UNIVERSIDAD DE ANTIOQUIA

Laboratorio virtual para el procesamiento y clasificación de señales e imágenes biomédicas.

Autor(es)
Pedro Iván Vásquez Hernández

Universidad de Antioquia
Ingeniería, Bioingeniería
Medellín, Colombia
2019

Laboratorio virtual para el procesamiento y clasificación de señales e imágenes biomédicas.

Pedro Iván Vásquez Hernández

Trabajo final de semestre de industria presentada(o) como requisito parcial para optar al título de:
Bioingeniero

Asesores (a):

Angelower Santana Velasquez, Bioingeniero

Línea de Investigación:
Informática

Universidad de Antioquia
Ingeniería, Bioingeniería
Medellín, Colombia
2019.

Laboratorio virtual para el procesamiento y clasificación de señales e imágenes biomédicas.

Resumen

El fin de este proyecto se centró en la reducción de la brecha que existe entre el aprendizaje y las nuevas tecnologías que se desarrollan en pleno siglo XXI en instituciones educativas y más concretamente en Universidades públicas donde la investigación es su principal fuente de ingresos y donde es muy difícil conseguir fuente de financiamiento para recursos en la ejecución de algoritmos complejos con alta demanda computacional[1]. En el área de Bioingeniería de la Universidad de Antioquia se cuenta con investigaciones en el área del procesamiento de imágenes y señales biomédicas, como lo es el caso de imágenes de FMRI (Functional Magnetic Resonance Imaging) y señales EEG (Electroencefalografía), donde actualmente tecnologías como Nipype, ICA y redes neuronales convolucionales consumen demasiados recursos locales, sin mencionar el factor tiempo que esto implica imposibilitando el avance en el área de la investigación y desarrollo de nuevas aplicaciones en el área médica. Es por esto por lo que se planteó el despliegue de servicios en *Amazon Web Services (AWS)* donde se pudieran ejecutar algoritmos computacionalmente complejos y de ejecución lenta con el fin de reducir los tiempos de procesamiento y el consumo de recursos locales. Se desarrolló un algoritmo en *Machine Learning* por medio de redes neuronales convolucionales usando como herramienta la librería TensorFlow, dando como resultado la reducción del tiempo de ejecución, un algoritmo de procesamiento de imágenes, muy común en el área de FMRI usando Nipype como herramienta donde se logró reducir el tiempo de ejecución de 1 hora a 7 minutos; y en señales se desplegó un servicio que realizará la descomposición de una mezcla de una señal EEG en sus componentes independientes, a pesar de que no se logró una reducción de tiempo considerable de tiempo si se evidenció una disminución de la carga en el computador para la realización de sus otros procesos. Finalmente se concluye que AWS es una gran oportunidad de implementar algoritmos complejos y ayudar a las instituciones educativas el aprendizaje de nuevas tecnologías en todas las áreas de investigación y de desarrollo para seguir mejorando la calidad de vida de todas las personas.

Introducción

En la actualidad se presenta gran cantidad de métodos para el procesamiento de señales e imágenes con el fin de obtener un mejor análisis de estas [2]. Uno de estos métodos en el campo del procesamiento de imágenes es la detección de contornos que permite obtener información acerca de los límites de un objeto dentro de una imagen que podrán ser utilizados para análisis y aplicación de filtrado, de igual forma se emplea para simplificar el análisis ya que permite reducir la cantidad de datos a

procesar. Existen diversas técnicas de detección de contornos como lo es el filtrado convolucional, el cual consiste en aplicar una matriz sobre cada píxel dentro de una imagen y realizar la convolución de los datos [3]. Además de esto existen métodos para el procesamiento y reconocimiento de señales como lo es la transformada Wavelet discreta, esta se ha utilizado para el filtrado de señales en el reconocimiento de patrones. También se ha venido implementado el uso de redes neuronales por retro-propagación para la clasificación de señales [4].

Para llevar a cabo estos métodos, y mucho más en el procesamiento de señales e imágenes, existen softwares que permiten realizarlos; sin embargo, para estas operaciones se requiere que los computadores empleados cuenten con buenas especificaciones de hardware lo que resulta en elevados precios de inversión[5]. Adicional al problema económico también se tiene el inconveniente que, si se quiere procesar gran cantidad de imágenes en algoritmos mucho más complejos, la velocidad de procesamiento del equipo se ve comprometida y el tiempo para obtener los resultados esperados se incrementa [6]. Los softwares actuales para el procesamiento de imágenes y señales requieren ciertos parámetros mínimos en el equipo, adicional a esto se tiene el problema de la disponibilidad, la concurrencia y la escalabilidad en cuanto al desarrollo e implementación de algoritmos para el procesamiento de señales e imágenes biomédicas ya que debido a su complejidad y que manejan grandes volúmenes de datos estas características se ven comprometidas; es por esto que se ve la necesidad de implementar estas aplicaciones dentro de un espacio virtual o utilizar los principios de la computación en la nube con el fin de reducir los costos y los tiempos de respuesta.

