BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

"Una arquitectura de la red neuronal convolucional para el reconocimiento de tuberculosis en imágenes."

Presenta:

Brayan Chavez Benavides

Tesis presentada para obtener el título de:

Licenciatura en Ingeniería en Ciencias de la Computación

Dirigida por:

Dr. Rafael Lemuz López

Puebla, Puebla Enero 2020

Dedicatoria:

A mis padres por su apoyo, dedicación y sacrificio. Los logros obtenidos no son propios son nuestros.

A mi hija por ser el motor de mi vida.

A mi familia por su unidad y cariño.

Contenido

Resumen

En esta tesis se evalúan diferentes arquitecturas de redes neuronales convolucionales para resolver el problema de clasificación del tipo de tuberculosis utilizando imágenes de tomografías computarizadas (CT).

La tuberculosis es una enfermedad infecciosa, provocada por un bacilo, que se transmite a través del aire y que se caracteriza por la formación de tubérculos o nódulos en los tejidos infectados; puede afectar a diferentes órganos del cuerpo, en especial a los pulmones, produciendo tos seca, fiebre, expectoraciones sanguinolentas y pérdida de peso. Los desplazamientos de población (viajeros, refugiados, personas sin hogar en países industrializados) han contribuido significativamente en los últimos 40 años a la propagación de la enfermedad en el planeta.

En el trabajo se evalúan cinco arquitecturas de redes neuronales convolucionales comparando su desempeño en la tarea de clasificación utilizando una base de imágenes pública. Los resultados de las diferentes arquitecturas y estrategias de evaluación se presentarán en tablas que muestren el desempeño de cada arquitectura.

Los resultados experimentales muestran que es posible alcanzar hasta un 70 por ciento de clasificación del tipo de tuberculosis cuando se utiliza una estrategia simple de reducción de dominio que analiza solo las imágenes centrales de la tomografía de los pacientes.

Según la Organización Mundial de la Salud , cada día mueren casi 4,500 personas a causa de la TB y aproximadamente 30,000 personas contraen esta enfermedad prevenible y curable. No obstante, reconoce que los esfuerzos por luchar contra la TB han salvado 54 millones de vidas desde el año 2000 y han reducido la tasa de mortalidad en un 42%.

En tanto, el Centro Nacional de Programas Preventivos y Control de Enfermedades (CENAPRECE) reporta que en el año 2016 se registraron 21,184 nuevos casos de tuberculosis a nivel nacional. De acuerdo con la Secretaría de Salud (SSA), en México “más de la mitad de todos los municipios notifica casos de tuberculosis cada año; sin embargo, las entidades federativas de mayor número de nuevos casos y muertes por esta causa son: Baja California, Veracruz, Guerrero, Sonora, Tamaulipas, Chiapas, Nuevo León y Tabasco. Datos que reflejan la importancia de la enfermedad y evidencian su aumento. [INSP19]

Introducción

Según estimaciones del informe anual sobre tuberculosis de la OMS [OMS16] presentado en 2016, hubo 1,4 millones de muertes debidas a la tuberculosis en 2015. En 2015 la tuberculosis fue una de las 10 principales causas de muerte [OMS16]. Otros 0.4 millones de tuberculosis produjeron muertes entre personas VIH positivas. 10,4 millones de nuevos casos de tuberculosis se presentaron en 2015. De estos, 56% eran hombres, 34% mujeres y 10% eran niños Los estudios realizados antes de que estuviera disponible la medicación mostraron que hasta un 70% de las víctimas de tuberculosis murieron en los próximos 10 años.

Gracias a un diagnóstico rápido y un tratamiento adecuado, la mayoría de los casos de tuberculosis pueden curarse. Se informaron tasas de éxito de al menos el 85% en los casos en que la tuberculosis no es farmacológicamente resistente.

Dado que la tuberculosis está tan extendida y se puede tratar bien, se convirtió en uno de los objetivos de Desarrollo (ODS) [Uni16] de 2030, emitidos por las Naciones Unidas para poner fin la epidemia de la tuberculosis. Para cumplir con este objetivo, la OMS desarrolló la Estrategia de Fin de la TB que se documenta en su Informe mundial sobre la tuberculosis [OMS16] de 2016. Su objetivo es reducir las muertes causadas por la tuberculosis en un 90% del 2015 al 2030.

Una de las partes principales de esta misión es el diagnóstico temprano de la tuberculosis en muchas personas. Por lo tanto, es útil desarrollar un método que pueda realizar una clasificación rápida de radiografías pulmonares en pulmones sanos y presencia de rasgos distintivos de la enfermedad. Este programa puede ser utilizado en exámenes de rayos X móviles en áreas muy afectadas por la tuberculosis. Esto puede permitir a los pacientes recibir tratamiento en una etapa más temprana de la enfermedad, lo que aumenta la posibilidad de curar la tuberculosis.

En esta tesis se estudian cinco diferentes arquitecturas de Redes Neuronales Convolucionales (CNN) en la tarea de clasificación de la tuberculosis pulmonar de tomografías de tórax. El objetivo es descubrir cuál de estas arquitecturas proporciona el mejor resultado en la tarea de clasificación del tipo de tuberculosis en radiografías de tórax. Nuestro estudio se centra en la tuberculosis pulmonar, pero existen diferentes formas, que afectan a otros órganos además de los pulmones. Estas formas se resumen en el término tuberculosis extrapulmonar [OMS16].

La importancia de las CNN radica en el entrenamiento basado en ejemplos donde el desarrollador no tiene la necesidad de extraer las características más importantes para cada clase ya que las CNN son capaces de extraer dichas características aplicando diversos filtros de convolución cuyos parámetros se ajustan en un proceso de optimización.

El aumento del poder de cómputo de los procesadores y las tarjetas gráficas modernas han permitido que estudiosos del área de inteligencia artificial sean capaces de implementar redes cada vez más complejas y eficientes; no es de extrañar que en la rama médica existan estudios basados en el uso de redes neuronales convolucionales para la clasificación y detección de enfermedades.

En cuando a su aplicación en el ámbito general las CNN pueden ser usadas para clasificar cualquier objeto e incluso ser entrenadas mediante algoritmos sofisticados para detectar objetos en imágenes. Algunas aplicaciones interesantes de las CNNs son:

- Detección de caracteres en imágenes para transcripción automática de textos
- Clasificación y detección de objetos en imágenes médicas
- Clasificación y detección para toma de decisiones en el ámbito de la industria alimenticia (selección y ordenamiento de productos)
- Sistemas de apoyo para la conducción de vehículos, etc.

La motivación de esta tesis es la de aplicar el conocimiento de las CNN que están pre entrenadas en otros ámbitos y poder incluirlas en una nueva herramienta para que se puedan aplicar en áreas como la medicina moderna con la intensión de brindar al usuario información más certera.

Planteamiento del problema

El objetivo de esta tesis es evaluar arquitecturas de redes neuronales convolucionales para clasificar inicialmente las tomografías de tórax como tomografías con pulmones sanos o con rastros de tuberculosis, basados únicamente en las imágenes de las tomografías. Además, de clasificar los diferentes tipos de tuberculosis pulmonar en función de los datos disponibles en una base de imágenes pública. Para lograr los mejores resultados posibles se plantea entrenar y evaluar diferentes arquitecturas de aprendizaje automático.

Enfoque metodológico.

El enfoque metodológico de esta tesis es aplicar un método simple de reducción de dominio previo a la utilización de redes convolucionales para obtener el mejor resultado en términos de clasificación de la tuberculosis en las tomografías de tórax.

Queremos evaluar el desempeño de los métodos recientes de aprendizaje profundo en tareas de clasificación con imágenes de tomografías de torax para identificar uno de los 5 tipos de tuberculosis.

La información que describe la base de datos utilizada se puede ver en la sección 3.2 del capítulo 3.

Estructura del trabajo

En el capitulo 1 se presenta la motivación del trabajo de tesis y se describe brevemente la tuberculosis pulmonar.

El capítulo 2 cubre el estado actual de la técnica en ingeniería de características y enfoques de redes neuronales convolucionales. La mayoría de las publicaciones discutidas tienen el mismo objetivo que esta tesis, a saber, la clasificación de la tuberculosis pulmonar utilizando imágenes de tórax.

En el capítulo 3, se puede encontrar una breve introducción a los conjuntos de datos importantes relevantes para esta tesis. Se describe el conjunto de datos de TBC disponible para esta tesis para realizar la clasificación. Además, describimos el gran conjunto de datos ImageNet etiquetado que hizo posible utilizar redes profundas pre-entrenadas con pequeños conjuntos de datos.

En el capítulo 4, se describen brevemente los conceptos necesarios para los tres enfoques implementados de esta tesis. Se trata de las redes neuronales convolucionales. Además, se describen algunas de las arquitecturas de redes neuronales convolucionales más populares y exitosas reportadas en la literatura.

En el capítulo 5 se describe la tecnología de hardware y software utilizado en esta tesis. Además, se explica la implementación de tres estrategias básicas de reducción de información. Cada enfoque está estructurado en los pasos de preprocesamiento, entrenamiento y evaluación de la clasificación. Finalmente se describen los resultados de los experimentos realizados.

En el capítulo 6, se presentan las conclusiones y el trabajo futuro de la tesis.

Capítulo 1 TBC

1.1 Tuberculosis pulmonar (TBC)

La tuberculosis es una enfermedad con un agente infeccioso [Leu99] causada por el bacilo de la tuberculosis. También se llama Mycobacterium tuberculosis. La transferencia del bacilo se realiza por infección por gotitas de saliva a través del aire que se expectora al toser los pacientes. Un paciente con TB no tratado puede infectar un promedio de 10 a 15 personas cada año. La epidemia de TBC es más frecuente en los países en desarrollo causada por las pobres condiciones sanitarias en comparación con los países del primer mundo [Leu99].

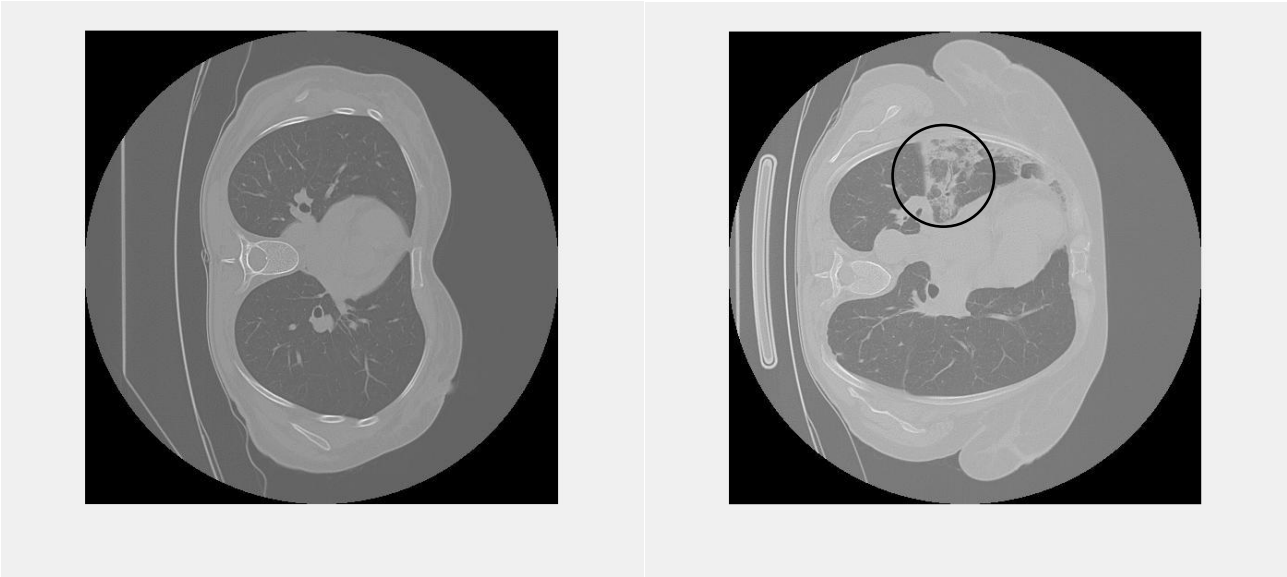


Figura 1.1: Tomografías de tórax que muestran un paciente sin TBC (izquierda) y un paciente con TBC (derecha). En la radiografía derecha, las manifestaciones de TBC son visibles en la parte

superior. Región del pulmón derecho como opacidades nodulares dentro del círculo.

Según lo descrito por la OMS [OMS16] en 2015, el 60% de los nuevos incidentes de TBC aparecieron en solo seis países (India, Indonesia, China, Nigeria, Pakistán, Sudáfrica). Para diagnosticar TBC bacteriológicamente existen las siguientes pruebas [WHO16]:

• Microscopía de esputo: Se están analizando muestras de esputo para detectar bacterias bajo un microscopio. Esta prueba solo funciona para el TBC pulmonar.

• Prueba Molecular Rápida: La prueba molecular rápida da mejores resultados que la microscopía de frotis de esputo. Esta prueba se recomienda por la OMS para niños y adultos para diagnosticar TBC pulmonar y algunos tipos de TBC extrapulmonares.

• Métodos de cultivo: La OMS define esta prueba como el estándar de referencia actual. La desventaja, sin embargo, radica en la duración de la evaluación, que dura hasta 12 semanas.

También hay otras pruebas para detectar TBC farmacorresistente. Leung [Leu99] describe que las radiografías de tórax facilitan el diagnóstico de TBC activo o anterior. Pero la diferencia de estas formas solo se puede diagnosticar con el paso del tiempo de las radiografías. Si no hay cambios en las imágenes durante 4 a 6 meses, la infección de TBC puede considerarse inactiva. Las radiografías de tórax tienen la ventaja de que están disponibles rápidamente en comparación con el estándar de referencia definido por la OMS, que toma hasta 12 semanas para el análisis. Las manifestaciones de TBC en las radiografías se pueden detectar a tiempo y se puede iniciar el tratamiento.

En su informe sobre sobre el TBC pulmonar, Leung [Leu99] describe las diferencias de cuatro formas de TBC pulmonares en las radiografías de tórax según la edad, el estado inmunológico y el TBC activo anterior.

TBC primario

Las principales características de la enfermedad primaria son los ganglios linfáticos agrandados (linfadenopatía), que ocurren en el 83-96% de los casos en niños. El porcentaje disminuye para los pacientes mayores y puede llegar a un mínimo del 10-43%. La linfadenopatía puede ser visible en cualquier parte del pulmón, pero con mayor frecuencia se manifiesta en las estaciones derecha paratraqueal e hilar.

TBC postprimaria

Las más frecuentes y más características de la enfermedad posprimaria son las opacidades parenquimatosas en los lóbulos pulmonares superiores en 83-85% y en la parte superior de los lóbulos inferiores en 11-14%.

Los tuberculomas son lesiones redondas, bien separadas, con un tamaño de 0,5-4,0 cm, que aparecen en el 3-6% del parenquimatoso de pacientes con TBC postprimarias. Las cavitaciones son visibles en un 40-45%. Los ganglios linfáticos calcificados y las manifestaciones fibróticas con una tasa de aparición del 20-40% indican la presencia primaria de TBC.

TBC Miliar

En las radiografías, esta forma de TBC se manifiesta en muchos nódulos pequeños, de 1 a 3 mm, distribuidos en todo el pulmón, que están presentes en aproximadamente el 30% de los casos.

TBC en el síndrome de inmunodeficiencia adquirida

El progreso de la inmunosupresión de pacientes con VIH influye en las manifestaciones de TBC. Si la función inmunitaria sigue siendo casi normal, parece similar a las radiografías de TBC de pacientes sin VIH. En los pulmones de pacientes con VIH se manifiestan más linfadenopatías y menos cavitación.

Para prevenir la TBC, existe una vacuna [WHO16] que protege a los niños de la infección de algunas formas de TBC, la vacuna de Bacille-Calmette-Guérin, sin embargo, no tiene efecto en los adultos.

El tratamiento de un TBC activo existente solo es posible con medicamentos. El tratamiento actual recomendado por la OMS [WHO16] para el TBC que es susceptible a la medicación estándar se fija en 6 meses. Otras formas de TBC que son resistentes a una o más sustancias activas requieren un tratamiento más costoso, que toma alrededor de 9-12 meses.

Capítulo 2

Estado del arte

Este capítulo proporciona una breve descripción de la metodología relevante en esta tesis para identificar la tuberculosis en las tomografías de tórax. Dado que se han evaluado diferentes métodos, el capítulo está estructurado de acuerdo con los métodos utilizados. En primer lugar, se introduce la ingeniería de características, cuyo objetivo es proporcionar una perspectiva de la extracción explícita de características, como la detección de bordes y la detección de formas. En otro paso, se examinarán diferentes enfoques de aprendizaje profundo. La última parte de este capítulo ofrecerá una visión general del desempeño humano. Esta información nos permite hacer una comparación directa del rendimiento entre humanos experimentados y los diferentes métodos presentados.

2.1 Ingeniería de características

La mayoría de las publicaciones disponibles se concentran en áreas individuales de las radiografías de tórax, por ejemplo [AGM98], [vGSL06], [DPU09]. Aparte de estos trabajos, también hay algunas publicaciones que están particularmente relacionadas con la clasificación de las TBC.

El método de Jaeger. [JKC14] comienza con la segmentación del pulmón, utilizando corte de grafos y un modelo de pulmón predefinido. Los campos pulmonares resultantes se utilizan para la extracción de características en términos de intensidad, gradientes, forma, bordes y otras características. Luego, la clasificación de las características se realiza mediante una Máquina de vectores de soporte (consulte la sección 4.2.1). Al clasificar, la precisión está entre 78-82,5%, según el conjunto de datos utilizado.

Meléndez. [MvGM16] introdujo su enfoque sobre la clasificación de TBC en radiografías de tórax utilizando una combinación de aprendizaje de múltiples instancias y aprendizaje activo. Su enfoque comienza con la segmentación pulmonar y la extracción

de características de textura basadas en la distribución de intensidad. Después de la clasificación inicial del conjunto de entrenamiento, las imágenes más útiles en bolsas positivas se separan en regiones de imagen, que luego son clasificadas por un experto. Con su enfoque, alcanzan un área bajo la curva (AUC) de hasta el 88%.

La publicación de Hogeweg. [HSM + 15] comienza con la segmentación de los pulmones como Jaeger. [JKC + 14] y Meléndez. [MvGM + 16]. En su enfoque, se centran en el estudio den la irregularidad de las manifestaciones de TBC. Se utilizan diferentes subsistemas, que clasifican las características de forma y textura para obtener diferentes puntuaciones. La combinación de estos subsistemas para formar una puntuación total, es un intento de alcanzar una buena generalización. El mejor AUC alcanzado por esta combinación es del 86%.

2.2 Redes neuronales convolucionales

El avance de las CNN en la computación visual fue en 2012 con la llamada AlexNet de Krizhevsky. [KSG12]. Ganó el desafío de reconocimiento visual a gran escala de ImageNet 2012 (ILSVRC). Desde entonces, los investigadores intentaron desarrollar diferentes métodos para hacer uso de las CNN. Además de la oportunidad de construir su propia red, es posible hacer uso de las arquitecturas existentes (consulte la sección 4.2.2 para más detalles). Esta sección cubre los enfoques en redes no entrenadas, así como las redes entrenadas en el conjunto de datos de ImageNet (consulte la sección 3.2) que se ajustaron o usaron como un extractor de características fijas. Más detalles sobre estos métodos están disponibles en el Capítulo 4. Shin. [SRG + 16] comparó algunas redes diferentes, principalmente AlexNet [KSG12] y GoogLeNet [SLJ + 15] entre sí en la tarea de detección de enfermedad pulmonar intersticial en tomografías computarizadas. Se realizaron pruebas en las redes no entrenadas y el ajuste fino de las redes pre-entrenadas. Los datos de entrenamiento consisten en 905 cortes de 120 pacientes. Lograron una precisión del 74% en AlexNet y del 75% en GoogLeNet entrenando los modelos desde cero con este conjunto de datos. Afirmaron que con un

conjunto de datos tan pequeño es difícil entrenar grandes redes. Sus resultados logrados en el ajuste de las redes pre-entrenadas fueron para AlexNet con 86,7% de precisión y para GoogLeNet 90,2%. Además, el ajuste fino se realizó en OverFeat de Sermanet. [SEZ + 13] con una precisión del 87,7% y en VGG16 de Simonyan. [SZ15] con una precisión del 90%.

