (o también llamado machine learning, ML) puede definirse como el proceso de diseño de un modelo que se calibra a partir de la información de entrenamiento y una función de pérdida a través de un algoritmo de optimización (Mjolsness y DeCoste, 2001). Basándose en estos patrones extraídos, los algoritmos pueden ahora predecir los resultados de datos desconocidos. Las principales formas de entrenamiento en ML pueden clasificarse en aprendizaje supervisado y aprendizaje no supervisado, el objetivo en la primera es predecir un valor discreto o valor continuo de cada punto de datos mediante el uso de un conjunto de ejemplo (llamado conjunto de entrenamiento), en el cual se tienen los datos etiquetados. En el aprendizaje no supervisado, que se basa en algoritmos de agrupamiento, el objetivo es aprender patrones inherentes dentro de los datos mismos (Zou et al., 2018).

El diseño e implementación de un sistema de ML es un proceso complejo que puede realizarse en tres pasos: 1) pre-procesamiento de datos brutos (por ejemplo, selección y extracción de características, imputación de datos, etc.), 2) aprendizaje o formación del modelo mediante el uso de un algoritmo o arquitectura ML apropiado (calibrar el modelo) y 3) evaluación del modelo mediante métricas (Ma et al.,2014).

El aprendizaje profundo (DL) integra en el propio problema de ML la tarea de seleccionar la representación correcta de los datos o las mejores características (Eraslan et al., 2019). Una de las características más útiles de las arquitecturas DL es que las DNN puede aprender características no lineales automáticamente, ya que cada capa utiliza múltiples modelos lineales y su salida es transformada por funciones de activación no lineales, como la función sigmoide o la unidad lineal rectificada (Eraslan et al., 2019). Este proceso facilita las tareas de clasificación, como la distinción entre retrotransposones Copia y Gypsy (Súper-familias de LTR retrotransposones). Nakano et al. (2018)

4. LOS OBJETIVOS

Objetivo General: Desarrollar un método computacional basado en redes neuronales completamente conectadas para clasificar LTR retrotransposones en genomas de plantas

Objetivos específicos:

● Definir una arquitectura de red neuronal profunda completamente conectada para la clasificación de LTR retrotransposones a nivel de linajes.

● Analizar y comparar el rendimiento de la arquitectura propuesta en términos de tiempos de ejecución y en precisión con otra arquitectura propuesta en la literatura.

5. METODOLOGÍA

Con el fin de dar solución al problema planteado se implementó una arquitectura de red neuronal profunda basada en el diseño que publicó Nakano et al., (2018), se verificaron los resultados para saber su rendimiento base y saber su fiabilidad. A continuación, se procedió a analizar qué híper parámetros deben cambiarse para mejorar el rendimiento, para lo cual se utilizó varias pruebas donde se entrena el modelo y para saber cuál cambió da mejores resultados. Primero se utilizó un dropout para eliminar un porcentaje de neuronas al azar y así bajar el sobre ajuste del modelo, que es el problema principal que se detectó. Se usó en primer lugar un valor de 0.2 que es el descarte del 20 por ciento, hasta llegar a 0.8. Debido a que esta medida no solucionó de manera efectiva el sobre ajuste, se utilizó a continuación un *Batch normalization* lo que hizo que llevara los datos a una tendencia más central para poder hacer más rápido y centralizar la red neuronal. Después se implementó regularización L1 y L2. En la regularización Lasso o L1, la complejidad C se mide como la media del valor absoluto de los coeficientes del modelo, la regularización Ridge o L2, la complejidad C se mide como la media del cuadrado de los coeficientes del modelo, se hizo con varios valores hasta encontrar una con mejores resultados, por separado e individual.

Una vez se afinó el modelo con el fin de reducir el sobre ajuste regulando el bias, kernel y los hiper parámetros se continuó a probar con una base de datos mayor que tiene LTR retrotransposones de 195 especies de plantas, con 60 mil muestras de las cuales se utiliza del 80% para training 10% para validación y 10% para test.

La arquitectura de red neuronal profunda se implementó sobre lenguaje Python versión 3 a través de las librerías Tensorflow 2.2 y Keras para una mayor facilidad, se ejecutó sobre Google Colaboratory con GPU como entorno de ejecución.