La finalidad de este proyecto fue emplear los conceptos y servicios de Amazon Web Service (AWS), que estuvieran al alcance dentro de la capa gratuita, para desplegar servicios en la nube que permitieran el procesamiento de imágenes y señales biomédicas de manera sencilla sin verse comprometidos los recursos locales. Se logró emplear servicios como Elastic Compute Cloud (EC2) para el procesamiento de señales e imágenes disminuyendo los tiempos de cómputo y los recursos físicos; para este servicio fue necesario utilizar protocolos de comunicación SSH, peticiones por dirección IP y puertos de conexión de red. También se empleó el servicio de Amazon SageMaker, en el que se pudo desarrollar un algoritmo en Machine Learning con redes neuronales convolucionales para el reconocimiento de imágenes de la Malaria y así determinar si se padecía o no la enfermedad. Adicional a esto se emplearon herramientas como Docker, Git y CLI para facilitar el despliegue de los algoritmos en AWS; siendo estos contrastados con el rendimiento en máquina local, usando herramientas como Anaconda. Finalmente, se buscó desplegar los servicios con el fin de que fueran utilizados como un laboratorio, un espacio para el aprendizaje y experimentación médica ya que actualmente no se cuenta con los recursos necesarios para desarrollar algoritmos complejos de procesamiento de

imágenes y señales biomédicas que podrían influir de manera positiva y proactiva las investigaciones en el área de la salud.

Objetivos

Objetivo general:

Diseñar e implementar un laboratorio virtual para el procesamiento de imágenes y señales biomédicas, utilizando las herramientas de Amazon web service que permitan tener una disponibilidad, accesibilidad y escalabilidad en la ejecución de las tareas propuesta en el laboratorio.

Objetivos específicos:

- Implementar algoritmos que permitan el procesamiento de imágenes y señales biomédicas en un IDE que este soportado por Amazon Web Service
- Diseñar una metodología utilizando herramientas de Machine Learning que permita hacer clasificación de imágenes médicas por medio de redes neuronales.
- Desplegar una arquitectura básica de servicios en Amazon web service que permita usar las herramientas de instancias, contenedores o bases de datos, IDEs y Machine Learning.
- Verificar las herramientas desplegadas en AWS y validar con las herramientas utilizadas actualmente.

Marco Teórico

Computación en la nube:

La computación en la nube se define como un conjunto de tecnologías que están configurando un nuevo orden mundial en las tecnologías de la información (TI). La idea clave es que las TI se conviertan en un servicio, de modo que las aplicaciones del software no tienen porqué existir en un lugar concreto, sino que pueden estar compuestos de múltiples piezas procedentes de múltiples sitios. Eso quiere decir, cualquier cosa que tenga que ver con la provisión de servicios a través del internet [6]. Estos servicios se dividen en tres categorías: Infraestructura como servicio (IaaS), plataforma como servicio (PaaS) y software como servicio (SaaS) donde cada uno presenta los mismos servicios, pero de diferente manera. Los servicios en la nube tienen tres características bien diferenciadas de los hostings tradicionales. Por un lado, el cobro se realiza en función del uso, el servicio es elástico, el usuario puede utilizar tanto como quiera y en el momento que lo desee; finalmente, el servicio es totalmente gestionado por el proveedor [7].

Dentro de los muchos proveedores que hay a nivel mundial se tiene Google Cloud que posee servicios como Compute Engine y App Engine, que facilitan el despliegue de máquinas virtuales; Cloud Storage, Cloud SQL, donde se pueden almacenar datos y desarrollar bases de dato [8]; entre otros. Google Cloud tiene su equivalente qué es Amazon Web Service, este provee servicios en la nube bajo demanda, dentro de los servicios más comunes se encuentra: EC2, es un servicio para desplegar servidores en la nube con la característica de auto-escalado para suplir los requerimientos por una aplicación; EBS, se usa como unidad de almacenamiento para las instancias de EC2; S3, es una unidad de almacenamiento simple, provee APIs para almacenar y recuperar datos y no está asociado a ningún servidor; cloud9, es un entorno de desarrollo integrado que permite escribir y depurar código en JavaScript, Python y PHP; DynamoDB, para bases de datos no relacionales; RDS, para bases de datos relacionales; entre otros [9].

AWS posee gran diversidad de capacidades de CPUs virtuales que suplen a las máquinas físicas, estas capacidades de CPUs se les conoce como tipos de instancias, Amazon cuenta con diferentes clases como lo son el modelo m5, que pueden llegar a poseer hasta 96 CPUs, memoria de 384 GB, anchos de banda de hasta 14000 Mbps, entre muchos otros tipos de instancias. Otra gran ventaja que posee Amazon es que sólo cobra por lo que se consume, dentro del tipo de instancia tipo t2, que son las instancias más económicas y de uso general, puede cobrar hasta 0.3712USD por hora o 270.98 USD por mes; este tipo t2 presenta un procesador intel Xeon de alta frecuencia de hasta 3.3 GHz, pero esto varía dependiendo del tipo de la instancia t2 que se use [10].

Herramientas de procesamiento

En la actualidad se emplean softwares como Matlab y Python para el desarrollo de aplicaciones en el manejo de imágenes y señales. Se tienen reportes del uso de la herramienta Matlab para el registro, procesamiento, reconocimiento y clasificación de palabras del lenguaje español mediante el análisis de las señales de voz de habla subvocal [4].