El enfoque de Hwang. [HKJK16] también se basa en AlexNet [KSG12] para el objetivo de la clasificación de TBC en las radiografías de tórax. También compararon la capacitación de la red no entrenada con el ajuste fino de la red pre-entrenada. Su conjunto de datos consta de 10848 imágenes, de las cuales 7020 son normales y 3828 tienen manifestaciones de TBC. Los datos se aumentaron mediante el recorte aleatorio de 520x520 imágenes a 500x500 de resolución y aplicando la transformación de reflejo. AlexNet está diseñado para recibir datos de entrada de tamaño 227x227, por lo que Hwang agregó una capa convolucional adicional para la extracción de características, para hacer uso de su tamaño de entrada de 500x500. Al entrenar la arquitectura AlexNet desde cero con pesos inicializados al azar, podrían alcanzar un AUC de 82,8% y una precisión de 78,8%. Con el ajuste fino, alcanzaron un AUC de 96,7% y una precisión de 90,5%. El papel de Lakhani. [LS17] vuelve a hacer una comparación de AlexNet [KSG12] y GoogLeNet [SLJ + 15], con pesos no entrenados y pesas pre entrenadas, para la clasificación de TBC en las radiografías de tórax. Tenían 1007 imágenes, de las cuales 150 se usaron para las pruebas (75 TBC positivas, 75 sanas), 685 para entrenamiento y 172 para validación. El aumento de datos se realizó cambiando el tamaño a 256x256, convirtiéndolo al formato de Gráficos de red portátiles, recorte aleatorio a tamaño 227x227, resta de la media y reflejo. Con aumento adicional de rotaciones de 90°, 180°, 270° y en el contraste, la ecualización adaptativa limitada del histograma pudo mejorar aún más sus modelos sin entrenamiento. Sus mejores resultados para redes no entrenadas fueron un AUC del 90% con AlexNet y un AUC de 88% con GoogLeNet. Un conjunto en la GoogLeNet y AlexNet mejor afinados y con mejor rendimiento alcanzó un AUC del 99%.

Otro método para hacer uso de las redes neuronales convolucionales es aplicar redes ya capacitadas como extractores de características fijas.

Shin [SRG16] utilizo este enfoque de redes pre-entrenadas utilizando AlexNet [KSG12]. Las características extraídas podrían clasificarse con una precisión del 76%. Razavian [RASC14] usó la red OverFeat [SEZ13] pre-entrenada en ImageNet [JWS09] para sus experimentos. Su conjunto de datos fue el conjunto de datos Pascal VOC 2007 [EEZ06], que consta de más de 10000 imágenes y 20 clases, que se aumentaron mediante el recorte y la rotación. La salida de la primera capa se usó como vector de características con dimensión de 4046 para la clasificación. La clasificación del vector de características se realizó utilizando una Máquina de vectores de soporte (consulte la sección 4.2.1). Su precisión promedio para todas las clases fue de 73,9% sin aumento de datos y 77,2% con aumento de datos.

La publicación de Ginneken [vGSJC15] utiliza las características extraídas de la red OverFeat para la detección de nódulos pulmonares en las tomografías computarizadas. Su conjunto de datos consta de 865 tomografías computarizadas con 1147 nódulos pulmonares. Al principio se extrajeron los nódulos candidatos utilizando un Sistema de detección de nódulos de última generación. Para cada candidato y para cada x, y, el eje z, las imágenes 2D se extrajeron de los escaneos y se redimensionaron a una resolución de 221x221. Estos parches se utilizaron como entrada para la red OverFeat y el vector de características dimensional resultante de 4096 se clasificó con una Máquina de vectores de soporte (consulte la sección 4.2.1). El sistema de detección de nódulos de última generación alcanzó una sensibilidad del 68% y podría mejorarse con el uso adicional de OverFeat como extractor de características al 71%.

2.3 Rendimiento humano

Para comprender mejor la calidad del desempeño de los diferentes enfoques de aprendizaje automático, presentamos los resultados del desempeño de humanos capacitados para leer radiografías en esta sección.

Jaeger [JKC14] se presentan los resultados de clasificación por dos radiólogos para una segunda y tercera lectura de las radiografías. Ambos radiólogos conocían la tarea de detección de TBC en las radiografías y leían las imágenes de forma independiente. La clasificación real de las imágenes se basó en datos clínicos y datos de pacientes, a los que los radiólogos no tuvieron acceso. Después de la primera lectura de las imágenes, los radiólogos coincidieron en el 84,8% de los casos. Para los casos en los que no estaban de acuerdo, decidieron sobre el estado del paciente juntos. Esta colaboración resultó en la detección de todos los casos positivos de TBC, por lo tanto, una sensibilidad del 100% y una especificidad del 68,8%.

Maduskar [MMA13] tenía 4 oficiales clínicos, que leían las radiografías de tórax.

Estos oficiales clínicos tenían un diploma de 3 años en medicina y están capacitados en la tarea de interpretar las radiografías de tórax. No se proporcionó información clínica de los pacientes.

Las imágenes se clasificaron con un puntaje entre 0 y 100, donde el puntaje expresa la confianza de los oficiales clínicos en el TBC activo. Una puntuación mayor a 50 se interpretó como una apariencia anormal. El AUC para los funcionarios clínicos fue del 89%, 90%, 91% y 92%, la sensibilidad del 86%, 85%, 83% y 96% y la especificidad del 88%, 76%, 85% y 46%.

Capítulo 3

Datos

Este capítulo proporciona una descripción general de los datos utilizados en esta tesis. Primero se describe el conjunto de datos TBC. Además, se presenta el conjunto de datos ImageNet [JWS09], porque es un conjunto de datos importante en términos de redes neuronales convolucionales. Gracias al gran tamaño del conjunto de datos, es posible entrenar adecuadamente las redes neuronales convolucionales. Estas redes se pueden reutilizar en otras tareas, con menos datos disponibles, como se describe en la sección 4.2.2.

3.1 Conjunto de datos TBC

El conjunto de datos utilizado en este trabajo ha sido proporcionado por la página de problemas para soluciones reales crowdAI.

Los datos se han anonimizado, por lo que del paciente únicamente se puede conocer la edad y el tipo de enfermedad para cada una de las tomografías presentes. Consiste en 1008 tomografías digitales contenidas en archivos de formato .NII. El conjunto de datos utilizado en esta tarea incluye tomografías computarizadas de tórax de pacientes con tuberculosis junto con el tipo de tuberculosis. Algunos pacientes incluyen más de una exploración. Todas las exploraciones que pertenecen al mismo paciente presentan el mismo tipo de TB.

El tamaño de las imágenes es de 696 x 625 pixeles. El número de imágenes por paciente varía de entre 110 imágenes a 140 para obtener en total por las 1008 tomografías se obtuvieron 131,794 imágenes distribuidas de la siguiente forma.

Tipos	Pacientes
Tipo 1 Infiltrativo	228 (376)
Tipo 2 Focal	210 (273)
Tipo 3 Tuberculoma	100 (154)
Tipo 4 Miliar	79 (106)
Tipo 5 Fibrocavernoso	60 (99)
Total, de pacientes	**677 (1008)**

Tabla 3.1. Número de pacientes de cada tipo de neumonía y entre paréntesis el número de imágenes disponibles

3.2 ImageNet

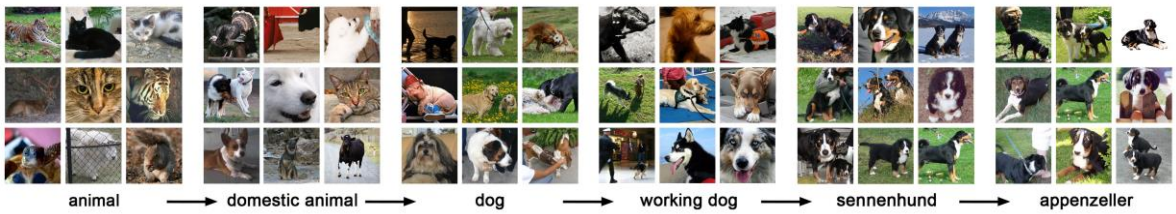


Figura 3.2: Las visualizaciones de una rama forman la raíz de la hoja de ImageNet. Para cada categoría, se presentan 9 imágenes muestreadas al azar.

El conjunto de datos de ImageNet [JWS09] es importante en términos de redes neuronales convolucionales.

Se compone de muchas imágenes naturales, que se utilizan para la formación de grandes redes neuronales convolucionales.

La estructura del conjunto de datos se basa en la estructura de WordNet [Fel98]. En la publicación de 2009 [JWS09] se afirma que el conjunto de datos consta de 3.2 millones

de imágenes en 12 subárboles con 5247 categorías diferentes. Cada categoría tiene 600 imágenes en promedio.

Los números actuales de la página de inicio de ImageNet [JWS09] indican que el conjunto de datos contiene 14,197,122 imágenes en 21841 categorías.

Para obtener una mayor diversidad, los objetos etiquetados en las imágenes se pueden ocluir y las imágenes pueden incluir el desorden de fondo y mostrar diferentes apariencias, posiciones, puntos de vista y poses.

La clasificación de las imágenes fue realizada por humanos con el uso del servicio de Amazon Mechanical Turk (AMT). Este servicio es una plataforma en línea, que paga a los usuarios por la finalización de las tareas proporcionadas. La precisión media en el etiquetado de las imágenes es del 99,7%.

Con la creación de este conjunto de datos, el desafío de reconocimiento visual a gran escala de ImageNet (ILSVRC) se introdujo en 2010 y se ha realizado anualmente desde entonces. Su objetivo es juzgar diferentes algoritmos para la localización de objetos y la detección de objetos.

En esta tesis, no utilizamos los datos de la imagen en sí, sino los pesos de las redes neuronales convolucionales formadas en este conjunto de datos.

En la figura 3.2, una rama de ejemplo se visualiza con imágenes seleccionadas al azar.

Capítulo 4

Metodología

Este capítulo describe los conceptos que son relevantes para esta tesis, comenzando con la descripción del método de ingeniería de características y las características extraídas en este trabajo.

Posteriormente se presentan Máquinas de vectores de soporte como técnica de clasificación. Luego se explican las redes neuronales convolucionales, así como diferentes métodos sobre cómo aplicar dichas redes. Adicionalmente se introducen diferentes arquitecturas conocidas.

Estas arquitecturas son AlexNet de Krizhevsky. [KSG12], GoogLeNet de Szegedy. [SLJ15], VGGNet de Simonyan. [SZ15] y ResNet de He. [HZRS16].

4.1 Clasificación

En esta sección se presentan dos enfoques para resolver el problema de clasificación.

El problema de clasificación es la tarea de encontrar la etiqueta correcta de un conjunto predefinido de categorías a una imagen de entrada. Queremos encontrar una función $f : X \mapsto Y$. De tal manera que las imágenes de entrada $x_i \in X, i = 1, \dots, N$ se asignen a las etiquetas correspondientes $y_i \in 1, \dots, M$.

En los siguientes párrafos se presentan la máquina de vectores de soporte y las redes neuronales convolucionales. Ambos son enfoques supervisados para resolver el problema de clasificación.

4.1.1 Máquinas de vectores soporte (SVM)

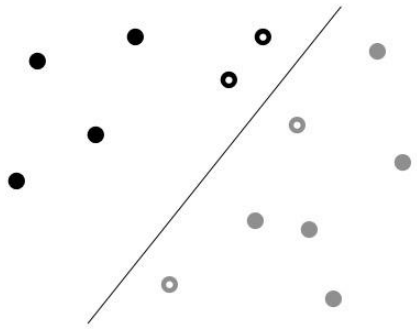


Figura 4.1: Conjunto de datos 2D separados por una línea que representa un clasificador lineal. Los vectores de soporte se visualizan con un punto blanco en el centro.

La Máquina de vectores de soporte [Bur98] es un algoritmo de aprendizaje supervisado para la clasificación de características.

La SVM lineal resuelve el problema de clasificación como una función lineal.

$$f(x) = X \cdot x + b \tag{4.1}$$

donde W es una matriz, conocida como ponderaciones, y b es un vector de sesgo. Para un problema de clasificación binaria, se aplica la función signo a la función lineal

$$y = f(x) = sgn(W \cdot x + b)$$

que conduce a un mapeo de la salida de 1 o −1. Esta salida puede interpretarse como etiquetas de clase.

Los SVM están buscando una separación de los datos de tal manera que la distancia entre las muestras de datos de las diferentes clases sea lo más grande posible. Para un SVM lineal, esto significa encontrar un hiperplano de separación, para un SVM no lineal el algoritmo está buscando una superficie de decisión.

El nombre de la SVM proviene de los llamados vectores de soporte, que son puntos de datos que influyen directamente en el límite de decisión. Si se elimina un vector de soporte, el límite de decisión resultante será diferente. Si todos los puntos de datos, excepto los vectores de soporte, se eliminan, el límite permanece igual. Los vectores de soporte tienen además la característica de que son los puntos de datos más cercanos al plano o superficie de separación.

Para obtener superficies de decisión no lineales, se introduce una función del núcleo [GBC16], que permite un cálculo más eficiente. Esto se hace reescribiendo la función lineal como:

$$W \cdot x + b = b + \sum_{i=1}^{m} aix^T x^{(i)}$$

Donde α es un vector de coeficientes con mayormente ceros. Resulta del aprendizaje de qué muestras de entrenamiento contribuyen a la superficie de decisión. Luego se reemplaza *x* con una función característica Φ(*x*). La función del kernel reemplaza el producto de puntos.

$$k\left(x, x^{(i)}\right) = \phi(x) \cdot \phi\left(x^{(i)}\right)$$

lo que resulta en la función

$$f(x) = b + \sum_{i} \alpha_i k(x, x^{(i)}).$$

En la práctica, los parámetros para los SVM deben estar predefinidos. Para encontrar los mejores parámetros se puede utilizar el método de optimización Grid Search. Para un conjunto dado de parámetros, el algoritmo de búsqueda de cuadrícula prueba todas las combinaciones posibles de parámetros en el conjunto de datos. Estas combinaciones se evalúan y se selecciona el mejor conjunto de parámetros para la clasificación.

4.1.2 Redes neuronales convolucionales (CNN)

Las redes neuronales convolucionales [LKJ] son un tipo especial de redes neuronales. Las redes neuronales consisten en capas apiladas, comenzando con una capa de entrada, varias capas ocultas y una capa de salida. La figura 4.2 muestra una visualización de esta estructura. La entrada es un solo vector. Una capa oculta está formada por múltiples neuronas donde cada neurona de una capa está conectada a cada neurona de la capa anterior, pero no hay conexiones entre las neuronas dentro de una capa. Una neurona tiene pesos y sesgos, como se ve en la función (4.1), que pueden entrenarse. La capa de salida es una capa totalmente conectada y genera las probabilidades de clase.

Las CNN suelen tomar imágenes RGB como entrada. Sin embargo, la entrada no necesariamente debe ser una imagen, solo necesita ser una matriz. Para procesar esta entrada, se utiliza la operación matemática de convolución. La estructura totalmente conectada de Redes neuronales está diseñada para procesar un solo vector [GBC16] y sería demasiado compleja para un enfoque en imágenes. Por lo tanto, las CNN tienen interacciones dispersas entre las neuronas, al emplear núcleos que son más pequeños que la entrada.

Al igual que las redes neuronales regulares, las CNN están formadas por varias capas apiladas.

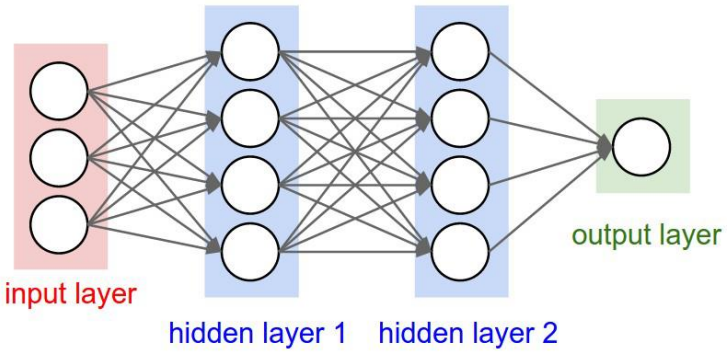


Figura 4.2: Visualizaciones de la estructura de una red neuronal por Li. [LKJ].

En la figura 4.3, se puede ver la arquitectura de la CNN llamada LeNet propuesta por LeCun. [LBBH98]. El tipo de capa más importante es la capa convolucional. En esta capa se calcula la convolución de la imagen de entrada con diferentes filtros. Una CNN contiene múltiples capas convolucionales, donde cada capa tiene filtros, cuya complejidad depende de la ubicación de la capa dentro de la red. Los filtros de las capas tempranas detectan características de bajo nivel como bordes y formas, mientras que los filtros posteriores pueden detectar características de alto nivel. Los resultados de las convoluciones de estos filtros con la imagen de entrada se combinan para recibir la salida de la red. La salida es un vector con probabilidades de clase, donde el tamaño del vector depende del número de clases.

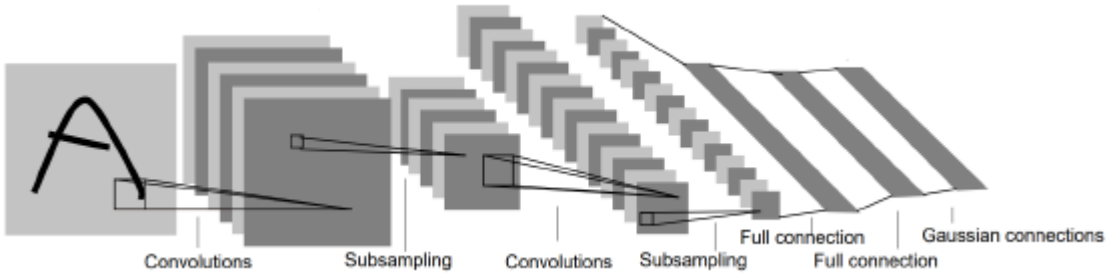


Figura 4.3: Visualización de la arquitectura LeNet por LeCun. [LBBH98].

Además de la capa convolucional existen varios tipos de capas con diferentes propósitos. En el tutorial de la Universidad de Stanford por Li. [LKJ] se presentan cinco tipos de capas:

• Capa de entrada

La capa de entrada de CNN es la entrada con estructura de matriz. En la mayoría de las aplicaciones, las entradas son imágenes con canales de color RGB.

• Capa convolucional

En esta capa se realiza la convolución del volumen de entrada con los pesos de las neuronas. Los pesos son parámetros que se aprenden durante la fase de entrenamiento de la red.

• Capa de unidad lineal rectificada (ReLU)

Por cada píxel se aplica una función de activación. Por ejemplo, max (0, x) establecería las entradas negativas en cero. Como esta es una función fija, ReLu Layer no proporciona parámetros entrenables.

• Capa de agrupamiento / submuestreo

La capa de agrupación es responsable de reducir el muestreo de la entrada en ancho y alto, pero no en profundidad. Este paso es útil para suavizar la imagen y, por lo tanto, reducir la influencia del ruido. Al igual que la capa ReLU, esta capa no tiene parámetros entrenables.

• Capa completamente conectada

La última capa de las CNN, es la capa totalmente conectada. Como se mencionó anteriormente, en el caso de las CNN, genera los puntajes de clase y cada neurona en esta capa está conectada con cada neurona de la capa anterior. Esta capa implementa el cálculo utilizando parámetros entrenables.

Entrada → [[Conv → ReLU]*N → Pool?]*M → [FC → ReLU]*K → FC

El símbolo de ? significa agregar una capa opcionalmente y los medios para repetir el término tan a menudo como se especifica. N, M y K se pueden elegir individualmente, pero deben ser números naturales, incluido el 0.

Entrenamiento de CNNs

Para entrenar una CNN se deben considerar varios bloques de construcción. Estos bloques de construcción y el proceso de entrenamiento en sí se describirán en los siguientes párrafos.

Una CNN es una función de puntuación $f\ (x)$ que obtiene a partir de las imágenes de entrada la probabilidad de que esa imagen pertenezca a una clase.

La función tiene un conjunto de parámetros, que se pueden controlar, o en términos de CNN, se pueden adaptar. El objetivo es adaptar los pesos de modo que las probabilidades de la clase coincidan con las etiquetas verdaderas lo más posible.