6. RESULTADOS

Una vez se implementa la arquitectura se prueba su rendimiento base, el cual se evidencia en la Figura 1.

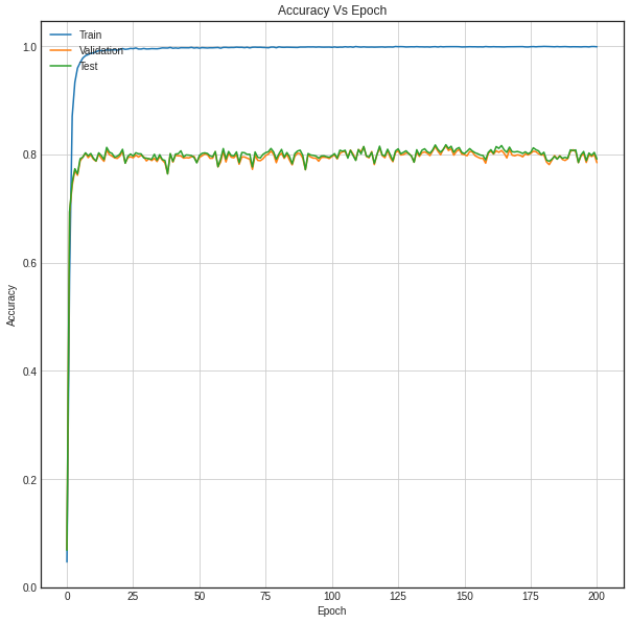

Figura 1: Precisión obtenida por la arquitectura de Nakano durante 200 épocas.

En la gráfica se muestra una gran diferencia entre la precisión del training y el test a lo que se le conoce como sobre ajuste. Para reducir el sobre ajuste se implementó un dropout a las capas de la arquitectura lo que indica que aleatoriamente se descartaran el porcentaje de las conexiones de las neuronas que indiquemos en cada capa. Se usaron valores de regularización con dropout de 0.2, 0.3, 0.4, 0.6, 0.7 y 0.8. La figura 2 muestra los resultados obtenidos por el modelo después de aplicar un dropout de 0.5 que fue el mejor resultado entre las mencionadas anteriormente

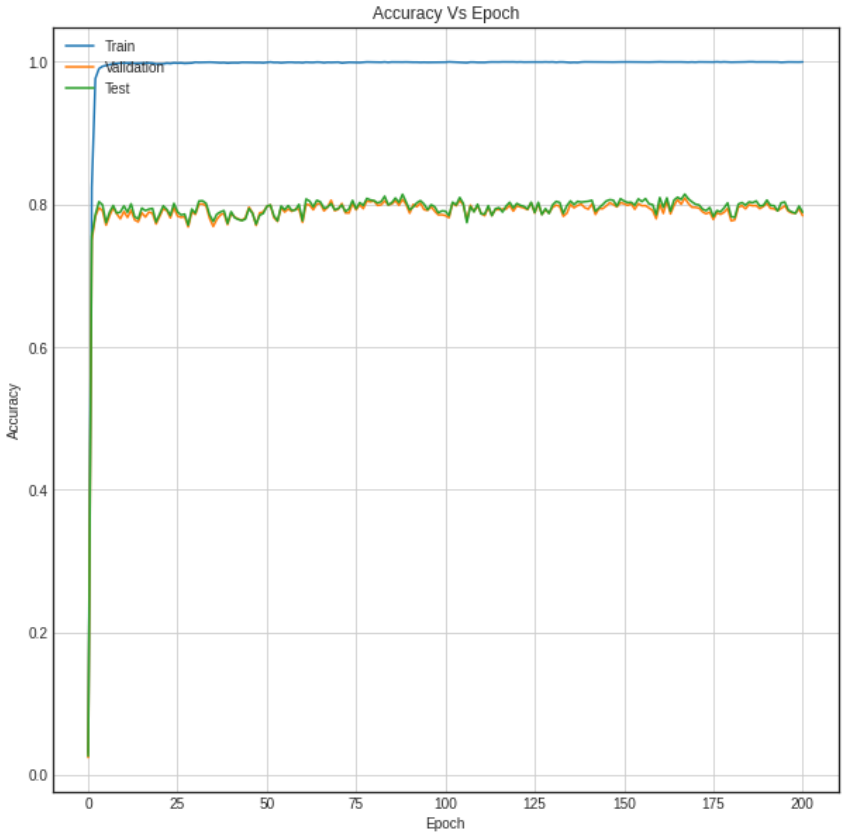

Figura 2: Precisión obtenida durante 200 épocas con un dropout 0.5.