Adicional a esto, Python ha tomado mucho auge en el procesamiento de señales e imágenes, por ejemplo, el caso de un estudio de los movimientos oculares, en especial los sacádicos, para el diagnóstico de desórdenes neurológicos, estos movimientos son generados por dos centros neuronales independientes y se pueden representar en forma de pulso y en forma de escalón. El pulso y el escalón se pueden ser separados usando una técnica llamada análisis de componente independiente (ICA), comprendiendo con profundidad los mecanismos de generación de los movimientos oculares sacádicos [11].

Debido a la problemática del manejo de Big data en el manejo de señales e imágenes se tienen fuentes en los que se exponen diversas contribuciones que se han ido desarrollando en el contexto de la enseñanza universitaria para carreras técnicas que facilitan la comprensión de los principales conceptos del procesamiento digital de señales y reconocimiento de patrones. Las prácticas se realizan bien en laboratorios virtuales mediante herramientas de simulación, de forma distribuida a través de internet y, en algunos casos, se han implementado en sistemas reales mediante DSP. La comprobación visual y en algunos casos auditiva del procesamiento aplicado permite la motivación del alumno, de igual modo reducir la brecha existente entre la teoría y la práctica real del procesamiento digital de señales [12]. Matlab posee una librería de tiempo de ejecución que permite trabajar en un ambiente virtual sin la necesidad de descargar el software, esta librería facilita el desarrollo de un laboratorio virtual para el procesamiento de imágenes digitales, está diseñada para mejorar y reconstruir imágenes de alta resolución, analizar algoritmos implementados, así como sus características ópticas. El desarrollo de laboratorios virtuales fortalece la evaluación experimental demostrando el buen desempeño de la herramienta-software virtual, aporta ventajas significativas, en comparación de otras herramientas, al lograr que los usuarios analicen y comprendan el conjunto de algoritmos utilizados en el procesamiento de imágenes y señales. Como parte del aprendizaje logrado con ayuda de los laboratorios virtuales se puede mencionar que facilita comprender que no hay una solución única al proceso de reconstrucción y restauración de imágenes y señales, sino que se tiene una variedad de algoritmos. Al modificar los parámetros de cada algoritmo y evaluar el desempeño cuantitativo y cualitativo de éstos se puede concluir de manera efectiva las ventajas y limitantes de cada uno, en un aprendizaje activo, lo cual permite fijar el conocimiento con base en la experiencia. También puede hacer inferencias en aspectos importantes como el costo computacional y el rendimiento de cada algoritmo [13].

Metodología

Debido a ciertas limitaciones que se tenía por tener una cuenta de capa gratuita, No fue posible el consumo de todos los servicios de AWS; por lo que fue necesario realizar una búsqueda bibliográfica con el fin de lograr los objetivos del proyecto de la mejor forma posible con los recursos disponibles, dando como resultado:

Para la ejecución de algoritmos en el procesamiento de imágenes y señales, se optó por consumir instancias en EC2, donde fue necesario implementar un protocolo de comunicación SSH para conectarse con los recursos de la instancia; la cual se le instaló Docker y se montó un contenedor con los recursos necesarios. Dentro de los muchos tipos de instancias que posee Amazon, se encuentran las de tipo C5; las cuales poseen características optimizadas para trabajos de computación intensiva y ofrecen un alto

rendimiento económico. Incluyen procesadores Intel Xeon personalizados de segunda generación con una frecuencia de núcleo individual de 3.9 GHz; poseen un número de CPUs que varía desde 2 hasta 96 CPUs, una memoria que puede alcanzar los 192 GiB y diferentes parámetros que se deben analizar dependiendo del tipo de aplicación que se quiera ejecutar ya que no todas las instancias funcionan para el mismo propósito [8]. Particularmente, para el proyecto se usó una instancia c5d.2xlarge que posee 8 CPUs virtuales, 16 Gb de memoria y un ancho de banda de red de hasta 10 Gbps.
Con el servicio de EC2 ejecutándose en nube, se procedió a implementar un algoritmo para el procesamiento de imágenes donde se usó un proyecto llamado Nipype. Para este servicio se consumió una instancia c5d.2xlarge.

Adicional a Nipype, también se implementó un algoritmo en Jupyter, con una imagen Docker conectada a una instancia en EC2 de igual tipo a la implementada en Nipype, donde se desarrolló la función ICA para la separación de los componentes independientes de señales electroencefalográficas (EEG). Para este algoritmo se usaron señales en formato .mat, de aproximadamente 64 sensores y entre 200.000 y 800.000 puntos*épocas. Python contiene una librería para calcular el ICA de una señal llamada ICA1, solo bastó con importarla y realizar la ejecución en Jupyter.

Finalmente, se desarrolló un algoritmo en Machine Learning donde se implementó una red neuronal convolucional con Tensor Flow para la clasificación de imágenes de muestras de sangre para determinar si se padecía o no la enfermedad de la Malaria, esta red se desarrolló con dos capas neuronales, 20 épocas, 1000 pasos y 300 pasos de validación. Para el set de datos se contó con 22009 imágenes para el entrenamiento; separadas entre 11010 para las imágenes infectadas y 10999 para las no infectadas. Para el set de datos de validación se tenían 5513, divididas en 2752 y 2761 imágenes para las infectadas y no infectadas; respectivamente [14] [15]. Esta red fue implementada en Python-Jupyter y luego fue subida a un servicio en AWS llamado SageMaker que facilita un entorno Jupyter para implementar código en Python con las configuraciones necesarias para Machine Learning. AWS SageMaker se puede desplegar fácilmente desde la consola de Amazon accediendo al servicio y simplemente configurando el tipo de instancias dependiendo de la carga computacional que el algoritmo requiera, para este modelo en Machine Learning se utilizó una instancia tipo ml.c5.4xlarge.