Para obtener esto, se necesita la función de pérdida. Su tarea es determinar qué tan buena es la predicción con respecto a la clase verdadera de cada ejemplo de entrenamiento. Una pérdida alta indica una clasificación pobre, mientras que una pérdida baja muestra una clasificación acertada.

La función de pérdida más popular utilizada con CNN es la pérdida de entropía cruzada que se calcula con la expresión:

$$L = -\sum_{i=0}^{N} (y_i \log q_i)$$

donde *q* se estima utilizando la función Softmax. La función Softmax genera las probabilidades de clase entre 0 y 1, que suman 1.

El proceso, de encontrar los pesos, que minimiza la función de pérdida es un problema de optimización.

El gradiente de la función de pérdida indica la mejor dirección para cambiar los pesos. El gradiente descendente es el proceso de realizar regularmente una actualización de parámetros mediante el cálculo y la evaluación de la perdida.

El problema con los gradientes es que no sabemos qué tan lejos tenemos que ir en la dirección especificada por el gradiente. Para avanzar, tenemos que establecer un tamaño de paso, que utilizamos para actualizar cuidadosamente los parámetros. Este tamaño de paso a menudo se conoce como tasa de aprendizaje. Si elegimos un tamaño de paso que es demasiado pequeño, solo se realiza un pequeño progreso. Si es demasiado grande, podríamos superar el óptimo.

Para un mejor rendimiento, el descenso de gradiente y la actualización de parámetros no se realizan para cada muestra de entrenamiento. Solo se realiza para lotes de muestras de entrenamiento.

La propagación hacia atrás es el proceso de calcular los gradientes mediante el uso recursivo de la regla de la cadena capa por capa de una CNN.

Antes de la capacitación de una CNN, los datos de la capacitación deben procesarse previamente. El preprocesamiento de los datos de entrada consiste en la estandarización para obtener el mismo rango para cada muestra de datos. Además, como las CNN están capacitadas para un tamaño de entrada explícito, la entrada debe redimensionarse o recortarse a ese tamaño específico.

El aumento de datos es tan importante como el paso de preprocesamiento. Ayuda a encontrar un modelo, que proporciona una buena generalización para los datos de entrada. El aumento sólo se aplica en el conjunto de entrenamiento. El conjunto de prueba permanece sin cambios, excepto para el preprocesamiento.

Los aumentos comunes de la entrada son recorte, traslaciones, rotaciones y giros.

Si se realiza el recorte de la imagen, una ventana del tamaño que se toma como entrada de cada red se recorta de una imagen más grande y se usa para entrenamiento. La Figura 4.4 muestra un ejemplo de cómo recortar 5 imágenes diferentes de una imagen más grande.

El proceso de formación de una CNN sigue un pipeline.

Un lote de muestras de entrenamiento se pasa a través de la red para obtener probabilidades de clase.

Con estas probabilidades y las etiquetas de las imágenes, se calcula la pérdida. La propagación hacia atrás se utiliza para calcular el gradiente de la pérdida y, posteriormente, se realiza la actualización del parámetro. Estos pasos se repiten hasta obtener los resultados deseados.

Existen diferentes técnicas para aplicar CNN para una tarea específica. La siguiente subsección describe el uso de redes no entrenadas, también conocido como capacitación desde cero.

Posteriormente se presentan diferentes métodos de transferencia de aprendizaje. Además, se presentan algunas de las arquitecturas CNN más populares.

Entrenamiento desde cero

Los parámetros de una CNN se inicializan usando una distribución gaussiana con una media de 0 y una desviación estándar de 0.01. Entrenar una red no entrenada desde cero puede ser una tarea difícil. Para el entrenamiento de redes profundas como InceptionV3 (ver sección 4.2.2) se requiere un gran conjunto de datos. Un conjunto de datos que es lo suficientemente grande como para entrenar redes profundas es ImageNet (consulte la sección 3.2). La capacitación de una CNN profunda en ImageNet lleva semanas en varias GPU [LKJ]. Pero no todas las tareas de clasificación son similares a las imágenes de ImageNet. Si el nuevo conjunto de datos diferente no es lo suficientemente grande como para entrenar una red desde cero, puede ser beneficioso utilizar modelos de CNN pre-entrenados y transferir el aprendizaje de un conjunto de datos a un conjunto de datos nuevo y diferente. Este método se describe en la siguiente sección.

Transferencia de aprendizaje

El aprendizaje por transferencia [YCBL14] se logra mediante el uso de redes pre-entrenadas. Su objetivo es utilizar los parámetros entrenados de una tarea para otra tarea nueva y diferente. Para las arquitecturas más conocidas (ver sección 4.1.2), los pesos de las CNN, entrenados en diferentes conjuntos de datos, están disponibles públicamente. Estas redes se denominan redes pre-entrenadas. La mayoría de los pesos disponibles se han calculado entrenando en ImageNet (ver sección 3.2). El aprendizaje por transferencia es útil debido al hecho de que las CNN aprenden características más generales y de bajo nivel como bordes y formas en las capas iniciales y las funciones más específicas de la tarea en las capas posteriores.

Para resolver el nuevo problema de clasificación, una opción es intercambiar la última capa completamente conectada de la red utilizada con una capa correspondiente al número de clases, requerido para la nueva tarea.

Esta nueva capa totalmente conectada debe ser entrenada utilizando el nuevo conjunto de imágenes del problema de interés.

Otra posibilidad es eliminar completamente la última capa totalmente conectada, luego extraer las características de todos los datos y, al final, clasificar estos vectores de características con una SVM (consulte la sección 4.2.1).

CNNs de ajuste fino

El ajuste fino de las redes significa entrenar nuevamente los pesos de una red pre-entrenada, pero con una tasa de aprendizaje más baja para clasificar un conjunto de datos diferente. Se ha demostrado que las características más generales, como los bordes y las formas, se pueden reutilizar de las redes y estas se encuentran en las capas iniciales [YCBL14].

Los valores de todas las capas se pueden ajustar o solo la última capa completamente conectada, esto depende de la similitud del nuevo conjunto de datos con el conjunto de datos de entrenamiento inicial.

Arquitecturas CNN

Existe una amplia variedad de diferentes arquitecturas CNN con diferentes características. Esta sección proporciona una breve descripción de las arquitecturas utilizadas en esta tesis en términos de clasificación de un conjunto de datos en imágenes saludables y de TBC.

La arquitectura AlexNet de Krizhevsky [KSG12], fue el ganador del desafío de reconocimiento visual a gran escala de ImageNet 2012 (ILSVRC12) y se visualiza en la figura 4.5.

Este modelo consta de cinco capas convolucionales y tres totalmente conectadas con un total de 60 millones de parámetros.

La arquitectura GoogLeNet de Szegedy [SLJ + 15]. ganó el ILSVRC14. En esta arquitectura se desarrolló un módulo llamado 'Inception' que representa una pequeña red dentro de otra red. El módulo Inception podría reducir la cantidad de parámetros a 12 veces menos que AlexNet. La arquitectura consta de varios módulos de inicio apilados. GoogLeNet tiene 22 capas de profundidad. Varias mejoras resultaron en varias versiones después de GoogLeNet. La versión mejorada de GoogLeNet es InceptionV3 de Szegedy [SVI + 15]. La arquitectura de InceptionV3 se puede ver en la figura 4.6.

La VGGNet por Simonyan. [SZ15] es otro enfoque, que intenta aumentar el rendimiento al aumentar la profundidad de la red.

Como se mencionó anteriormente, las características de bajo nivel se pueden encontrar en capas anteriores y las características específicas en capas posteriores. Esto significa que al aumentar la profundidad se pueden aprender características aún más complejas.

La red participó en el ILSVRC14, al igual que GoogLeNet, que pudo superar a la VGGNet. La VGGNet resultó de pruebas en profundidad creciente entre 11 y 19 capas. Se publicaron los modelos con mejor rendimiento, VGG16 con 16 capas y VGG19 con 19 capas. Con los modelos más profundos, el número de parámetros aumentó a 138 millones de parámetros en VGG16 y a 144 millones de parámetros en VGG19.

La arquitectura de ResNet por He. [HZRS16] ganó el primer lugar en el ILSVRC15.

A medida que la complejidad aumenta con la profundidad de la red, He. Intentaba mejorar el aprendizaje introduciendo conexiones de salto y construyendo modelos más profundos. Con una profundidad de hasta 152 capas, es 8 veces más profundo que VGG19. Afirman que su red es menos compleja y que pueden obtener precisión a partir de la mayor profundidad. En esta tesis utilizamos ResNet50 con una profundidad de 50 capas y en la figura 4.8 se visualiza ResNet34 con 34 capas.

Network	Depth	Size	Parameters (Millions)	Image Input Size
alexnet	8	227 MB	61.0	227-by-227
vgg16	16	515 MB	138	224-by-224
vgg19	19	535 MB	144	224-by-224
squeezenet	18	4.6 MB	1.24	227-by-227
googlenet	22	27 MB	7.0	224-by-224
inceptionv3	48	89 MB	23.9	299-by-299
densenet201	201	77 MB	20.0	224-by-224
mobilenetv2	53	13 MB	3.5	224-by-224
resnet18	18	44 MB	11.7	224-by-224
resnet50	50	96 MB	25.6	224-by-224
resnet101	101	167 MB	44.6	224-by-224
xception	71	85 MB	22.9	299-by-299
inceptionresnetv2	164	209 MB	55.9	299-by-299
shufflenet	50	6.3 MB	1.4	224-by-224
nasnetmobile	*	20 MB	5.3	224-by-224
nasnetlarge	*	360 MB	88.9	331-by-331

Figura 4.4.1 Redes pre entrenadas contenidas en Matlab, profundidad, tamaño, imágenes utilizadas en millones y tamaño de imagen de entrada. [PrtCNN]

Figure 4.5: Arquitectura de AlexNet visualizada por Cohen [Coh].

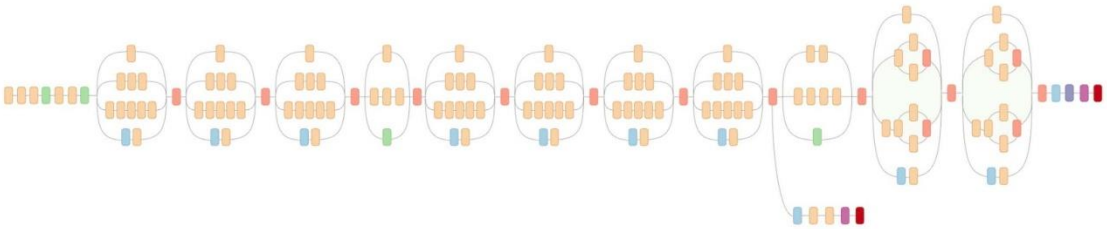


Figura 4.6: Arquitectura de InceptionV3 visualizada por Shlens [Shl].

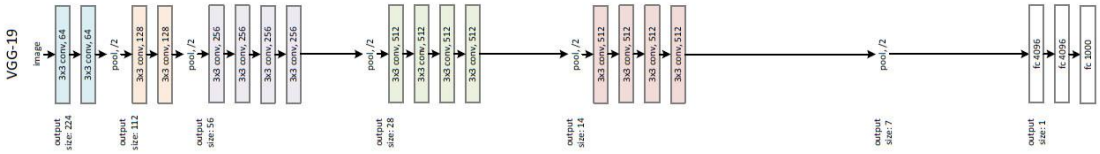


Figura 4.7: Arquitectura de VGG19 visualizada por He [HZRS16].

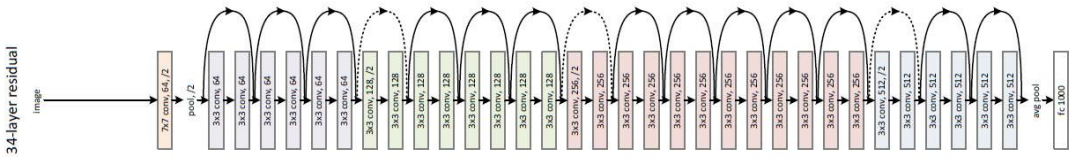


Figura 4.8: Arquitectura de ResNet34 visualizada por He [HZRS16].

Capítulo 5 Implementación

5.1 Obtención de imágenes y preprocesamiento

En el momento de la obtención de imágenes de las tomografías que están en el formato .NII se utilizó como preprocesamiento el siguiente código. Resaltando que la librería para poder hacer manipulación de dichos formatos fue Niftiread contenida dentro de Matlab.

```
function convert

directory = dir('*.nii');
SizeDir = size(directory,1);

for j = 1:SizeDir

aux = strrep(directory(j).name,'.nii','');
V = niftiread(aux);

      NumSlides = size(V,3);
      figure(1);


      mkdir(['C:\Users\User\Documents\Task2_TBT_TrainingSet\',aux]);

      for k = 1:NumSlides

          k1 = num2str(k);
          imshow(V(:,:,k),[])
          drawnow
          s = strcat(aux,'/nii_tbt',k1,'.jpg');
          frame = getframe(1);
          im = frame2im(frame);
          [imind,cm] = rgb2ind(im,256);
          [Y,newmap] = imresize(imind,cm,[32 32]);
          imwrite(Y,newmap,s,'jpeg')

          pause(.1)
      end

      end
```

Posterior a realizar el proceso, se obtuvieron las 1008 carpetas correspondientes a cada uno de los pacientes de las tomografías previamente proporcionadas. Un dato

importante por destacar es el costo computacional en tiempo, el ejecutar esta instrucción llevo un tiempo aproximado de 13 horas.

Posterior a esto se encontró el siguiente problema, las CNN admiten para su entrenamiento y validación una única imagen, al obtener toda la información acerca de las tomografías se obtiene entre 90 y 140 imágenes por paciente. El nuevo reto fue en compactar todas las imágenes para poder obtener una sola que recabara toda la información de cada paciente y cumpliera los requisitos para poder procesarlas e incluirlas en los algoritmos de procesamiento.

Después de contemplar varios paradigmas y por no desviarse del tema principal de esta tesis se utilizó el algoritmo Fuse, que compacta la información de dos imágenes y resalta la información no repetitiva creando una nueva imagen sintética, la primera carpeta de imágenes a entrenar fue recabada en escala de grises de la misma forma en que fueron retomadas con el algoritmo anterior.

```
close all;
directory = dir();
SizeDir = size(directory,1);

for i=3:SizeDir
D=directory(i).name;
directory_nii = dir(fullfile(D,'nii*.jpg'));
SizeDir_nii = size(directory_nii,1);

          A = imread(fullfile(D,directory_nii(1).name));
          B = imread(fullfile(D,directory_nii(2).name));
          C = imfuse(A,B,'blend','Scaling','joint');


      for j = 1:SizeDir_nii
            A = imread(fullfile(D,directory_nii(j).name));
            C = imfuse(A,C,'blend','Scaling','joint');

       end
      figure(1);
      imshow(C);

      %imwrite(C,'Imagen final.jpg');
      imwrite(C,fullfile('C:\Users\user\Documents\MATLAB\Task2_TBT_Tra
   iningSet\img_to_train',[num2str(i-2),'.jpg']));

      end
```

La segunda propuesta para obtener una comparativa es la de dividir en tres regiones iguales a las imágenes recabadas y cada una ingresarla dentro del espacio de cada color utilizando RGB como base, todas las CNN están pre entrenadas con imágenes RGB por lo que utilizando esa filosofía se contempla obtener resultados diferentes a los recabador con el primer algoritmo para cada CNN.

```
close all;
directory = dir();
SizeDir = size(directory,1);

for i=3:SizeDir
D=directory(i).name;
directory_nii = dir(fullfile(D,'nii*.jpg'));
SizeDir_nii = size(directory_nii,1);

      R =
   rgb2gray( imread(['.\' directory(i).name '\' directory_nii(1).name]));
      G =
   rgb2gray(imread(['.\' directory(i).name '\' directory_nii(round(SizeDi
   r_nii/3)).name]));
      B =
   rgb2gray(imread(['.\' directory(i).name '\' directory_nii(round(2*Size
   Dir_nii/3)).name]));


      for j = 2:round(SizeDir_nii/3)-1
          R1 =
   rgb2gray(imread(['.\' directory(i).name '\' directory_nii(j).name]));

          R = imfuse(R,R1,'blend','Scaling','joint');
      end
      for j = 1+round(SizeDir_nii/3):round(2*SizeDir_nii/3)-1
          G1 =
   rgb2gray(imread(['.\' directory(i).name '\' directory_nii(j).name]));

          G = imfuse(G,G1,'blend','Scaling','joint');
      end

      for j = 1+round(2*SizeDir_nii/3): SizeDir_nii
          B1 =
   rgb2gray(imread(['.\' directory(i).name '\' directory_nii(j).name]));

          B = imfuse(B,B1,'blend','Scaling','joint');
      end

```

```
29.       RGB(:,:,1)=R;
30.       RGB(:,:,2)=G;
31.       RGB(:,:,3)=B;
32.       figure(1);
33.       imshow(RGB);
34.
35.       imwrite(RGB,fullfile('C:\Users\user\Documents\MATLAB\Task2_TBT_T
   rainingSet\img_to_train_rgb',[num2str(i-2),'.jpg']));
36.
37.       end
```

Como tercer método se prueba se propuso considerar no aplicar ningún método de compactación en las imágenes únicamente utilizar todas las imágenes extraídas de las tomografías de cada paciente y asignarlas a cada tipo de TBT que tienen cada uno, la primera prueba se utilizaron 20 pacientes por cada tipo haciendo un total de 12 445 imágenes distribuidas en 5 carpetas. Con imágenes obtenidas de la siguiente manera.

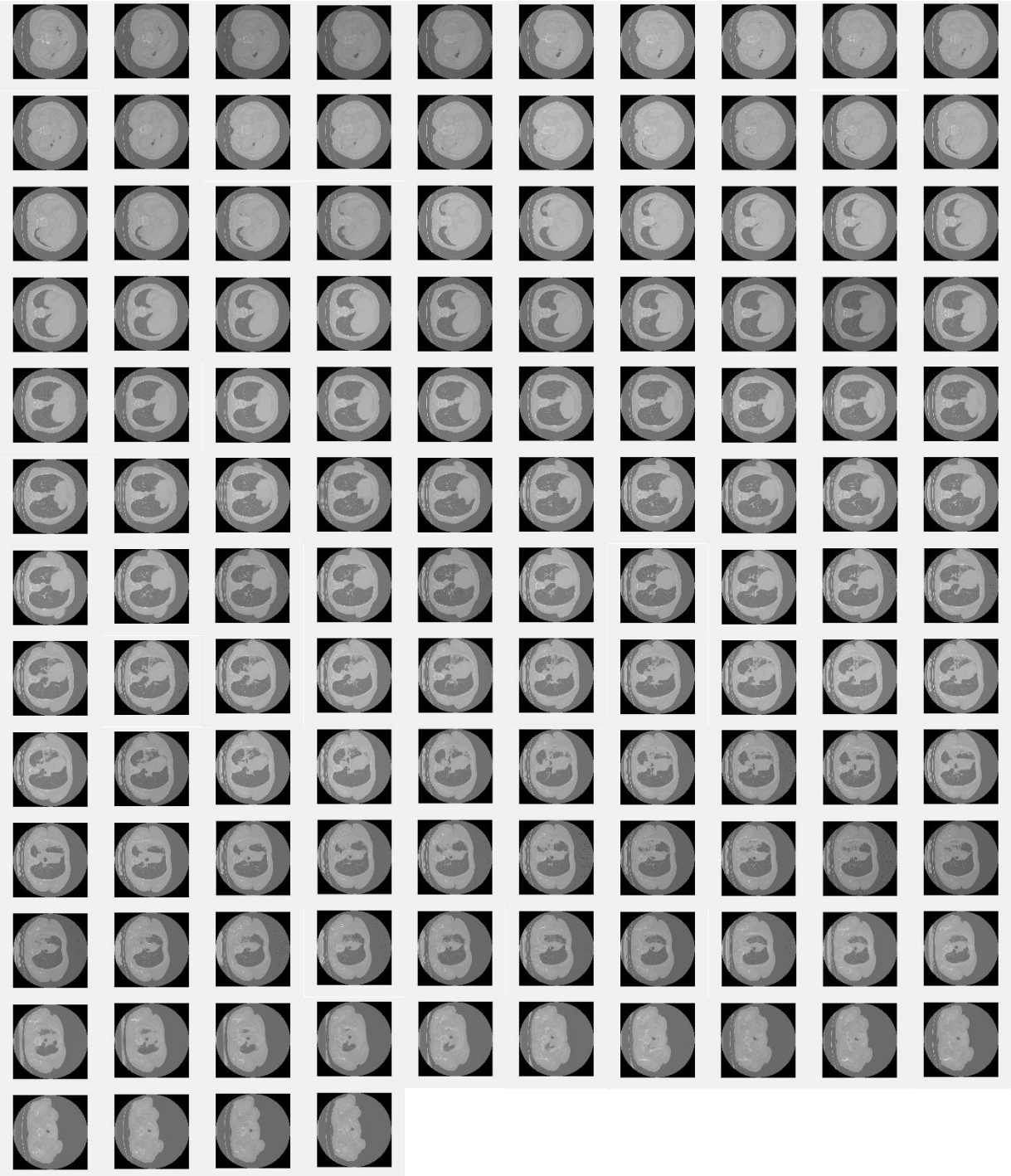


Figura 5.1 Collage representativo de la extracción de imágenes de cada paciente.