El siguiente paso consistió en aplicar otra técnica de regularización con el fin de disminuir el sobre ajuste a través del método de *Layer weight regularizers*, en cada capa de la arquitectura se le hizo dicha regularización a el kernel. Inicialmente se probó con dos valores para L1=1e-5 y L2=1e-4 y se obtuvieron errores del 0.13 en el conjunto de entrenamiento, 16.96% en validación y 17.29 en test.

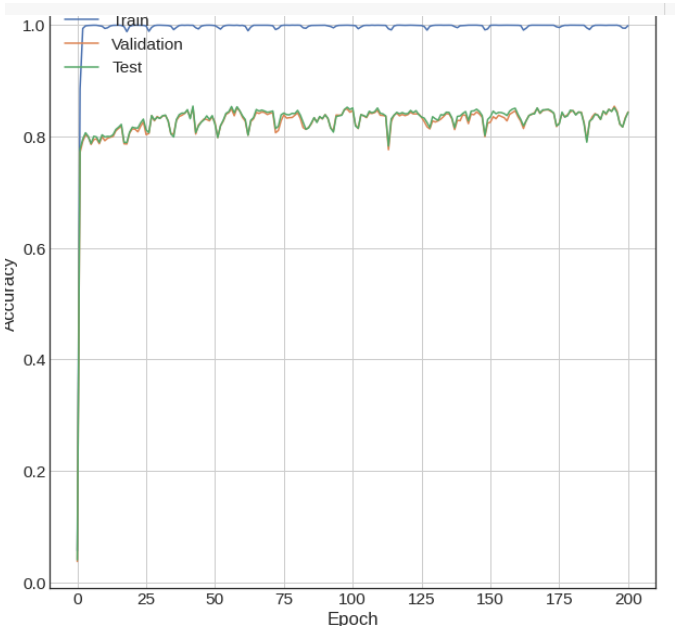

Figura 3: Precisión obtenida durante 200 épocas con regularización L1=1e-5 y L2=1e-4

En la Figura 3 se puede ver que la precisión en el conjunto de validación y test subió hasta 83.04% pero aún lejos del resultado esperado. se probó con distintos valores en regularización L1 y L2, pero los resultados no mejoraban a comparación del anterior, primero se probó con L1 que dió como resultado errores del 0.05% en el conjunto de entrenamiento, 17.91% en validación y 18.08% en test, esto no mejoro el rendimiento, por lo tanto, se probó solo con L2 y el resultado fue 0.01% en el conjunto de entrenamiento, 18.05% en validación y 17.84% en test.

Para lograr un mejor resultado se combinaron las regularizaciones en L1 y L2 en cada capa del modelo junto con el dropout de 0.5 que fue el que mejor resultado había dado con L1=1e-4 y L2= 1e-5 junto con dropout =0.5 el resultado fue 0.40% en el conjunto de entrenamiento, 16.89% en validación y 17.43% en test.

El anterior método tampoco mejoraba los resultados, se pasa a hacer regularización de L1 a el kernel y de L2 a la ordenada del origen (bias). Con valores de 0.0001 para L1 en el kernel y de 0.01 para L2 en el bias dio como resultado 0.21% en el conjunto de entrenamiento, 16.25% en validación y 16.01% en test.

Los resultados que muestra el modelo con la regularización anterior muestra un resultado óptimo para poder hacer pruebas con la base de datos de test que se necesitaba. Una vez que se cargó la base de datos nueva que tiene 195 especies de plantas, con 60 mil muestras se hace la prueba de rendimiento

con la misma arquitectura del último resultado lo que dio un rendimiento un 0.54% en el conjunto de entrenamiento, 1.53% en validación y 2.34% en test.