Resultados y análisis

Procesamiento de imágenes con Nipype:

Nipype provee interfaces de software para neuroimagen y facilita las interacciones con simples flujos de trabajo. Nipype provee un ambiente que

favorece la exploración de algoritmos de diferentes paquetes como ANTS, SPM, FSL, FreeSurfer, entre otros. Para comprobar la efectividad de los servicios implementados en AWS se desarrolló un pipeline o flujo de trabajo para obtener la imagen por resonancia magnética funcional (FMRI) con Nipype en un ambiente Jupyter desplegado como imagen en un contenedor Docker el cual contenía los siguientes nodos: Drop Dummy Scans (FSL): Se utiliza para no se ha alcanzado un estado de estabilidad en la toma de la imagen y elimina los escaneos fallidos, y así obtener regiones de interes; Slice Time Correction (SPM): Esta función corrige las diferencias en los tiempos de adquisición de cortes. Esta rutina está destinada a corregir el orden escalonado de adquisición que se utiliza durante el escaneo; Motion Correction (SPM): Herramienta para corrección de movimiento en el escaner; Artifact Detection: Utiliza parámetros de intensidad y movimiento para inferir valores atípicos, Segmentation (SPM): Separa imágenes estructurales en diferentes clases de tejidos. Crea mapas de probabilidad para la materia gris; Coregistration (FSL): Se asegura de que las imágenes funcionales estén registradas conjuntamente con la imagen anatómica; Smoothing (FSL): Se aplica para la reducción de ruido usando filtros no lineales, Apply Binary Mask (FSL): Aislar partes aisladas de una imagen para futuros análisis; y Remove Linear Trends (Nipype): Elimina tendencias cuadráticas y lineales en las imágenes suavizadas, se utiliza principalmente para obtener características especiales eliminando lo que no se desea de una imagen [16].

Dentro de los resultados en el algoritmo se obtuvo un pipeline que genera un procesamiento básico para FMRI, este flujo se ejecutó para únicamente una imagen en formato .nii. Cada nodo del pipeline generaba una imagen y datos nuevos dependiendo de la funcionalidad del nodo, como es el caso de la figura 1 donde se evidencia la corrección que se obtuvo para los movimientos del paciente cuando este se movía durante la toma de la resonancia. En esta figura se observa cómo resultan los datos después de haber aplicado corrección de movimiento, suavizado de la imagen y un filtrado de tendencias dentro de la imagen; esto permite obtener una un resultado más limpio, con menos ruido y sin tendencias que estropeen la recolección información necesaria para el análisis de la misma.