5.2 Implementación de las CNN

En este capítulo se procederá a implementar cada una de las CNN descritas y verificar después del proceso de reentrenamiento los resultados en el problema de clasificación.

El algoritmo utilizado para reentrenar las redes, fue el mismo para todas las redes utilizando las mismas configuraciones.

```
net = alexnet;
%net = googlenet;
%net = vgg16;
%net = vgg19;
%net = resnet18();

analyzeNetwork(net);

directory = uigetdir;
      image_dataset = imageDatastore(directory,'IncludeSubfolders',true,'LabelSource','foldernames');
      [imdsTrain,imdsValidation] = splitEachLabel(image_dataset,0.7);


      net.Layers(1)
      inputSize = net.Layers(1).InputSize;

      if isa(net,'SeriesNetwork')
        lgraph = layerGraph(net.Layers);
      else
        lgraph = layerGraph(net);
      end

      [learnableLayer,classLayer] = findLayersToReplace(lgraph);

      numClasses = numel(categories(imdsTrain.Labels));

      if isa(learnableLayer,'nnet.cnn.layer.FullyConnectedLayer')
          newLearnableLayer = fullyConnectedLayer(numClasses, ...
              'Name','new_fc', ...
              'WeightLearnRateFactor',10, ...
              'BiasLearnRateFactor',10);

      elseif isa(learnableLayer,'nnet.cnn.layer.Convolution2DLayer')
          newLearnableLayer = convolution2dLayer(1,numClasses, ...
              'Name','new_conv', ...
              'WeightLearnRateFactor',10, ...
              'BiasLearnRateFactor',10);
      end

      lgraph = replaceLayer(lgraph,learnableLayer.Name,newLearnableLayer);
```

```
41.
42.      newClassLayer = classificationLayer('Name','new_classoutput');
43.      lgraph = replaceLayer(lgraph,classLayer.Name,newClassLayer);
44.
45.      layers = lgraph.Layers;
46.      connections = lgraph.Connections;
47.
48.      layers(1:10) = freezeWeights(layers(1:10));
49.      lgraph = createLgraphUsingConnections(layers,connections);
50.
51.      pixelRange = [-30 30];
52.      scaleRange = [0.9 1.1];
53.      imageAugmenter = imageDataAugmenter( ...
54.          'RandXReflection',true, ...
55.          'RandXTranslation',pixelRange, ...
56.          'RandYTranslation',pixelRange, ...
57.          'RandXScale',scaleRange, ...
58.          'RandYScale',scaleRange);
59.      augimdsTrain =
   augmentedImageDatastore(inputSize(1:2),imdsTrain, ...
60.          'DataAugmentation',imageAugmenter);
61.
62.      augimdsValidation =
   augmentedImageDatastore(inputSize(1:2),imdsValidation);
63.
64.      miniBatchSize = 10;
65.      valFrequency = floor(numel(augimdsTrain.Files)/miniBatchSize);
66.      options = trainingOptions('sgdm', ...
67.          'MiniBatchSize',miniBatchSize, ...
68.          'MaxEpochs',6, ...
69.          'InitialLearnRate',3e-4, ...
70.          'Shuffle','every-epoch', ...
71.          'ValidationData',augimdsValidation, ...
72.          'ValidationFrequency',valFrequency, ...
73.          'Verbose',false, ...
74.          'Plots','training-progress');
75.
76.      net = trainNetwork(augimdsTrain,lgraph,options);
77.
78.      [YPred,probs] = classify(net,augimdsValidation);
79.      accuracy = mean(YPred == imdsValidation.Labels);
80.
81.      idx = randperm(numel(imdsValidation.Files),4);
82.      figure
83.      for i = 1:4
84.          subplot(2,2,i)
85.          I = readimage(imdsValidation,idx(i));
86.          imshow(I)
87.          label = YPred(idx(i));
88.          title(string(label) + ", "
   + num2str(100*max(probs(idx(i),:)),3) + "%");
89.      end
```

Cabe destacar que al principio se encuentran comentadas todas las CNN utilizadas y como primer elemento de prueba Alexnet. Y a continuación las pruebas realizadas y resultados obtenidos.

Alexnet:

- Profundidad: 8 capas
- Tamaño en memoria: 227 MB
- Parámetros: 61 M
- Tamaño de imagen de entrada: 227 x 227 px

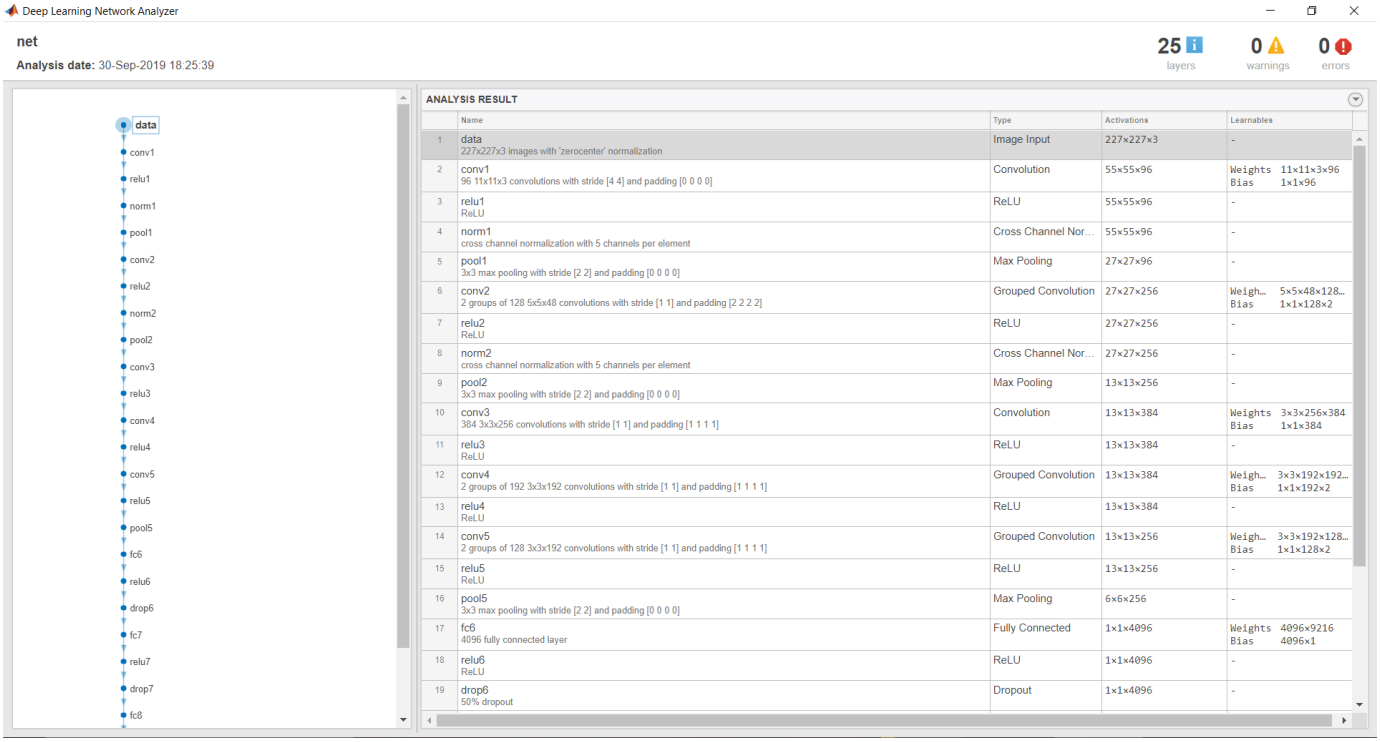


Figura 5.2.1.1 CNN analizada por Matlab.

Utilizando el primer algoritmo se obtuvieron los siguientes resultados.

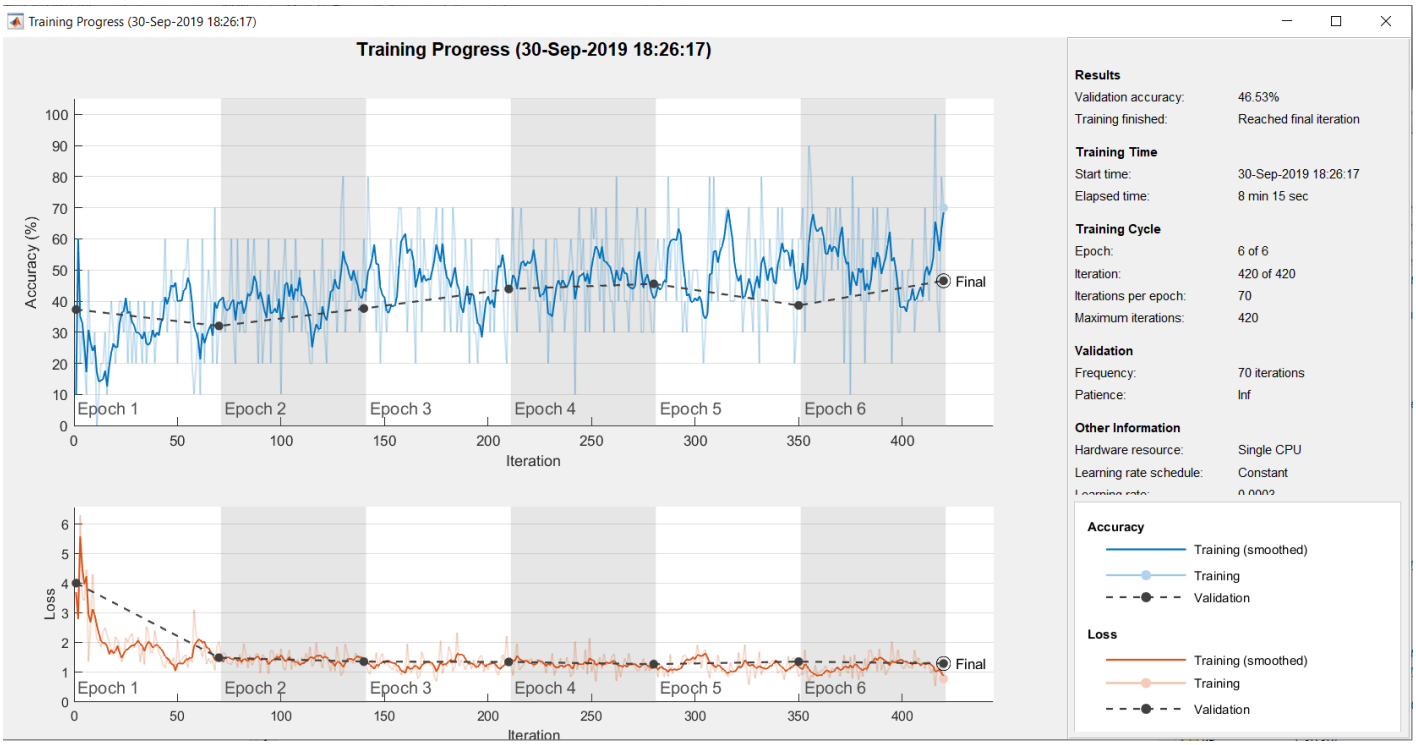


Figura 5.2.1.2 Resultados de la ejecución de Alexnet con banco de imágenes del Algoritmo 1 obteniendo una validación de 46.52% en un tiempo de 8 min 15 seg.

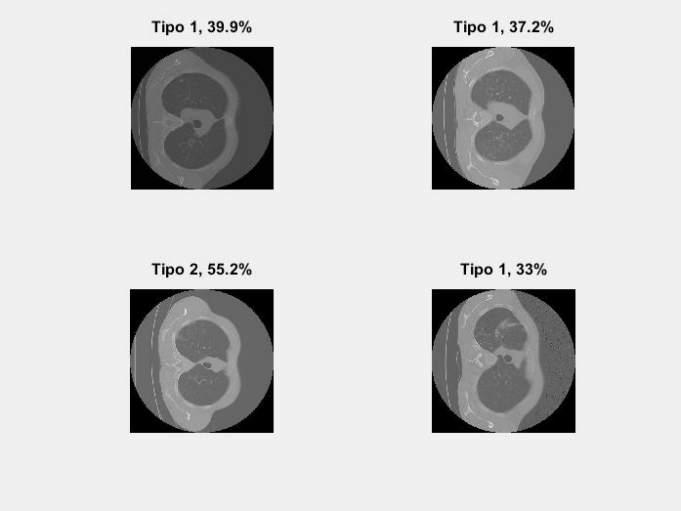


Figura 5.2.1.3 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 1.

Utilizando el segundo algoritmo se obtuvieron los siguientes resultados.

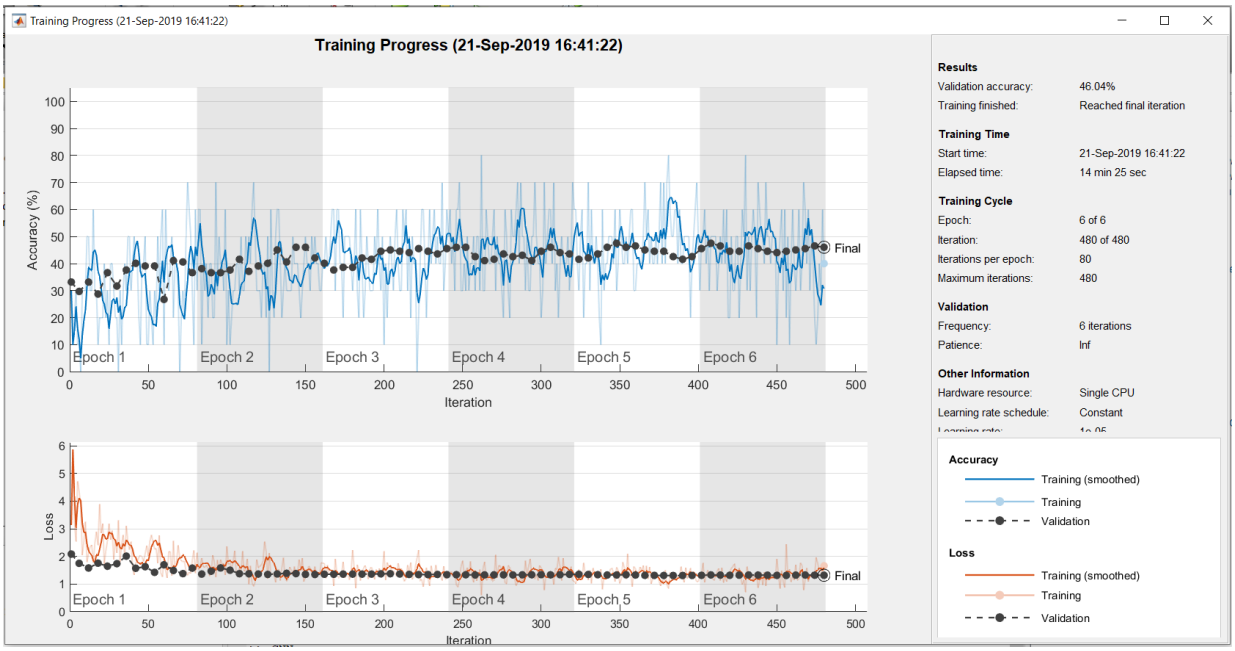


Figura 5.2.1.4 Resultados de la ejecución de Alexnet con banco de imágenes del Algoritmo 2 con una eficiencia de 46.06% con un tiempo de 14 min 25 seg.

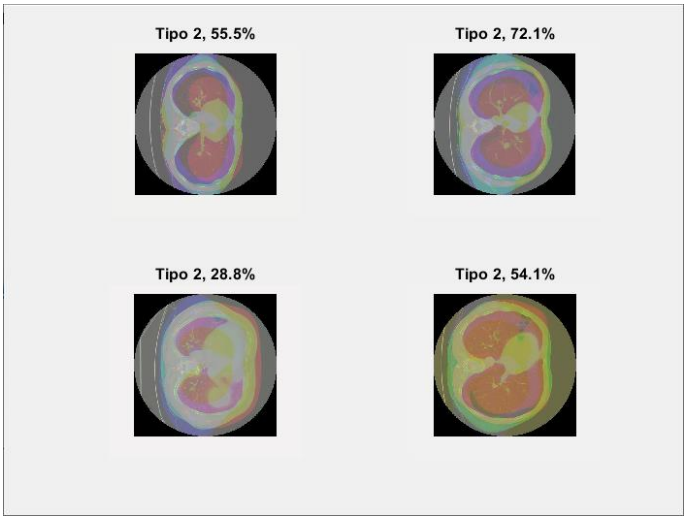


Figura 5.2.1.5 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 2.

Aplicando la prueba de colocar todas las imágenes de 20 pacientes se obtuvo el siguiente resultado.

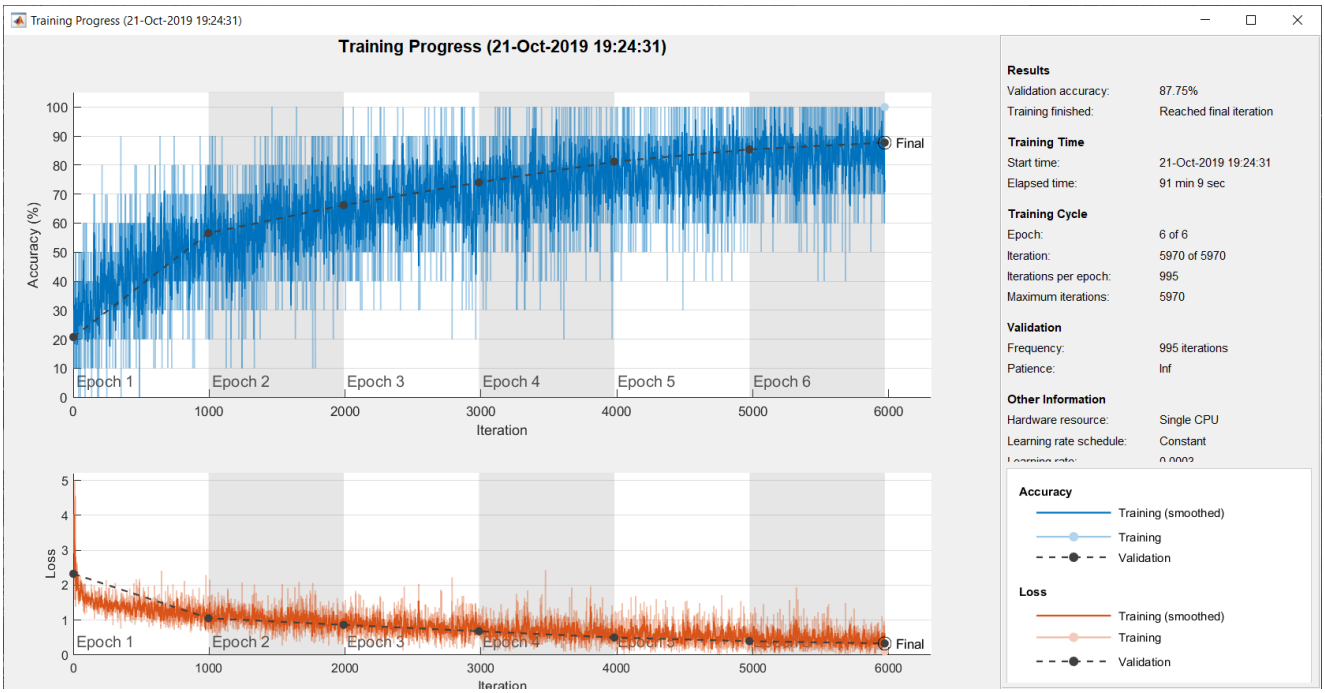


Figura 5.2.1.6 Resultados de la ejecución de Alexnet con banco de imágenes de 20 pacientes con una eficiencia de 87.75% y con un tiempo de 91 min 9 seg.