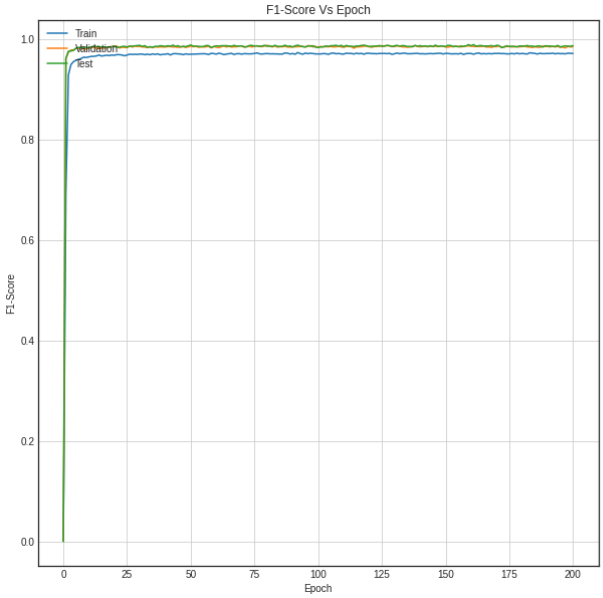

Figura 4: Precisión obtenida durante 200 épocas con nueva base de datos

La arquitectura muestra un rendimiento en el conjunto de test del 98.47% el cual indica que ya no hay un sobre ajuste en el modelo puesto que la diferencia entre la precisión del conjunto de entrenamiento y validación es de 1.01%.

Estas son las métricas obtenidas para la arquitectura de Nakano modificada, es un dataset muy desbalanceado.

Tabla 1. Rendimiento (F1-Score) de la arquitectura por cada linaje

Linaje	precisión	recall	f1-score	support
ALE/RETROFIT	0.99	0.99	0.99	1248
ANGELA	0.94	0.99	0.96	155
BIANCA	0.99	0.99	0.99	180

IKEROS	0.00	0.00	0.00	6
IVANA/ORYCO	0.95	0.97	0.96	341
TORK	0.97	0.95	0.96	599
SIRE	0.99	0.98	0.98	340
CRM	0.98	0.97	0.98	218
GALADRIEL	0.98	0.91	0.95	57
REINA	0.99	1.00	0.99	430
TEKAY/DEL	0.99	0.99	0.99	1055
ATHILA	0.99	0.98	0.99	362
TAT	1.00	1.00	1.00	1739

7. DISCUSIÓN DE RESULTADOS

Una red neuronal profunda es un modelo computacional que tiene la capacidad de aprender relaciones entrada-salida a partir de una gran cantidad de ejemplos. Estas redes están formadas por muchas unidades simples interconectadas entre sí llamadas neuronas, las cuales consisten de un modelo lineal seguido por una no-linealidad (Loncomilla, P. 2016). Estas redes han mostrado muy buenos rendimiento para procesar datos de gran tamaño y muy complejos, como secuencias de ADN.

El ADN no está disperso en nuestras células, está cuidadosamente empaquetado en estructuras llamadas cromosomas. El ADN está compuesto principalmente por cuatro sustancias químicas: adenina, timina, guanina y citosina A, T, G y C (G.Christopher P. Austin, M.D.). La mejor manera de

extraer la recurrencia de la frecuencia de aparición de las sub cadenas formadas por adenina, timina, guanina y citosina es utilizando k-mers (Orozco-Arias, et al., 2020).

Los k-mers son subsecuencias de longitud k contenida dentro de una secuencia biológica. Se utiliza principalmente en el contexto de la genómica computacional y el análisis de secuencias, en el que los k-mers están compuestos de nucleótidos (es decir, A, T, G y C), los k-mers se capitalizan para ensamblar secuencias de ADN, de manera que la secuencia AGAT tendría cuatro monómeros (A, G, A y T), tres en 2-mers (AG, GA, AT), dos 3-mers (AGA y GAT) y un 4-mer (AGAT). De manera más general, una secuencia de longitud L tendrá L - K + 1 k-mers y n^k serán los k-mers posibles (Ounit, Rachid; Wanamaker, Steve; Close, Timothy J; Lonardi, Stefano (2015)).