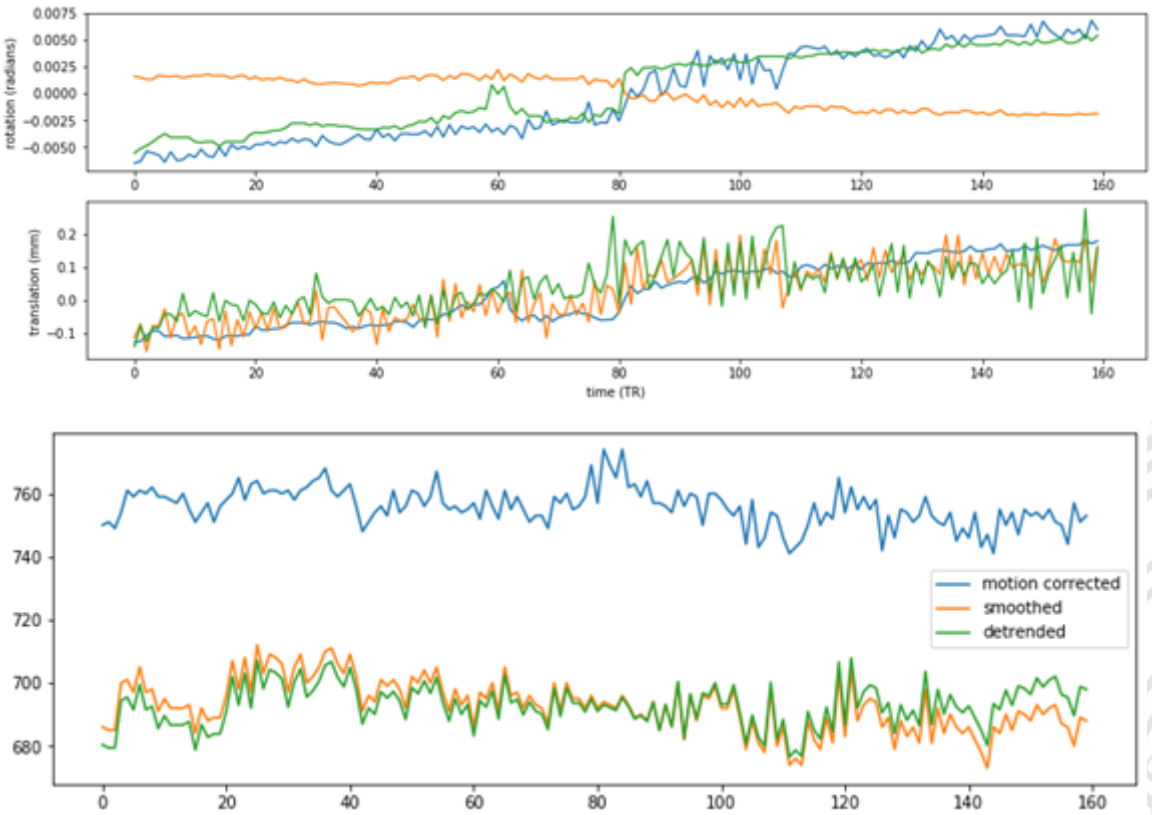

Figura 1. Implementación de corrección de movimiento, suavizado y eliminación de tendencias

Adicional a esto se pueden ver en las figuras 2 y 3 el resultado de la imagen al realizar una segmentación y la aplicación de máscara binaria para seleccionar partes atípicas de la imagen, la segmentación permite separar la imagen en diferentes planos sobre un mismo eje (eje Z) y con la ayuda de la máscara realizar un mapeo de un área específica de la segmentación.

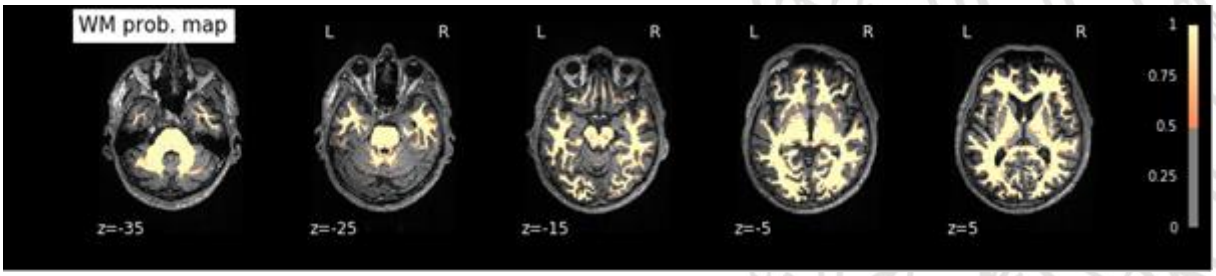

Figura 2: Resultado del nodo de segmentación

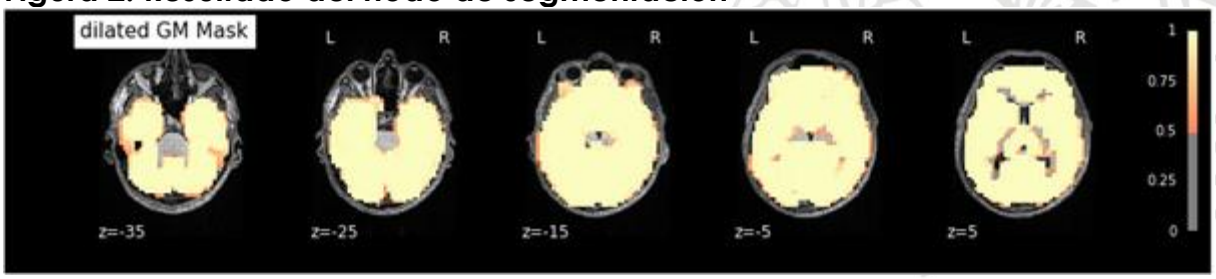

Figura 3: Resultado de aplicar máscara binaria.

Finalmente, en las figuras 4 y 5, para las imágenes funcionales, se puede observar los resultados obtenido en los nodos de suavizado y la imagen resultante de todo el proceso del pipeline donde se eliminan tendencias cuadráticas y lineales; esto con el fin de eliminar señales de baja frecuencias que no permiten destacar regiones del cerebro que poco se activan.