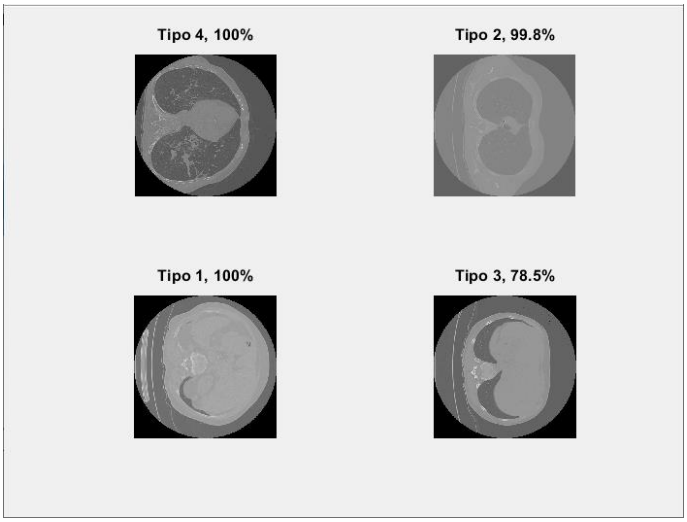


Figura 5.2.1.7 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes.

Googlenet:

- Profundidad: 22 capas
- Tamaño en memoria: 27 MB
- Parámetros: 7 M
- Tamaño de imagen de entrada: 244 x 244 px

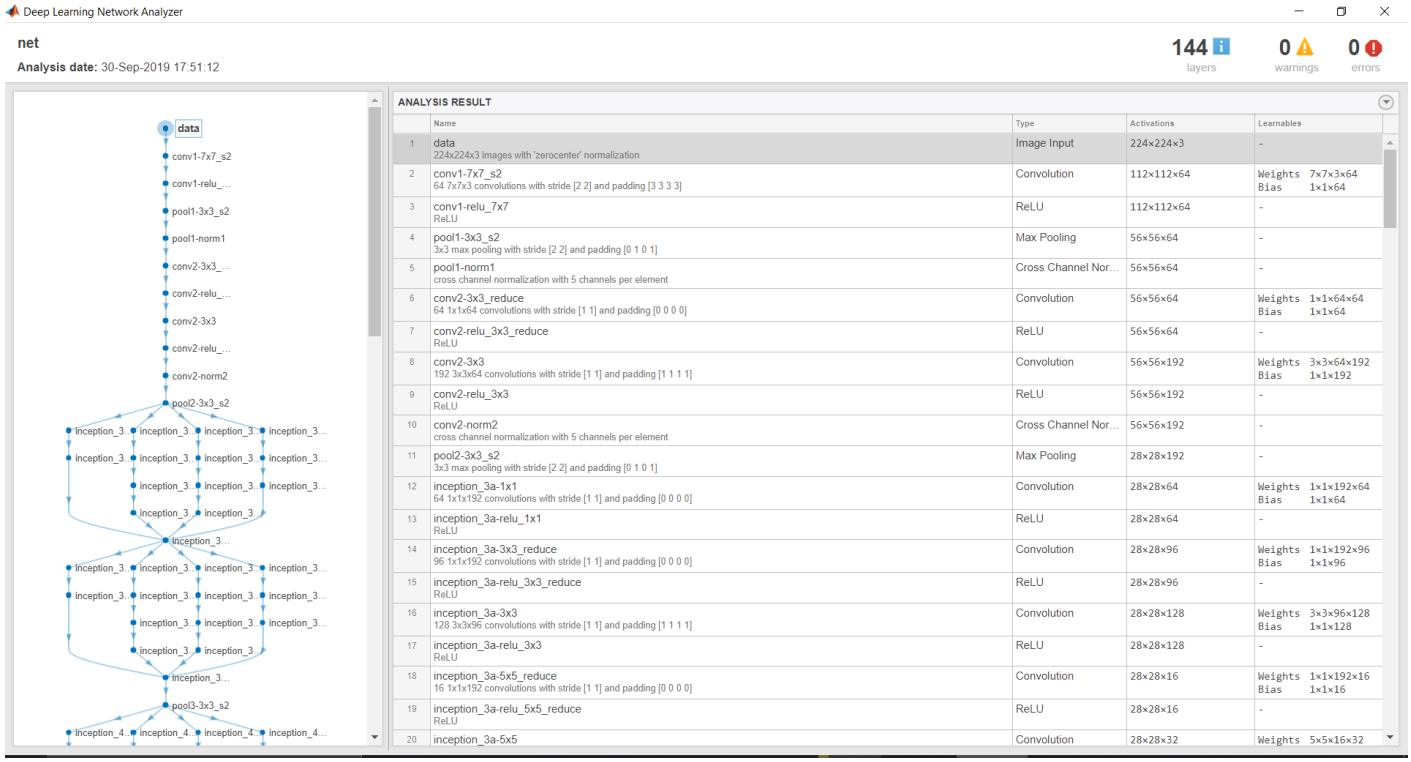


Figura 5.2.2.1 CNN analizada por Matlab.

Utilizando el primer algoritmo se obtuvieron los siguientes resultados.

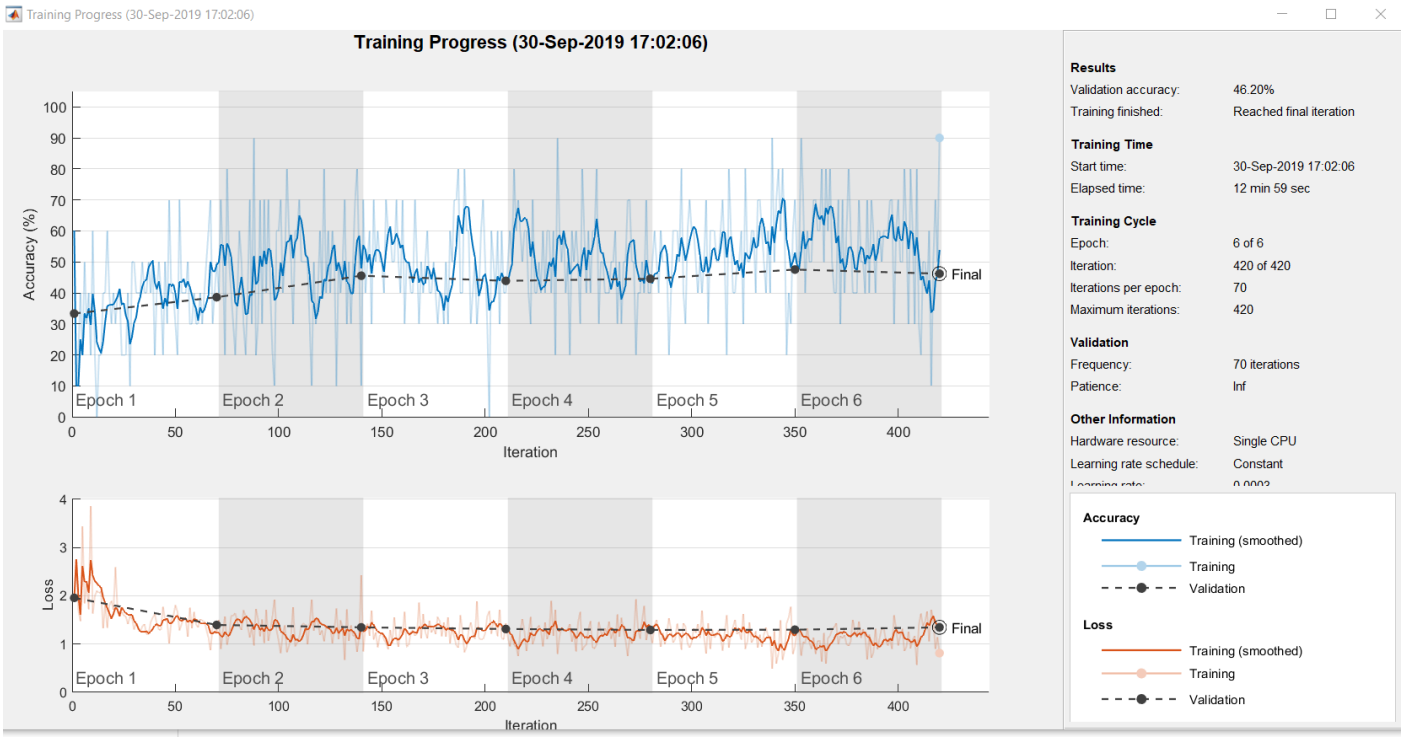


Figura 5.2.2.2 Resultados de la ejecución de Googlenet con banco de imágenes del Algoritmo 1 con una eficiencia de 46.20% y un tiempo de 12 min 59 seg.

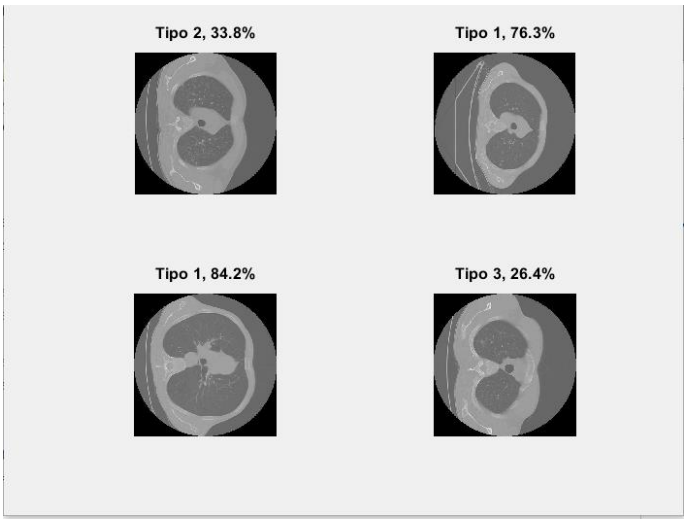


Figura 5.2.2.3 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 1.

Utilizando el segundo algoritmo se obtuvieron los siguientes resultados.

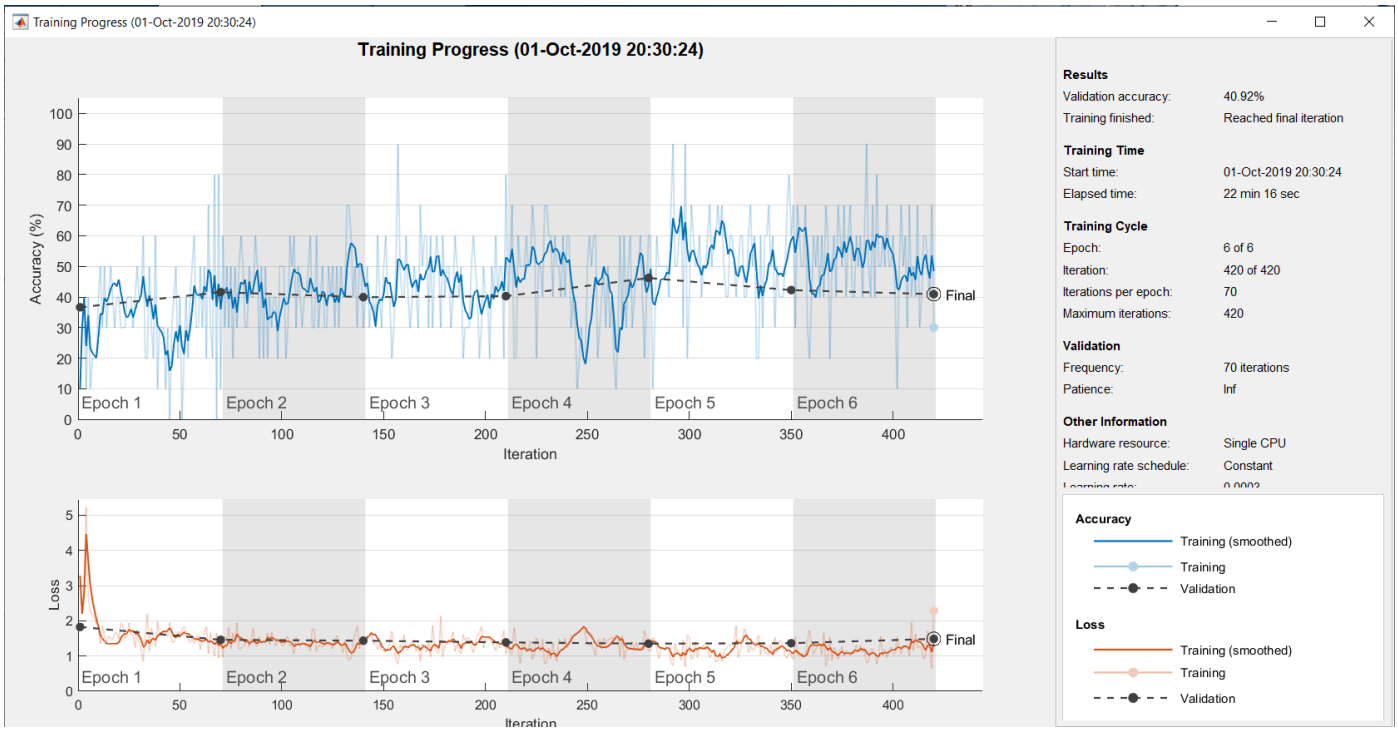


Figura 5.2.2.4 Resultados de la ejecución de Googlenet con banco de imágenes del Algoritmo 2 con una validación de 40.92% y un tiempo de 22 min 16 seg.

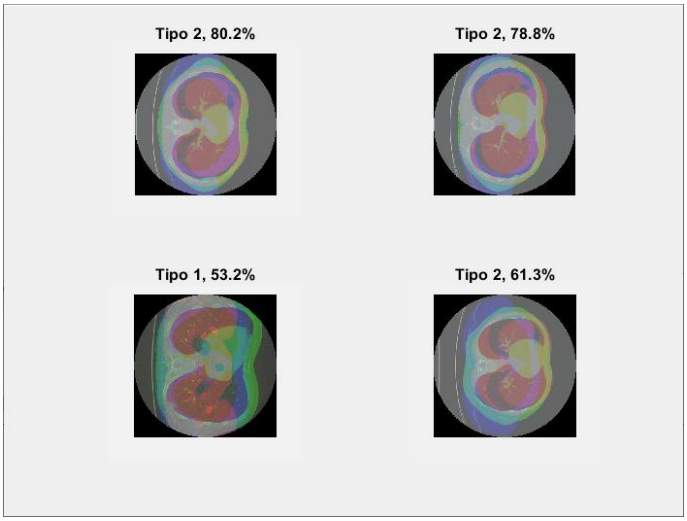


Figura 5.2.2.5 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 2.

Utilizando el banco de imágenes de 20 pacientes.

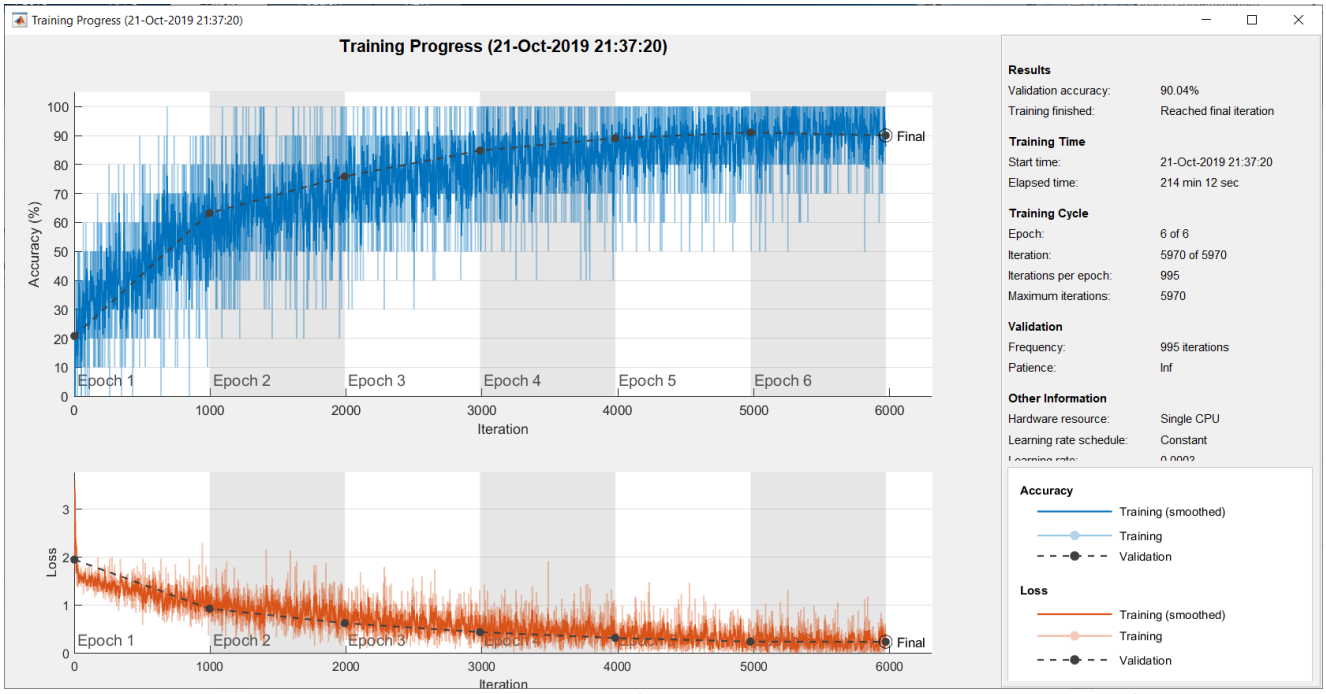


Figura 5.2.2.6 Resultados de la ejecución de Googlenet con banco de imágenes de 20 pacientes con una validación 90.04% y un tiempo 214 min y 12 seg.

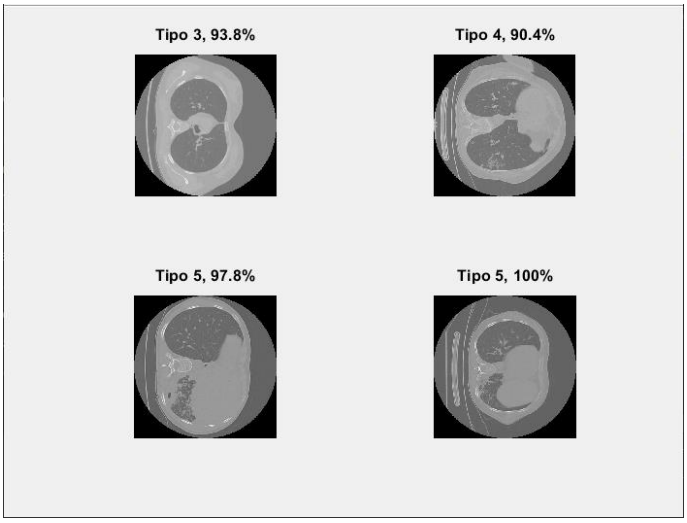
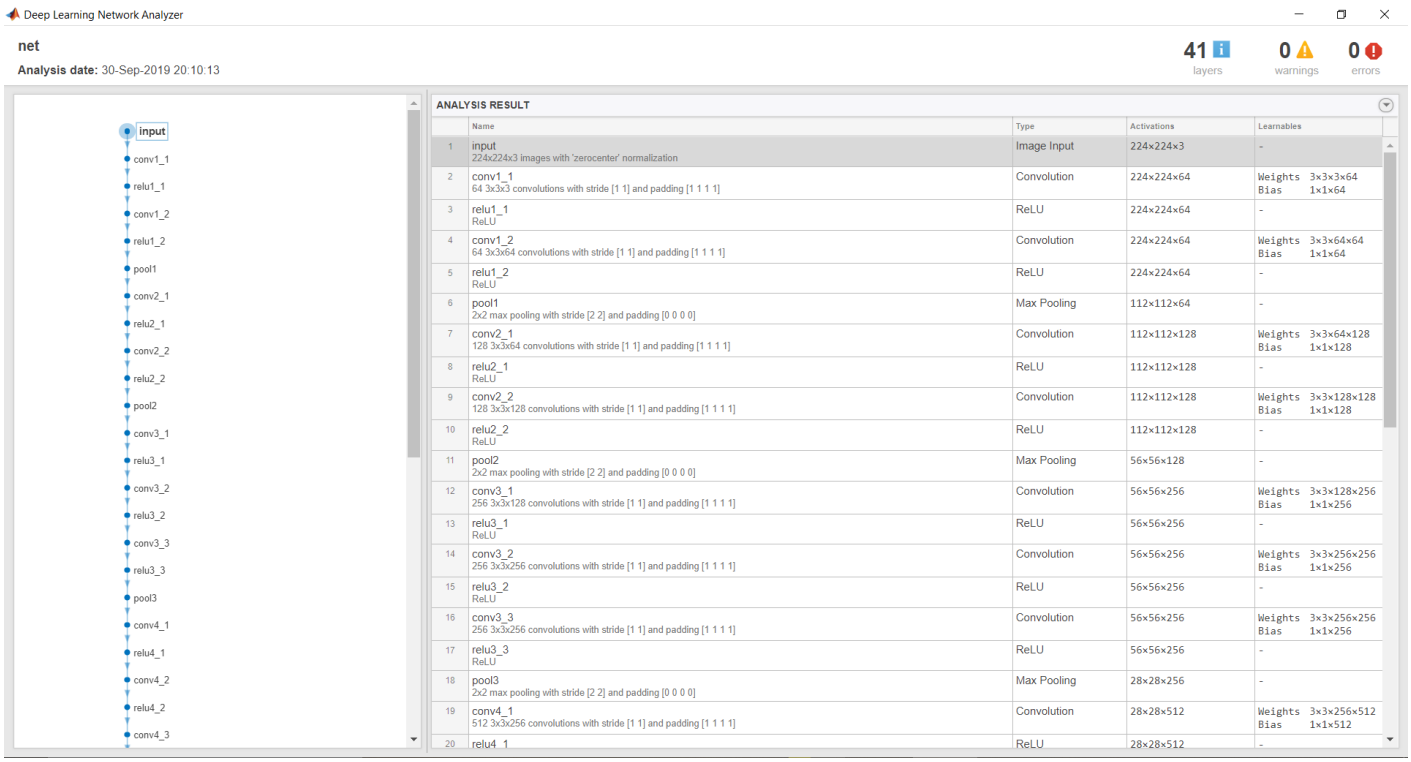


Figura 5.2.7 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes.