Se utilizaron los k-mers para extraer el conteo en que las cadenas se frecuentan y así poder obtener un arreglo de dichas frecuencias, adicionalmente se hace una curación de datos para tener una uniformidad en dicho conjunto de datos, lo que hace es que aquellos datos sucios como (características nulas, o campos en blanco, cadenas cuando se deben usar en forma numérica) se manejan para poder ser utilizados de forma correcta y no tener errores de algún tipo en el resultado final del modelo.

Se puede observar que el valor en F1-Score obtenido por el modelo afinado es del 98.47% (Figura 4) mientras que la reportada por Nakano es del 96% en el primer nivel de jerarquía, siendo este el modelo que tomamos como referencia para implementar la arquitectura, modificando en las capas del modelo para ajustarlo a el problema, aplicando un dropout en las capas para rebajar el sobre ajuste inicial, posteriormente una regularización de bias y kernel para mejorar el rendimiento, estos cambios en el modelo original permitió mejores resultados. Además, un factor que también influyó en la mejora de la precisión fue que el problema se centra en un conjunto de datos más específicos que el modelo planteado por Nakano, et at. Los autores planearon la clasificación de varios órdenes de elementos transponibles hasta superfamilias esta clasificación no usan datos exclusivamente de plantas, lo que podría generar resultados menos precisos que los alcanzados por la arquitectura propuesta en este proyecto puesto que fue planteada para la clasificación de un solo orden hasta el nivel de linaje.

8. CONCLUSIONES

Se concluye que la arquitectura de Nakano puede ser utilizada para clasificar los LTR retrotransposones, se implementa y se observa que su rendimiento puede ser mejor por lo que la regularización de bias y kernel con L1 y L2 en la capas de la arquitectura resultan ser un buen método para disminuir el sobre ajuste del modelo.

La profundidad de clasificación de los LTR retrotransposones afecta el rendimiento de la arquitectura puesto que es un problema más general y por ende más complicado de clasificar por lo tanto es mejor centrarse en un solo orden y no en muchos como hizo Nakano et al. con su clasificación de varios órdenes de elementos transponibles hasta superfamilias.

9. RECOMENDACIONES

Principalmente se recomienda el lenguaje Python para la implementación de machine learning ya que la legibilidad del código desarrollado en Python es sencillo, elegante y busca ser consistente. Esto permite que la estructura del lenguaje se asemeja a la estructura que implementamos los seres humanos y a lo que conocemos como lenguaje matemático permitiendo que este sea leído como un pseudocódigo.

Por otro lado, se sugiere la utilización de Keras puesto que es librería de Python que proporciona, de una manera sencilla, la creación de una gran gama de modelos de Deep Learning usando como backend otras librerías como TensorFlow.

Finalmente, se recomienda la utilización de Google Colaboratory puesto que nos proporciona un entorno de ejecución para jupyter notebook, esto facilita la ejecución de la implementación sino contamos con una computadora con las características necesarias, puesto que Google Colaboratory brinda una gran capacidad de procesamiento en RAM y entornos como GPU o TPU.

10. EVIDENCIA DE RESULTADOS

Relacionados con la generación de conocimiento y/o nuevos desarrollos tecnológicos

Resultado/Producto esperado	indicador	Beneficiario
Documento de Diseño	1	comunidad científica y comunidad UAM
Implementación	1	comunidad científica y comunidad UAM

11. IMPACTOS LOGRADOS

Impacto esperado	**Plazo (años) después de finalizado el proyecto: corto (1-4), mediano (5-9), largo (10 o más)**	**Indicador verificable**	**Supuestos[2]**
Aporte de herramientas automáticas para la clasificación de LTR retrotransposones	Mediano	Artículos científicos y otras investigaciones	La cantidad de información genómica seguirá creciendo
los resultados esperados contribuirán con información relevante	Mediano	Investigadores que tomen como referencia la	Se secuenciarán un gran número de

[2] Los supuestos indican los acontecimientos, las condiciones o las decisiones, necesarios para que se logre el impacto esperado.