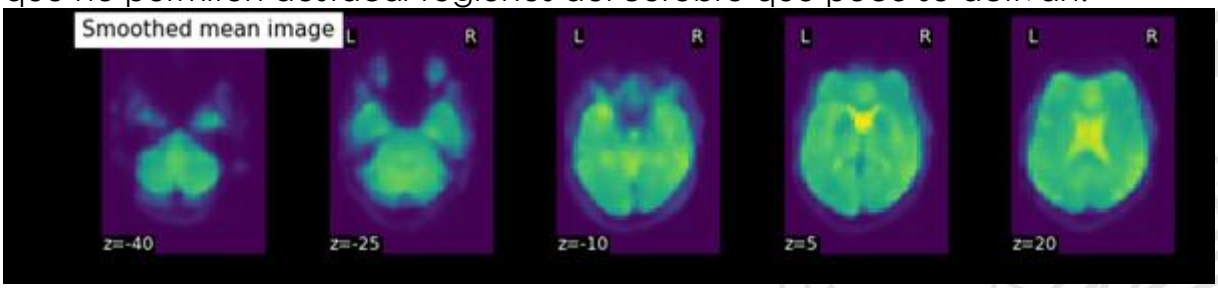

Figura 4: Resultado imagen suavizada

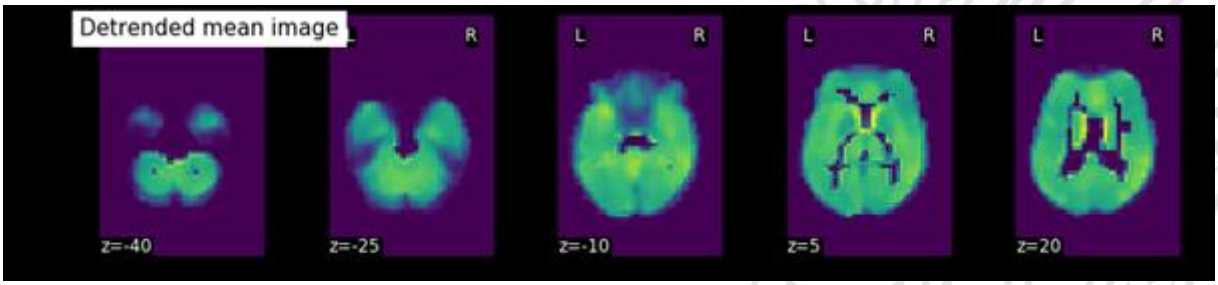

Figura 5. Resultado de la eliminación de tendencias de la imagen

AWS presenta ciertas ventajas con respecto al rendimiento y velocidad en máquinas locales, un ejemplo de esto es el resultado que se obtuvo al aplicar este algoritmo de Nipype en local y hacer lo mismo en nube para evaluar su comportamiento; dando como resultado la reducción de 1 hora, 12 minutos, 50 segundos; duración en local, a 7 minutos con 6 segundos; duración en la nube. Esto es un rendimiento importante ya que al momento de implementar un pipeline mucho más complejo y con diferente cantidad de imágenes, el procesamiento en AWS será mucho más rápido y eficiente que en una máquina local. Eso se vio reflejado al correr ambos códigos en los diferentes ambientes dando una disminución de tiempo en nube con la instancia seleccionada. Sin embargo cabe destacar que la instancia c5d.2xlarge no es la mejor instancia que existe en el servicio de EC2 para este tipo de algoritmos, su tipo C5 es el indicado pero existen instancias mucho más grandes e incluso con una unidad de procesamiento gráfico (GPU), pero la capa gratuita que fue la usada para este proyecto, no tiene toda la capacidad de los tipos de instancias; lo que quiere decir que al usar una instancia mucho más grande o con GPU el tiempo de ejecución se reduciría aún más.

Procesamiento de señales EEG con ICA:

El ICA es un método computacional que sirve para separar una señal multivariante en subcomponentes aditivos suponiendo que la señal de origen

tiene una independencia estadística y es no-Gausiana; esta función implementada en Python devuelve un parámetro importante y es un vector de las fuentes independientes de la señal EEG [17].

El algoritmo donde se implementaba el ICA dio como resultado la separación de los componentes de una señal generada aleatoriamente (ver figura 6), este resultado se muestra únicamente para fines ilustrativos ya que el conjunto de datos EEG es anónimo por lo cual no se tiene mucha información del comportamiento del sistema. No obstante, se puedo evidenciar que el algoritmo genera el arreglo donde contiene la señal dividida en sus componentes. En la misma figura, la primera imagen se tiene la señal fuente o la mezcla, la segunda imagen, vendría siendo la señal real separa entre sus componentes y finalmente, la tercera imagen es el resultado del ICA; esto permite concluir que se logró separar una señal mezclada entre sus diferentes componentes.
Se puede utilizar el servicio en AWS para cualquier señal donde el investigador pueda subir cualquier código de su preferencia y el servicio respondería de la misma manera.

Los resultados del servicio no fueron lo más prometedores en cuanto a tiempo debido a que el tipo de instancia no era la ideal y la capa gratuita no permitía el uso de otro tipo; sin embargo, se logró desplegar el servicio con la imagen Docker en el cual es útil para procesar cualquier tipo de señales. Si hubo una reducción del tiempo de ejecución del algoritmo donde se aplicaba el ICA; dando una disminución de 14 minutos, 46 segundo a 10 minutos 20 segundos. Esta reducción de tiempo podrá no parecer mucho pero también hay que considerar la parte de rendimiento de Hardware de una máquina local, ya que se evidenció que, corriendo este tipo de códigos en el computador con un procesador Intel Core I7 y velocidad de procesamiento de 2,10 GHz, 6 GB de memoria RAM y 1 Tera de almacenamiento, este presentaba una considerable reducción de respuesta en sus otros procesos dejando imposibilitado la continuación de otras tareas. AWS presta todos los insumos para que los recursos del computador no se vean comprometidos y sus demás procesos se ejecuten con normalidad, lo único necesario es tener una conexión a internet para poder ejecutar todos los servicios que Amazon Web Services posee.