Vgg 16:

- Profundidad: 16 capas
- Tamaño en memoria: 515 MB
- Parámetros: 130 M
- Tamaño de imagen de entrada: 224 x 224 px



ANALYSIS RESULT

	Name	Type	Activations	Learnables
1	input 224x224x3 images with 'zerocenter' normalization	Image Input	224×224×3	-
2	conv1_1 64 3x3x3 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	224×224×64	Weights 3×3×3×64 Bias 1×1×64
3	relu1_1 ReLU	ReLU	224×224×64	-
4	conv1_2 64 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	224×224×64	Weights 3×3×64×64 Bias 1×1×64
5	relu1_2 ReLU	ReLU	224×224×64	-
6	pool1 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	112×112×64	-
7	conv2_1 128 3x3x64 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	112×112×128	Weights 3×3×64×128 Bias 1×1×128
8	relu2_1 ReLU	ReLU	112×112×128	-
9	conv2_2 128 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	112×112×128	Weights 3×3×128×128 Bias 1×1×128
10	relu2_2 ReLU	ReLU	112×112×128	-
11	pool2 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	56×56×128	-
12	conv3_1 256 3x3x128 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	56×56×256	Weights 3×3×128×256 Bias 1×1×256
13	relu3_1 ReLU	ReLU	56×56×256	-
14	conv3_2 256 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	56×56×256	Weights 3×3×256×256 Bias 1×1×256
15	relu3_2 ReLU	ReLU	56×56×256	-
16	conv3_3 256 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	56×56×256	Weights 3×3×256×256 Bias 1×1×256
17	relu3_3 ReLU	ReLU	56×56×256	-
18	pool3 2x2 max pooling with stride [2 2] and padding [0 0 0 0]	Max Pooling	28×28×256	-
19	conv4_1 512 3x3x256 convolutions with stride [1 1] and padding [1 1 1 1]	Convolution	28×28×512	Weights 3×3×256×512 Bias 1×1×512
20	relu4_1	ReLU	28×28×512	-

Figura 5.2.3.1 CNN analizada por Matlab

Utilizando el primer algoritmo se obtuvieron los siguientes resultados.

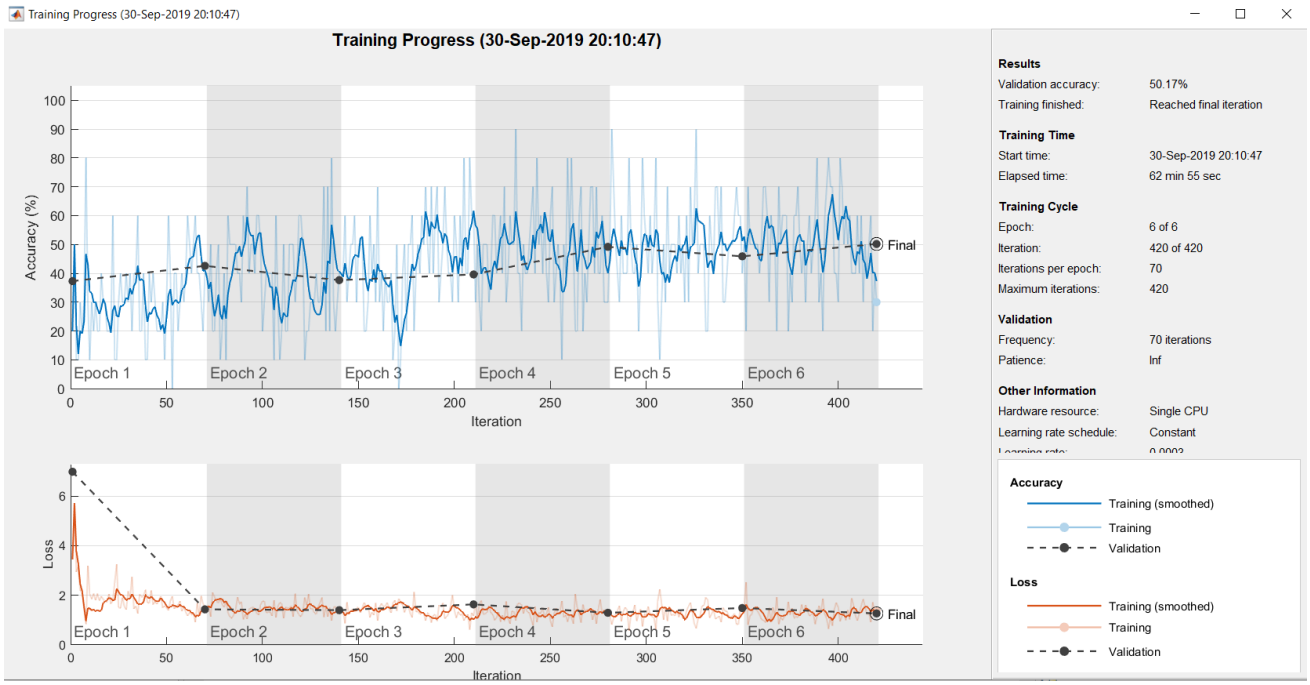


Figura 5.2.3.2 Resultados de la ejecución de Vgg16 con banco de imágenes del Algoritmo 1 con una validación de 50.17% y un tiempo de 62 min 55 seg.

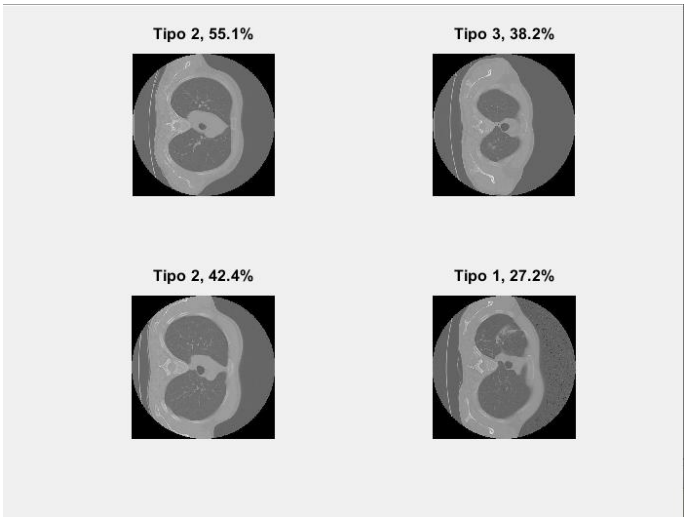


Figura 5.2.3.3 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 1.

Utilizando el segundo algoritmo se obtuvieron los siguientes resultados.

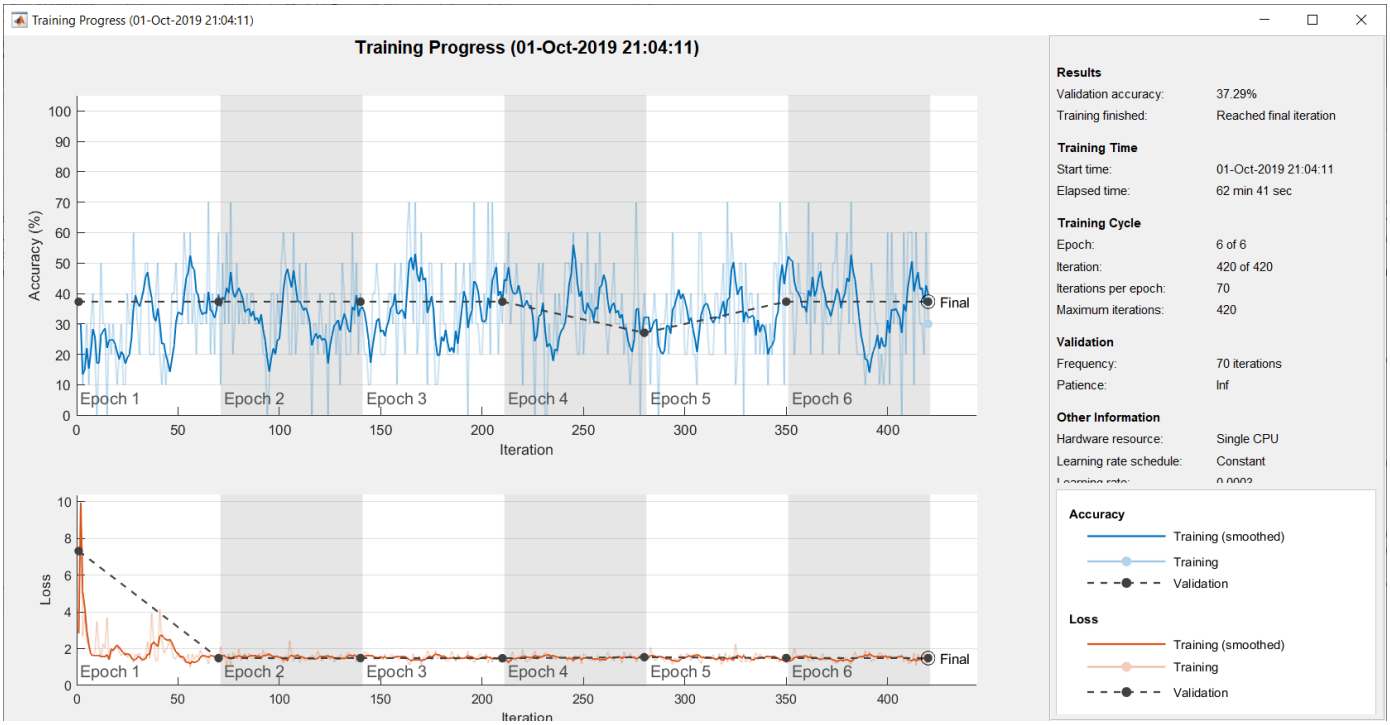


Figura 5.2.3.4 Resultados de la ejecución de Vgg16 con banco de imágenes del Algoritmo 2 con una validación de 37.29% y un tiempo de 62 min 41 seg.

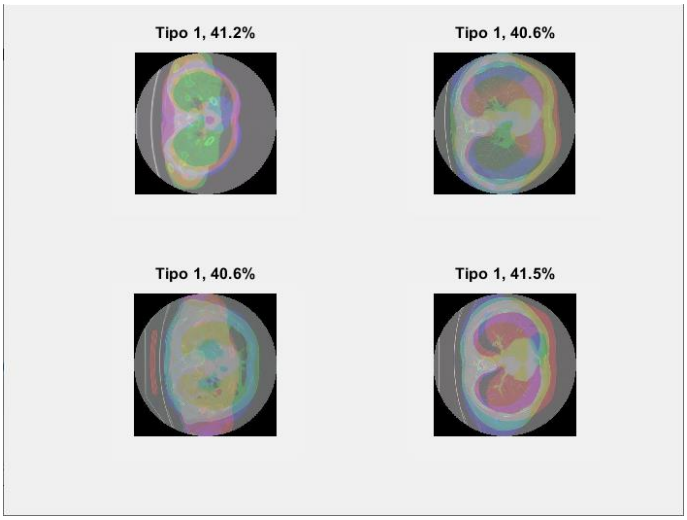


Figura 5.2.3.5 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 2.

Utilizando banco de imágenes de 20 pacientes.

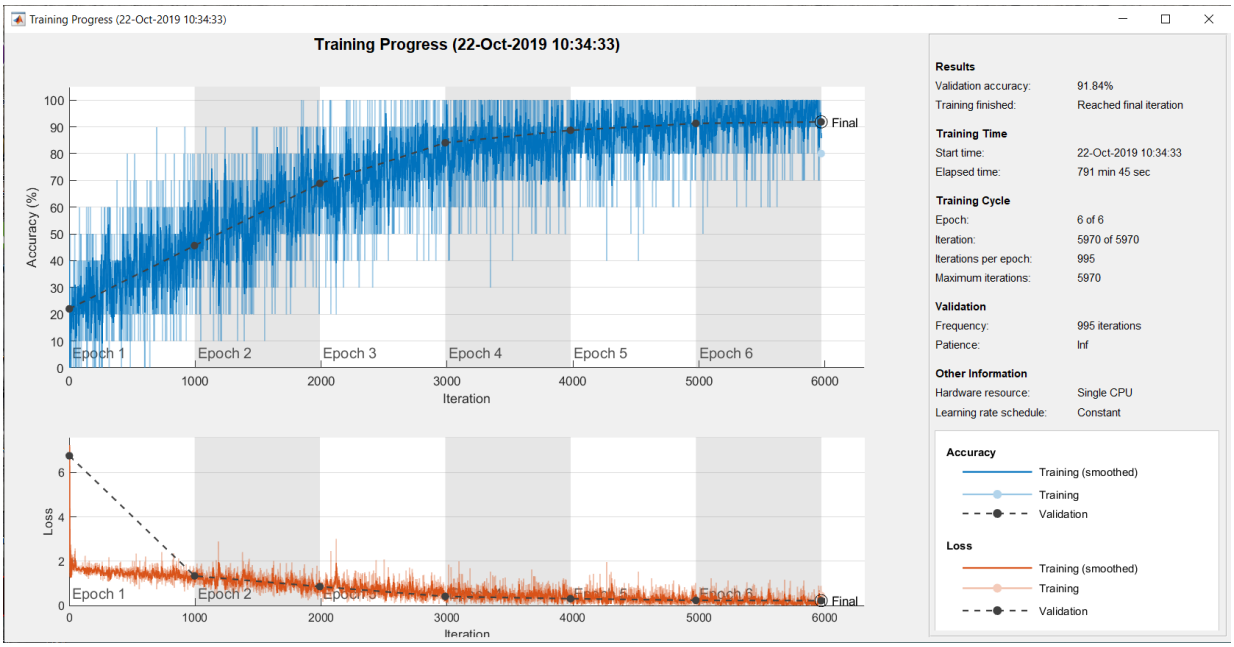


Figura 5.2.3.6 Resultados de la ejecución de Vgg16 con banco de imágenes de 20 pacientes con una validación de 91.84% 79 y con un tiempo de 791 min 45 seg.

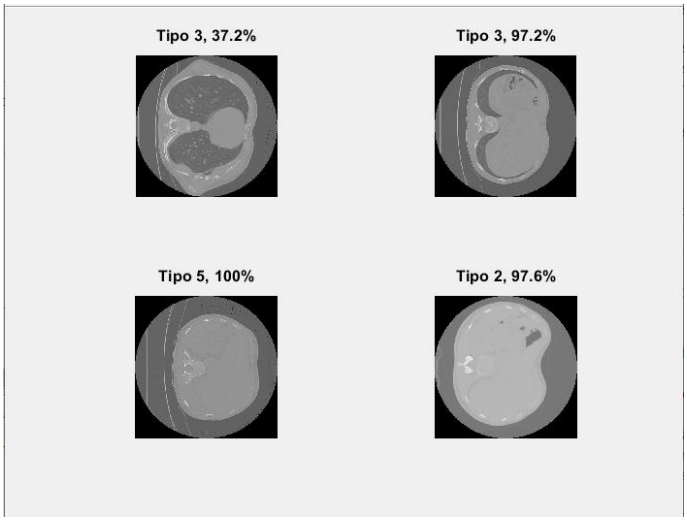


Figura 5.2.3.7 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes.

Vgg19:

- Profundidad: 19 capas
- Tamaño en memoria: 539mb
- Parámetros: 144 M
- Tamaño de imagen de entrada: 244 x 244 px

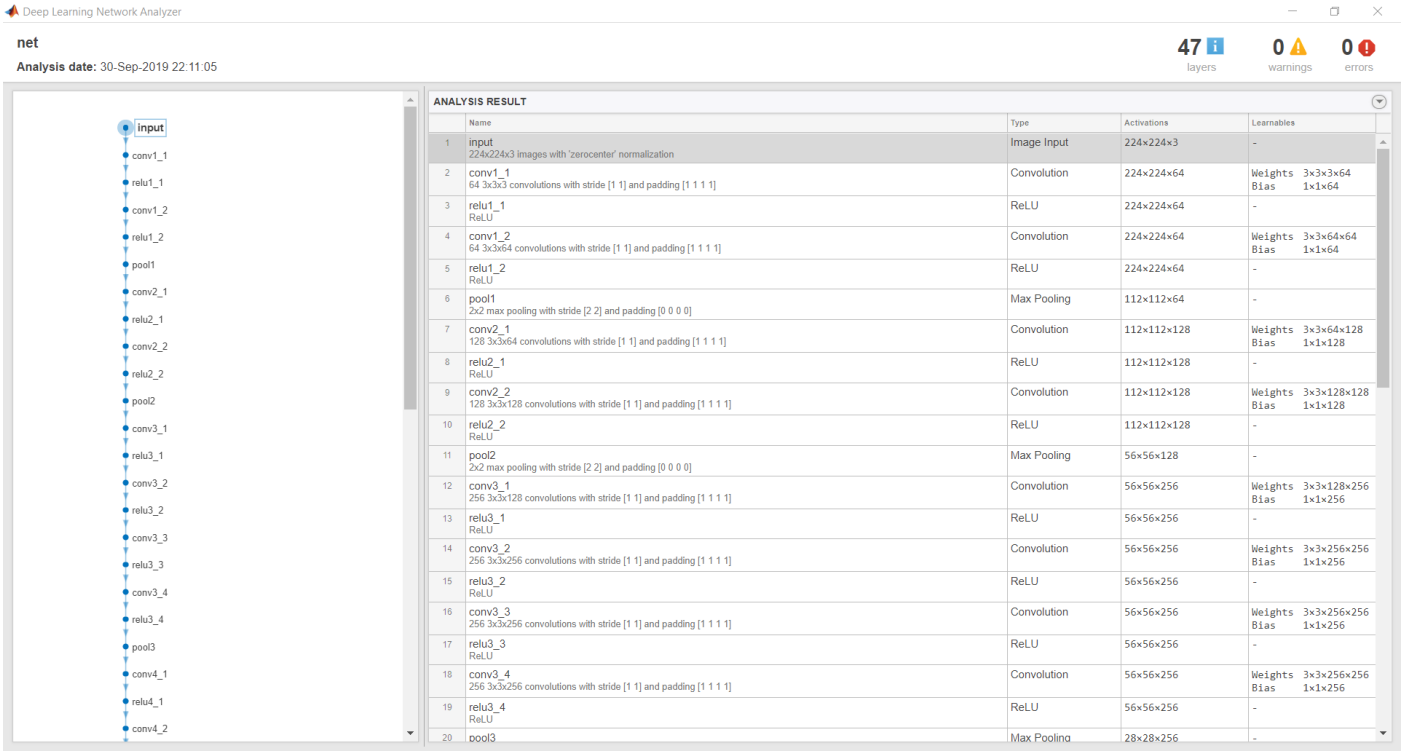


Figura 5.2.4.1 CNN Analizada por Matlab

Utilizando el primer algoritmo se obtuvieron los siguientes resultados.