sobre el aprovechamiento de técnicas de redes neuronales profundas para la clasificación de LTR retrotransposones		arquitectura planteada	especies de plantas en los próximos años
Contribuir a la formación de estudiantes	2	Número de estudiantes vinculados al semillero	Divulgación de la investigación en la comunidad UAM

12. BIBLIOGRAFÍA

Bousios A., Minga E., Kalitsou N., Pantermali M., Tsaballa A., Darzentas N. (2012). MASiVEdb: the Sirevirus Plant Retrotransposon Database. *BMC Genomics, 13*(1): 158.https://doi.org/10.1186/1471-2164-13-158

Chang W., Jääskeläinen M., Li S., Schulman A. H. (2013). BARE retrotransposons are translated and replicated via distinct RNA pools. *PLoS One, 8*(8): e72270. https://doi.org/10.1371/journal.pone.0072270

Cossu R. M., Buti M., Giordani T., Natali L., Cavallini A. (2012). A computational study of the dynamics of LTR retrotransposons in the Populus trichocarpa genome. *TREE Genet. GENOMES, 8*(1): 61–75. https://doi.org/10.1007/s11295-011-0421-3

Eickbush T. H., Jamburuthugoda V. K. (2008). The diversity of retrotransposons and the properties of their reverse transcriptases. *VIRUS Res., 134*(1–2): 221–234. https://doi.org/10.1016/j.virusres.2007.12.010

Eraslan G., Avsec Ž., Gagneur J., Theis F. J. (2019). Deep learning: new computational modelling techniques for genomics. *Nat. Rev. Genet.* https://doi.org/10.1038/s41576-019-0122-6

Garbus I., Romero J. R., Valarik M., Vanžurová H., Karafiátová M., Cáccamo M., ... Echenique V. (2015). Characterization of repetitive DNA landscape in wheat homeologous group 4 chromosomes. *BMC Genomics, 16*(1): 375. https://doi.org/10.1186/s12864-015-1579-0

Negi P., Rai A. N., Suprasanna P. (2016). Moving through the Stressed Genome: Emerging Regulatory Roles for Transposons in Plant Stress Response. *Front. Plant Sci., 7*. https://doi.org/10.3389/fpls.2016.01448

Nakano F. K., Mastelini S. M., Barbon S., Cerri R. (2018). Improving Hierarchical Classification of Transposable Elements using Deep Neural Networks. *Proceedings of the International Joint Conference on Neural Networks* Vol. 2018–July. Rio de Janeiro, Brazil. Disponible en http://dx.doi.org/10.1109/IJCNN.2018.8489461

Orozco S., Arango J. (2016). Aplicación de la inteligencia artificial en la bioinformática, avances,definiciones y herramientas. *UGCiencia, 159–171*.

Orozco-Arias, S., Piña, J. S., Tabares-Soto, R., Castillo-Ossa, L. F., Guyot, R., & Isaza, G. (2020). Measuring Performance Metrics of Machine Learning Algorithms for Detecting and Classifying Transposable Elements. Processes, 8(6), 638.

Nakano F. K., Pinto W. J., Pappa G. L., Cerri R. (2017). Top-down strategies for hierarchical classification of transposable elements with neural networks. *Proceedings of the International Joint Conference on Neural Networks* Vol. 2017–May, pp. p2539–2546. Anchorage, AK, United states. Disponible en http://dx.doi.org/10.1109/IJCNN.2017.7966165

Huang C. R. L., Burns K. H., Boeke J. D. (2012). Active transposition in genomes. *Annu. Rev. Genet., 46*(1):651–675. https://doi.org/10.1146/annurev-genet-110711-155616

Hermann D., Egue F., Tastard E., Nguyen D.-H., Casse N., Caruso A., ... Rouault J. D. (2014). An introduction to the vast world of transposable elements - what about the diatoms? *DIATOM Res., 29*(1): 91–104. https://doi.org/10.1080/0269249X.2013.877083

Loncomilla, P. (2016). Deep learning: Redes convolucionales. *Recuperado de https://ccc. inaoep. mx/~ pgomez/deep/presentation*