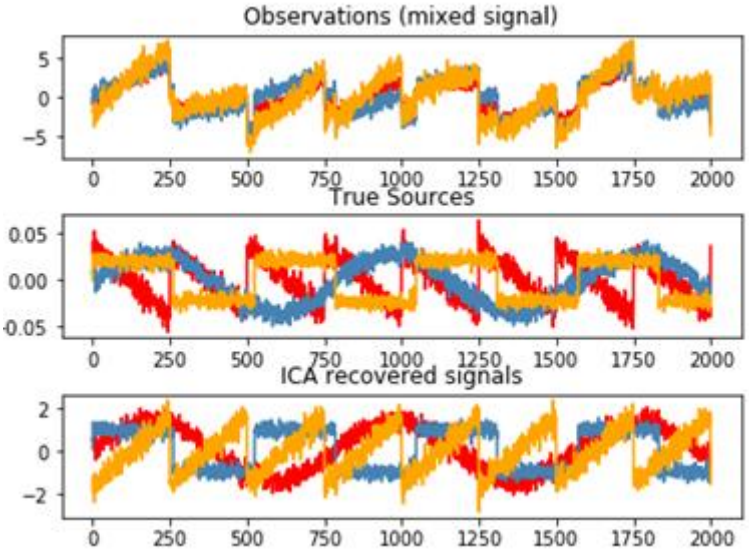

Figura 6: Resultado ICA

Clasificación de imágenes con Machine Learning:

Finalmente, los resultados con la red neuronal convolucional fueron los esperados. El entrenamiento dio como resultado una exactitud del 94,7% en un tiempo de ejecución de 9 horas, 35 minutos y 43 segundo en local, se logró obtener un modelo que clasificaba imágenes y determinar si se padece o no la enfermedad de la Malaria, este modelo recibe una imagen donde se evidencia la enfermedad y otra donde no, dando como respuesta infección o no infección (Ver figura 7 y 8). SageMaker dio como resultados un tiempo de ejecución de 53 minutos, 15 segundo; este servicio dentro de la capa gratuita si permite utilizar instancias con mejores características, pero aun así no permite consumir instancias con GPU, se concluye que al utilizar una instancia con GPU el tiempo de ejecución de un entrenamiento de un algoritmo en Machine Learning se vea reducido aún más.

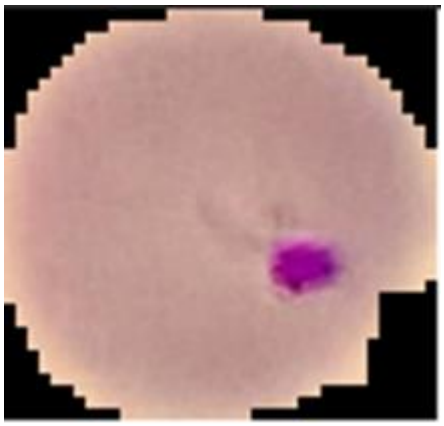

Figura 7: Resultado modelo resultante del entrenamiento para imagen infectada dando como resultado "Sujeto infectado"

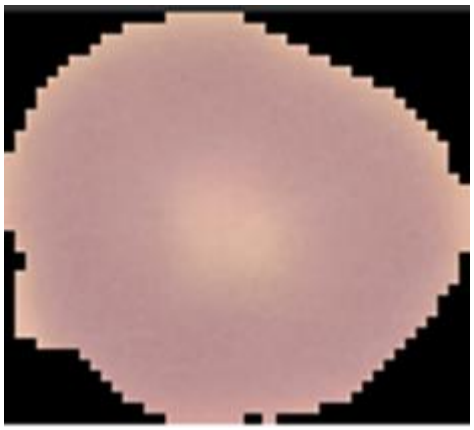

Figura 8: Resultado modelo resultante del entrenamiento para imagen no infectada dando como resultado "Sujeto no infectado"

Conclusiones

● Una de las características de Amazon Web Services es el poder soportar varios de los lenguajes de programación que se utilizan actualmente para el análisis de señales e imágenes biomédicas, muchos de los algoritmo que se implementan actualmente para la investigación son realizados en lenguajes como Matlab, que a pesar de que es una poderosa herramienta, no es libre y no está soportado por AWS, con el fin de dar solución a esta situación y poder consumir los recursos de Amazon, se implementaron tres algoritmos en Python para recurrir a EC2 y desplegar el procesamiento de imágenes con Nipype, análisis de componentes independientes para señales EEG y clasificación de imágenes de sangre para determinar si se padece o no la enfermedad de la Malaria.

● Los algoritmos en Machine Learning y más que todo las redes neuronales, son algoritmos que consumen muchos recursos y tiempo de procesamiento cuando se ejecutan en máquinas locales, es por esto que con la ayuda de AWS se logra reducir el tiempo de procesamiento y disminución de recursos que estos algoritmos implican, SageMaker es un servicio que logra todos estos beneficios proporcionando un ambiente en Jupyter Notebook y consumir tipos de instancias que sean necesarias para suplir las demandas del procesamiento en Machine Learning.