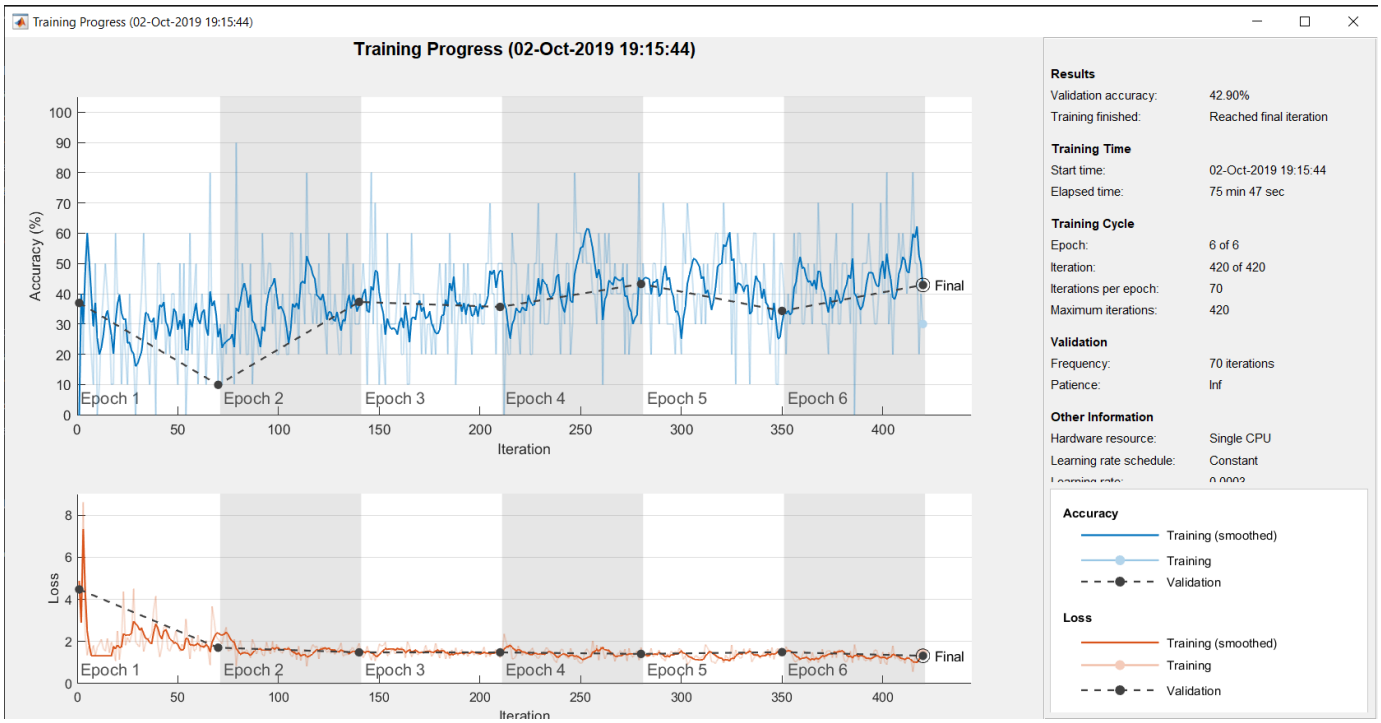


Figura 5.2.4.2 Resultados de la ejecución de Vgg19 con banco de imágenes del Algoritmo 1 con una validación de 42.90%con un tiempo de 75 min 47 seg.

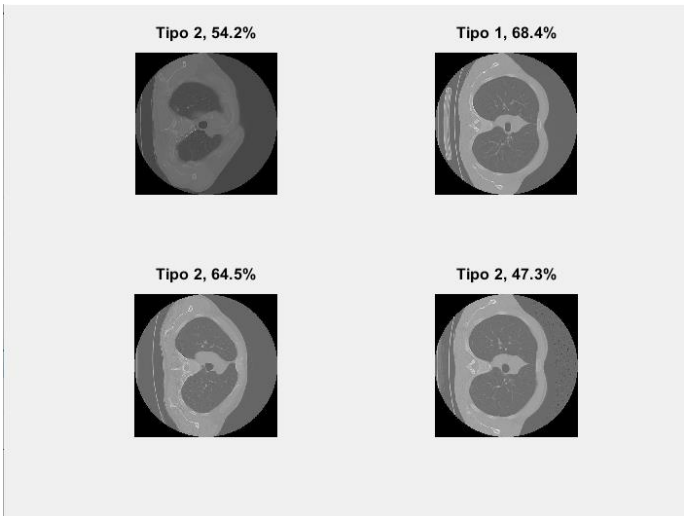


Figura 5.2.4.3 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 1.

Utilizando el segundo algoritmo se obtuvieron los siguientes resultados

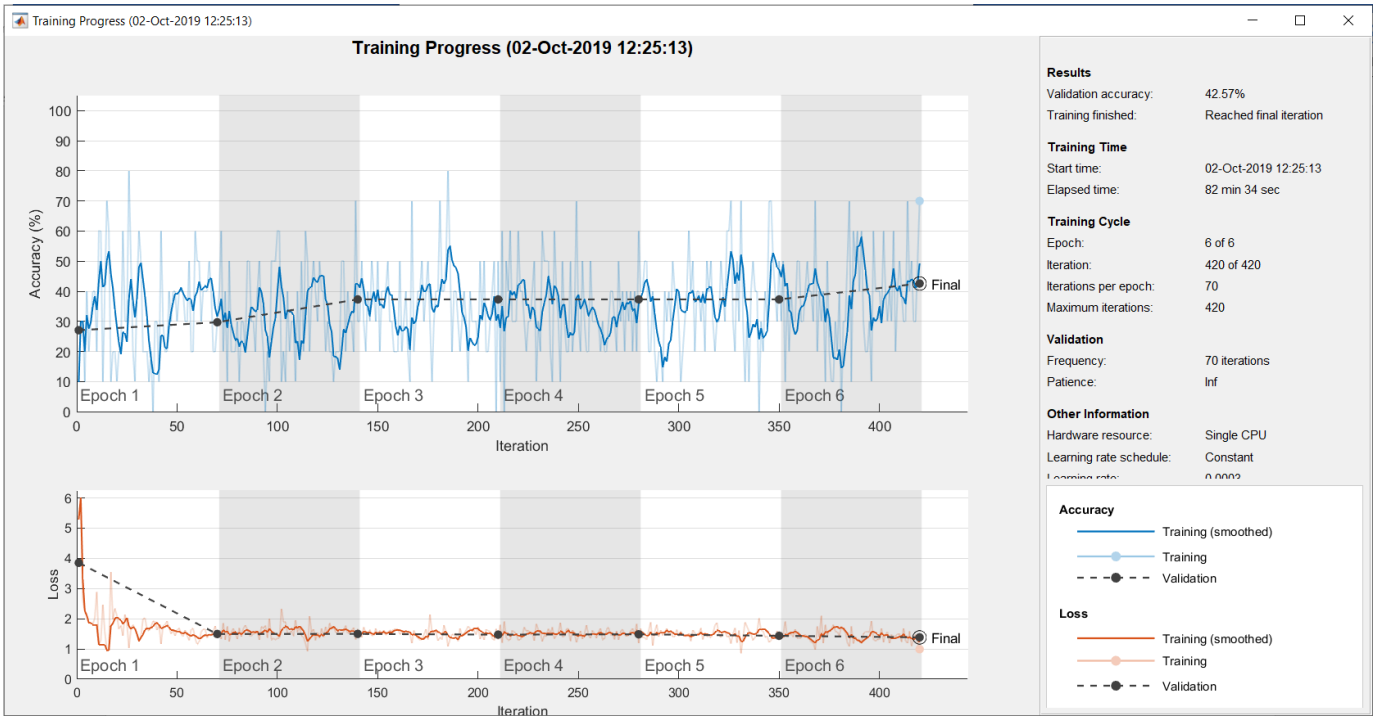


Figura 5.2.4.4 Resultados de la ejecución de Vgg19 con banco de imágenes del Algoritmo 2 con una validación 42.57% y un tiempo de 82 min 34 seg.

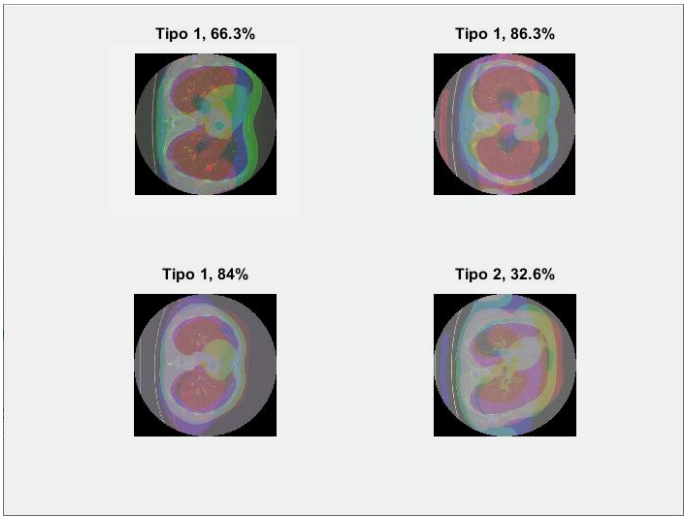


Figura 5.2.4.5 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 2.

Utilizando banco de imágenes de 20 pacientes.

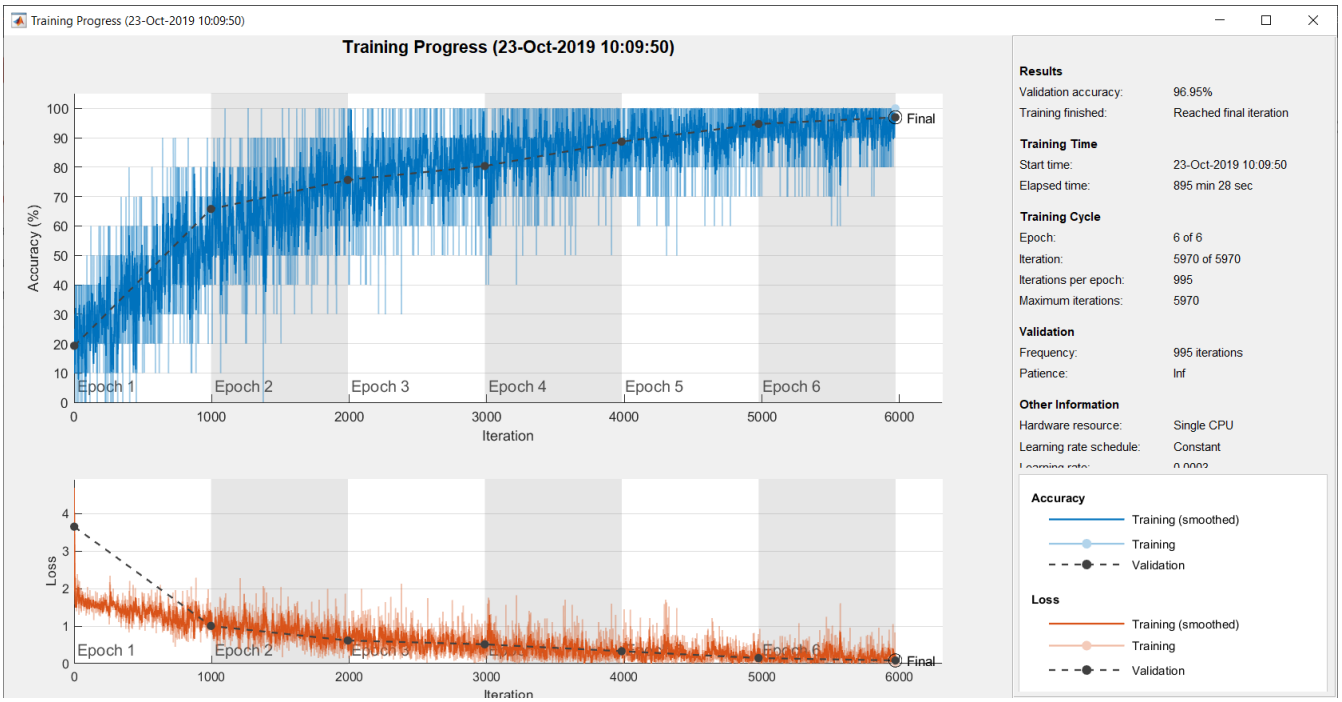


Figura 5.2.4.6 Resultados de la ejecución de Vgg19 con banco de imágenes del Algoritmo con una validación de 96.95% y un tiempo de 895 min y 29 seg.

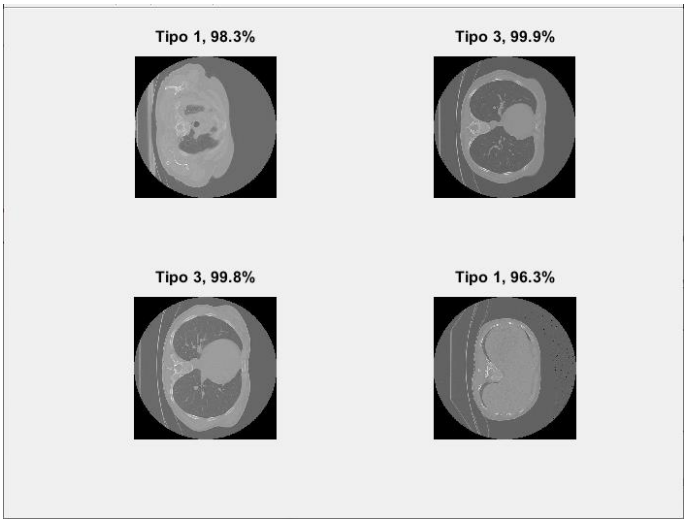


Figura 5.2.4.7 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes.

Resnet18:

- Profundidad: 18 capas
- Tamaño en memoria: 44 MB
- Parámetros: 11.7 M
- Tamaño de imagen de entrada: 224 x 224 px

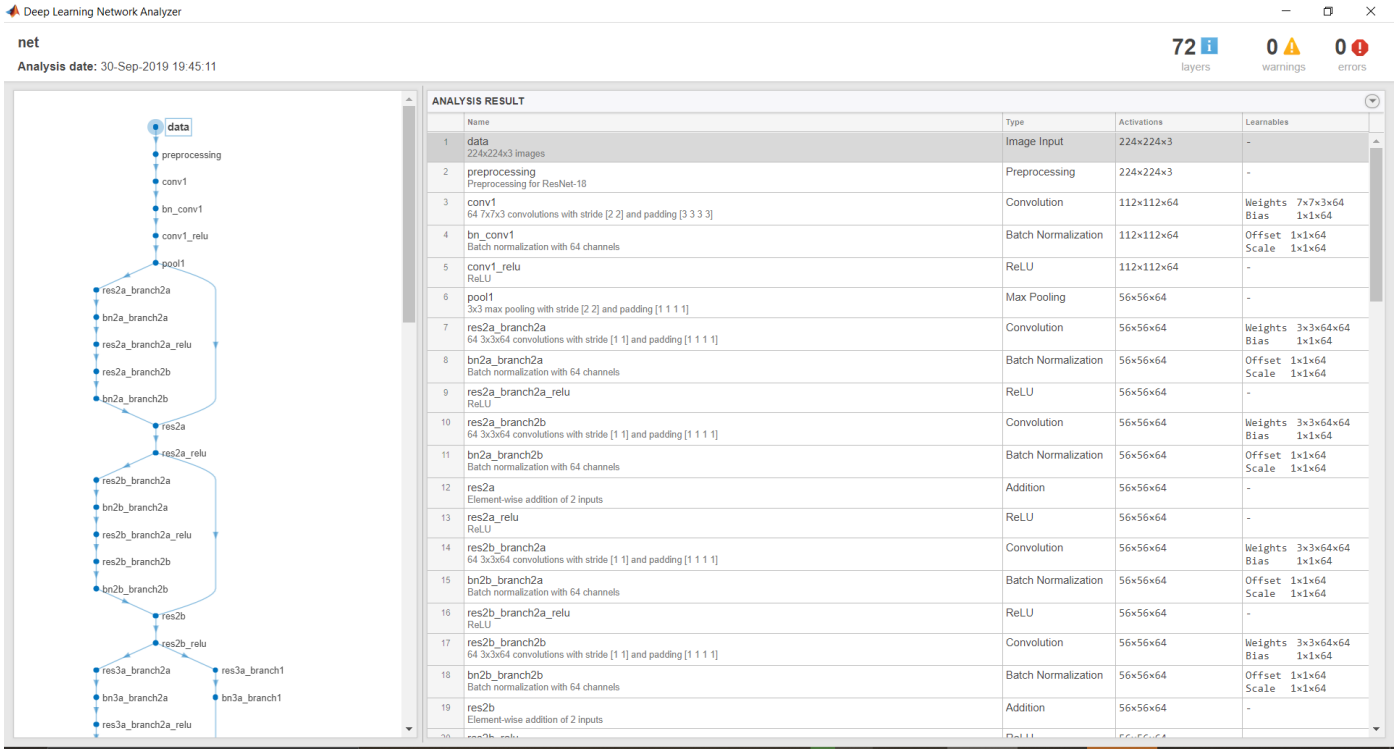


Figura 5.2.5.1 CNN analizada por Matlab

Utilizando el primer algoritmo se obtuvieron los siguientes resultados

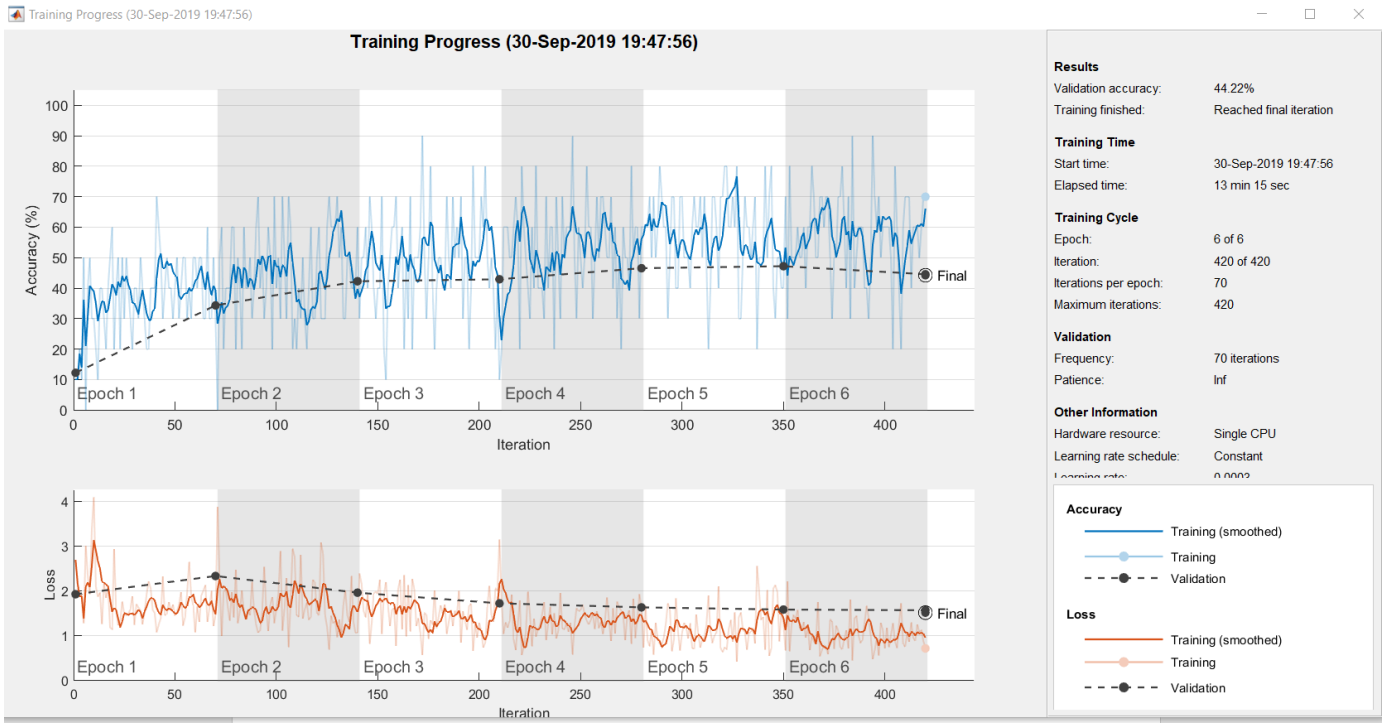


Figura 5.2.5.2 Resultados de la ejecución de Resnet18 con banco de imágenes del Algoritmo 1 con una validación 44.22% y un tiempo de 13 min 15 seg.