Grandbastien M.-A. (2015). LTR retrotransposons, handy hitchhikers of plant regulation and stress response. *Biochim. Biophys. Acta, 1849*(4): 403–416. https://doi.org/10.1016/j.bbagrm.2014.07.017

Li Q., Zhang Y., Zhang Z., Li X., Yao D., Wang Y., ... Xiao Y. (2018). A D-genome-originated Ty1/Copia-type retrotransposon family expanded significantly in tetraploid cottons. *Mol. Genet. Genomics, 293*(1): 33–43. https://doi.org/10.1007/s00438-017-1359-4

Zhang Q.-J., Gao L.-Z. (2017). Rapid and Recent Evolution of LTR Retrotransposons Drives Rice Genome Evolution During the Speciation of AA- Genome Oryza Species. *G3 Genes, Genomes, Genet., 7*(6): 1875–1885. https://doi.org/10.1534/g3.116.037572

Godinho S., Paulo O. S., Morais-Cecilio L., Rocheta M. (2012). A new gypsy-like retroelement family in Vitis vinifera. *VITIS, 51*(2): 65–72.

Mascagni F., Giordani T., Ceccarelli M., Cavallini A., Natali L. (2017). Genome-wide analysis of LTR-retrotransposon diversity and its impact on the evolution of the genus Helianthus (L.). *BMC Genomics, 18*(1): 634. https://doi.org/10.1186/s12864-017-4050-6

Ounit, Rachid; Wanamaker, Steve; Close, Timothy J; Lonardi, Stefano (2015).CLARK: fast and accurate classification of metagenomic and genomic sequences using discriminative k-mers.

Grover A., Sharma P. C. (2017). Repetitive Sequences in the Potato and Related Genomes. En: pp. 143–160. https://doi.org/10.1007/978-3-319-66135-3_9

Alzohairy A. M., Sabir J. S. M., Gyulai G., Younis R. A. A., Jansen R. K., Bahieldin A. (2014). Environmental stress activation of plant long-terminal repeat retrotransposons. *Funct. PLANT Biol., 41*(6): 557– 567. https://doi.org/10.1071/FP13339

Paz R. C., Kozaczek M. E., Rosli H. G., Andino N. P., Sanchez-Puerta M. V. (2017). Diversity, distribution and dynamics of full-length Copia and Gypsy LTR retroelements in Solanum lycopersicum. *Genética, 145*(4–5): 417–430. https://doi.org/10.1007/s10709-017-9977-7

Zhou M., Liang L., Hänninen H. (2018). A transposition-active Phyllostachys edulis long terminal repeat (LTR) retrotransposon. *J. Plant Res., 131*(2): 203–210. https://doi.org/10.1007/s10265-017-0983-8

Giordani T., Cossu R. M., Mascagni F., Marroni F., Morgante M., Cavallini A., Natali L. (2016). Genome-wide analysis of LTR-retrotransposon expression in leaves of Populus x canadensis water-deprived plants. *TREE Genet. GENOMES, 12*(4). https://doi.org/10.1007/s11295-016-1036-5

Gao D., Chen J., Chen M., Meyers B. C., Jackson S. (2012). A highly conserved, small LTR retrotransposon that preferentially targets genes in grass genomes. *PLoS One, 7*(2): e32010. https://doi.org/10.1371/journal.pone.0032010

Gao D., Jimenez-Lopez J. C., Iwata A., Gill N., Jackson S. A. (2012). Functional and structural divergence of an unusual LTR retrotransposon family in plants. *PLoS One, 7*(10): e48595. https://doi.org/10.1371/journal.pone.0048595

Mjolsness E., DeCoste D. (2001). Machine learning for science: state of the art and future prospects. *Science (80-.)., 293*(5537): 2051–2055.

Zou J., Huss M., Abid A., Mohammadi P., Torkamani A., Telenti A. (2018). A primer on deep learning ingenomics. *Nat. Genet*. https://doi.org/10.1038/s41588-018-0295-5

Ma C., Zhang H. H., Wang X. (2014). Machine learning for Big Data analytics in plants. T*rends Plant Sci., 19*(12): 798–808. https://doi.org/10.1016/j.tplants.2014.08.004

13. ANEXOS

Nakano_FNN.ipynb

Documento de Diseño