● Amazon contienen múltiples servicios que se pueden utilizar para disminuir tiempo de recursos de procesamiento, sin embargo, debido a las limitaciones que se tenía en la capa gratuita en la cual se basa este proyecto, una arquitectura mucho más compleja para el despliegue de servicios en el procesamiento de imágenes y señales biomédicas se vio limitado para sacarle el máximo provecho a AWS. No obstante, se pudo demostrar que, a pesar de las limitaciones que se tenían por el tipo de cuenta, AWS es una buena opción para invertir en recursos que disminuyan la brecha que existe con el aprendizaje y las nuevas tecnologías que existen actualmente, ya que muchas de ellas se ven limitadas por los recursos que

las máquinas locales poseen y en caso contrario, para poseer una máquina con un buen hardware se necesita la inversión de mucho capital. Dependiendo de los recursos que se usen en AWS, este cobraría por cantidad de ejecución del servicio o por cantidad de peticiones que se hagan a una API, por ejemplo. Para el despliegue de servicios como SageMaker y EC2, el cobro total por los 4 meses que duró la ejecución del proyector Amazon cobró 15.16 dólares. Se debe tener en cuenta que mucho de este cobro se debió a ensayo y error para determinar qué tipo de instancia en EC2 presentaría un mejor resultado para los objetivos planteados, con esto se concluye que el cobro de los servicios pudo haber sido menor ya que si se tiene el conocimiento de lo que se requiere usar, un algoritmo sería ejecutado una vez, se analizan los resultados y Amazon reduciría los tiempos de cobro del servicio. Es importante tener en cuenta que, para fines investigativos, una cuenta en AWS requiere de consumo de servicios por varias personas, lo que podría requerir de más inversión en algunos casos dependiendo del servicio. Sin embargo, EC2; que es un servicio únicamente de instancias, AWS seguirá cobrando únicamente por el tiempo de ejecución que la instancia permanezca encendida sin importar el número de personas que lo consuman.

● Las tecnologías actuales para el desarrollo de algoritmos de procesamiento de imágenes y señales biomédicas se pueden implementar de igual manera que se implementan en ejecuciones en nube, la única diferencia está en las capacidades del equipo donde se realicen y ejecuten estos algoritmos. Gracias a AWS los tiempos de respuesta de los algoritmos y el consumo de recursos de los computadores locales fueron reducidos considerablemente, favoreciendo la optimización de recursos, invirtiendo la diferencia de tiempo en otras tareas que sean necesarias para la investigación y así favorecer el aprendizaje de las nuevas tecnologías que existen actualmente dentro de instituciones educativas. Estos servicios desplegados en nube serían de gran utilidad en universidades donde se requiera aprender sobre nuevas técnicas de procesamiento en el área de Bioingeniería, ya que en muchas ocasiones no se cuenta con los recursos físicos para la enseñanza de estos nuevos desarrollos en tecnología, con ayuda de AWS se podría solucionar este problema y seguir avanzando en el área de la investigación y desarrollo para mejor la calidad de vida de las personas.

Referencias Bibliográficas

[1] S. Guerrero, "FINANCIACIÓN Y GOBIERNO DE LAS UNIVERSIDADES PÚBLICAS," pp. 1–38, 2011.

[2] C. Iii, "Procesamiento de imágenes capitulo iii 33," pp. 33–48.

[3] Catarina, "Capítulo 1. teoría de procesamiento de imágenes. 1.1.," pp. 7–29, 2007.

[4] L. E. Mendoza, "Provenientes del Habla Subvocal usando Wavelet Packet y Redes Neuronales Speech Subvocal Signal Processing using Packet Wavelet and Neuronal Network," pp. 655–667, 2013.
[5] FreeSurfer, "FreeSurfer System Requirements," 2018. [Online]. Available: https://surfer.nmr.mgh.harvard.edu/fswiki/DownloadAndInstall.
[6] A. Luis and J. Aguilar, "La Computación en Nube (Cloud Computing) : El nuevo paradigma tecnológico para empresas y organizaciones en la Sociedad del Conocimiento," pp. 95–112, 2009.
[7] J. T. Ardura, "Mayores en la nube," pp. 1–10, 2013.
[8] Google, "Productor y servicios," 2018. [Online]. Available: https://cloud.google.com/products/?hl=es.
[9] Amazon, "Explore our products," 2018. [Online]. Available: https://aws.amazon.com/products/.
[10] Amazon, "Tipos de instancias de Amazon EC2," 2018. [Online]. Available: https://aws.amazon.com/es/ec2/instance-types/.
[11] A. A. Fernández-higuera, C. E. Velázquez-rodríguez, R. A. Becerra-garcía, R. V García-bermúdez, G. Joya-caparrós, and M. Velázquez-mariño, "Aplicaciones del Análisis de Componentes Independientes al procesamiento de registros de movimientos oculares sacádicos Aplicaciones del Análisis de Componentes Independientes al procesamiento de registros de movimientos oculares sacádicos," no. May 2015, 2014.
[12] M. S. Peñas and G. F. Castro, "Laboratorios virtuales de procesamiento de señales," no. August, 2015.
[13] A. C. Atoche, "Virtual Laboratory for Digital," no. February, 2015.
[14] M. Poostchi, K. Silamut, R. J. Maude, and S. Jaeger, "Image analysis and machine learning for detecting malaria," *Transl. Res.*, vol. 194, pp. 36–55, 2016.
[15] R. López, "TensorFlow y Redes Neuronales," 2016. [Online]. Available: https://relopezbriega.github.io/blog/2016/06/05/tensorflow-y-redes-neuronales/.
[16] Nipype, "Nipype: Neuroimaging in Python Pipelines and Interfaces.," 2009. [Online]. Available: https://nipype.readthedocs.io/en/latest/.
[17] Alvarouc, "Independent Component Analysis," 2018. [Online]. Available: https://github.com/alvarouc/ica.