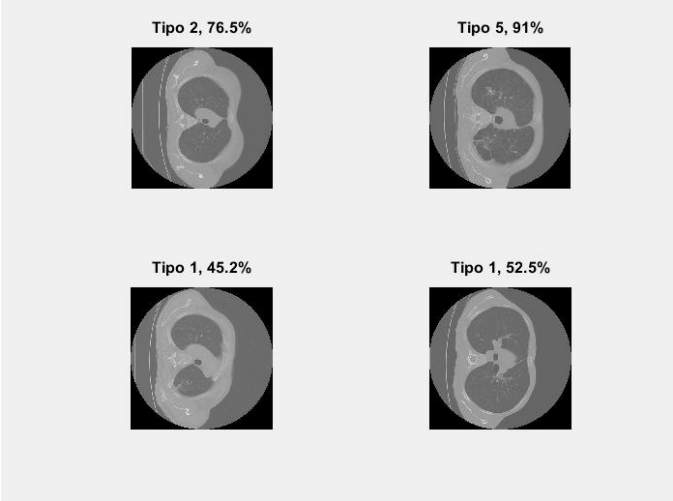


Figura 5.2.5.3 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 1.

Utilizando el segundo algoritmo se obtuvieron los siguientes resultados

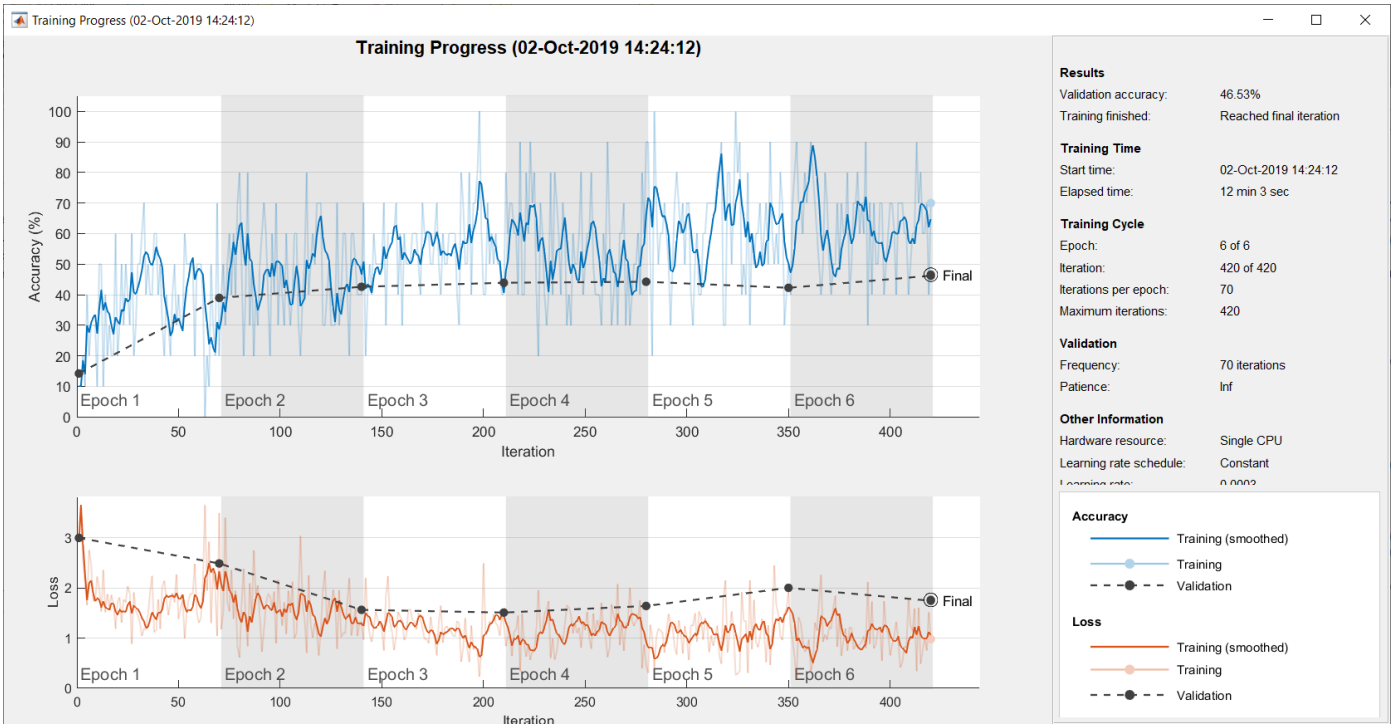


Figura 5.2.5.4 Resultados de la ejecución de Resnet18 con banco de imágenes del Algoritmo 2 con una validación de 46.53% y un tiempo de 12 min 3 seg.

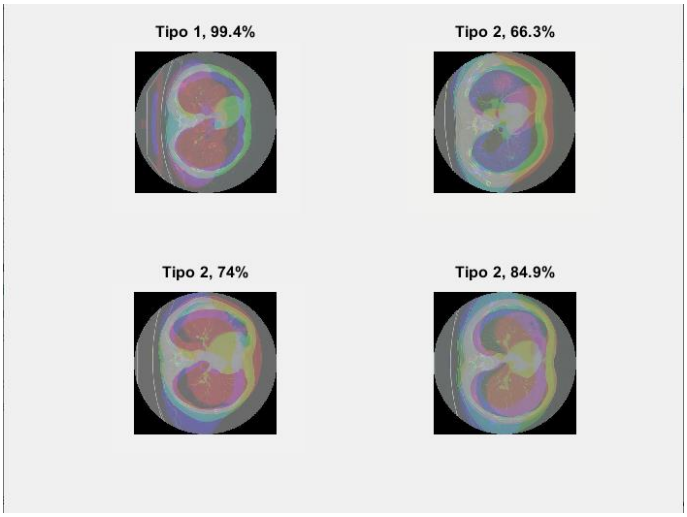


Figura 5.2.5.5 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes 2.

Utilizando banco de imágenes de 20 pacientes con todas las imágenes de las tomografías.

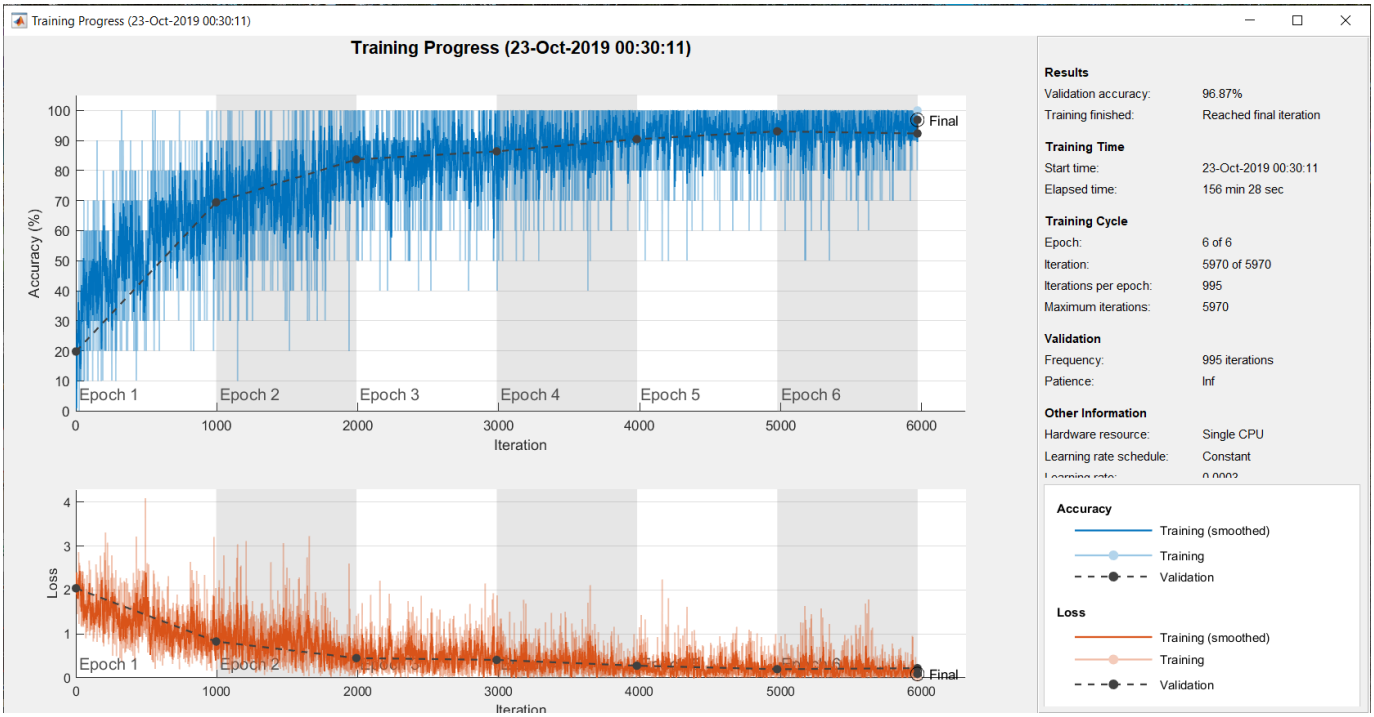


Figura 5.2.5.6 Resultados de la ejecución de Resnet con banco de imágenes del Algoritmo con una validación 96.87% y un tiempo de 156 min y 28 seg.

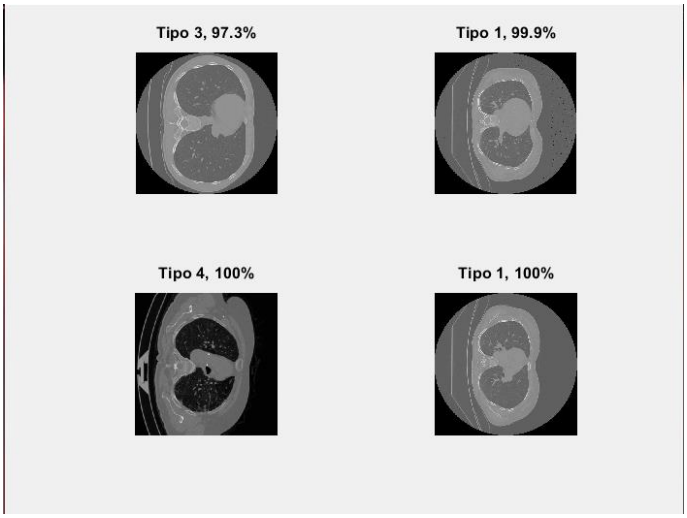


Figura 5.2.5.7 Ejemplos aleatorios por tipo y porcentaje de validación de banco de imágenes.

5.3 Implementación de pruebas

Tomando como referencia la arquitectura Resnet que obtiene el mejor resultado, se propuso hacer una prueba adicional. Se eliminan aproximadamente entre 15% y 20% de las imágenes con información no relevante para las CNN y dejar solamente las imágenes centrales de la tomografía donde está contenida la información más relevante de la Tuberculosis, como lo muestra en la representación de la figura 5.3.

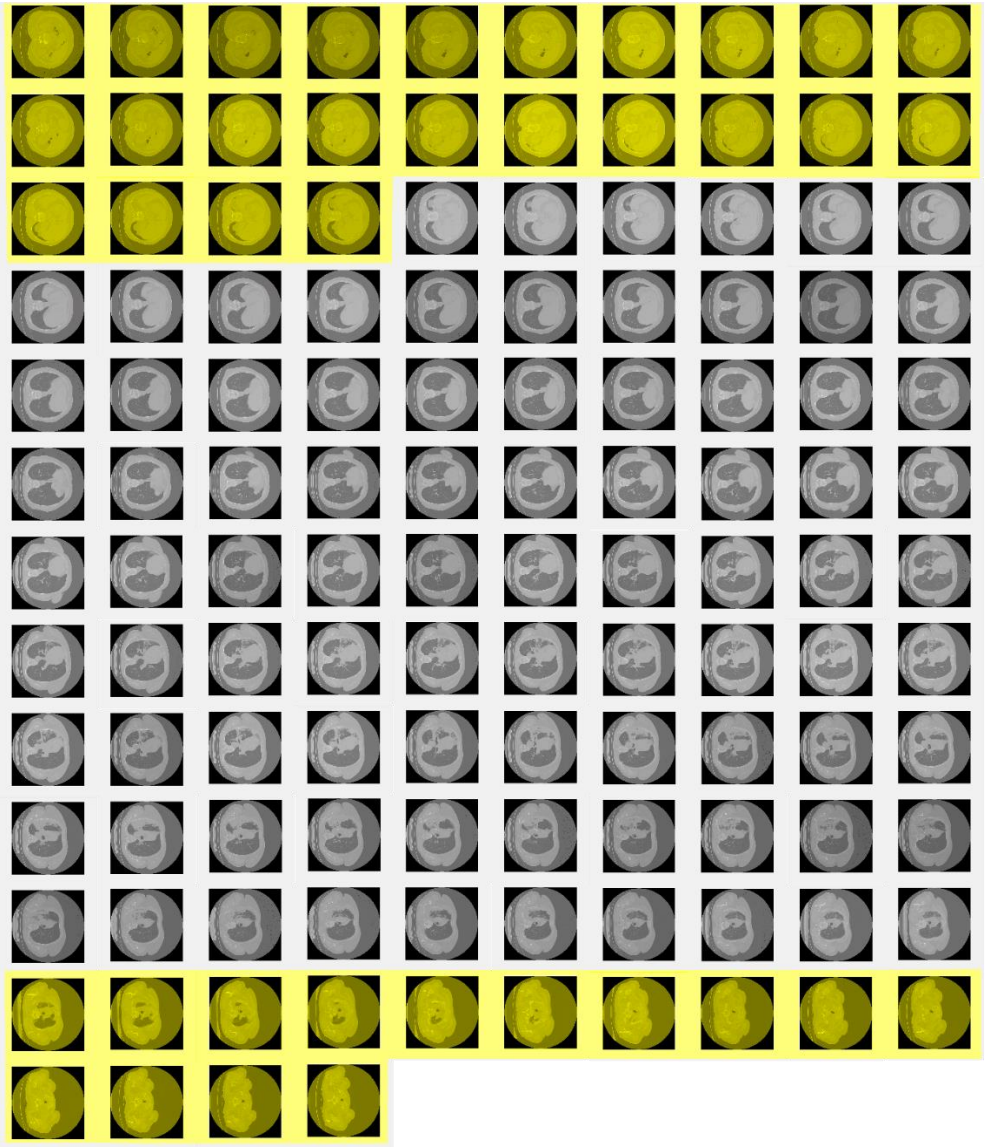


Figura 5.3 Representación de un paciente y en tonalidad amarilla las imágenes con información no relevante que fueron omitidas.

La figura 5.3.1 muestra el resultado del reentrenamiento de la red Resnet al eliminar entre el 15 y el 20 por ciento de las imágenes por pacientes. Así en lugar de tener 12,445 imágenes en la primera prueba se utilizan únicamente 9,589 en esta nueva prueba.

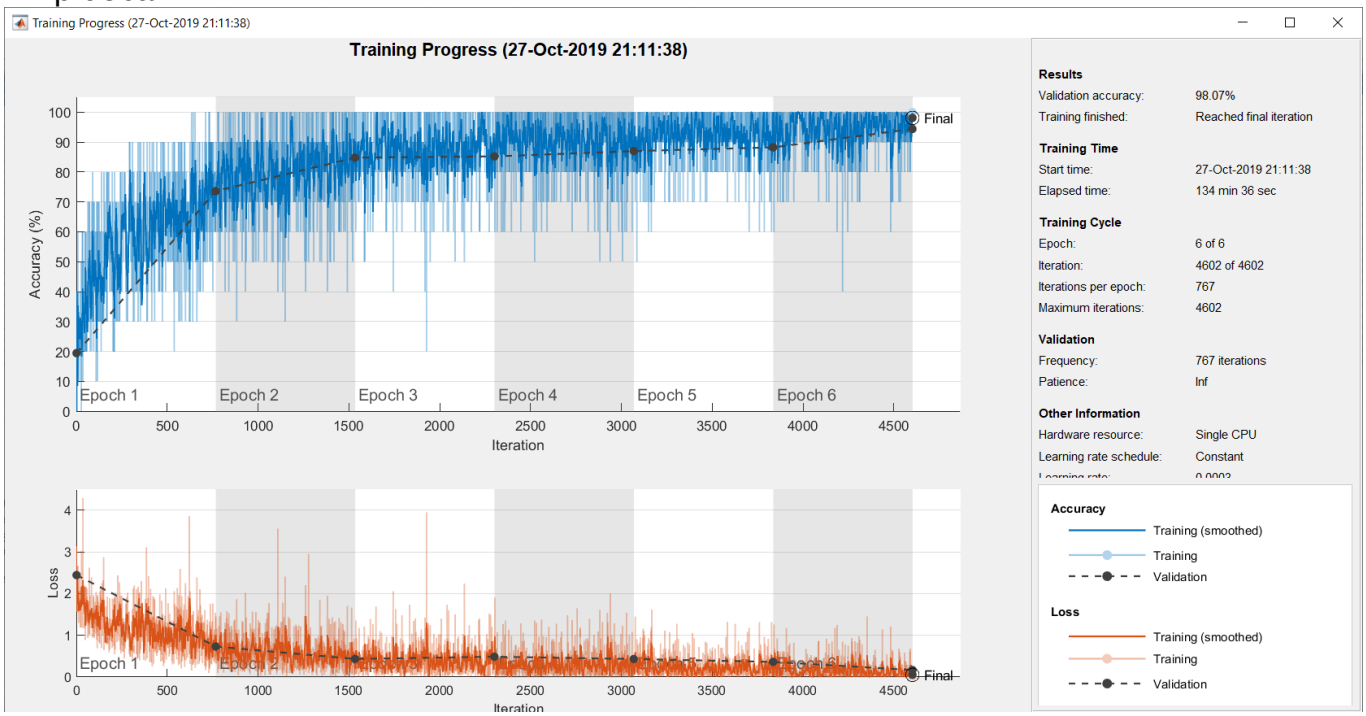


Figura 5.3.1 Resnet aplicando una reducción de imágenes del 15 al 20% por cada paciente con una validación de 98.07% y un tiempo de 134 min y 36 seg.

Aplicando esta prueba se llegó al resultado de 98.07% en 134 minutos y 36 segundos, lo que representa el mejor resultado de las pruebas realizadas. Visto desde otro punto de vista el ingresar a la CNN los cortes de las imágenes intermedias facilita a la red neuronal el procesamiento de las mismas mejorando los índices de clasificación y con menos costoso computacional.

Matriz de confusión

Una matriz de confusión es una tabla que se usa para describir el rendimiento de un modelo de clasificación (o "clasificador") en un conjunto de datos de prueba para los que se conocen los valores verdaderos.

Para obtener la matriz de confusión y evaluar el clasificador con datos no utilizados en las pruebas de entrenamiento y evaluación anteriores, se tomaron como referencia de entrada un nuevo conjunto de imágenes, tomando 10 nuevos pacientes por cada tipo obteniendo la siguiente tabla generada por la función *confusionchart* de MatLab.

En la figura 5.3.2, se muestra la matríz de confusión clasificando todas las imágenes de las tomografías de 10 pacientes por tipo de tuberculosis.

Figura 5.3.2 Matriz de confusión obtenida al evaluar el clasificador con todas las imágenes de 10 nuevos pacientes, el número indicado por fila representa al tipo de cada TBT y por columna el resultado del clasificador. En la prueba se utilizaron 179,161,172,170 y 190 imágenes por tipo de tuberculosis de I al V respectivamente.

En la figura 5.3.2, además, se puede observar que a pesar de que el clasificador tiene una validación de 98.07% (ver figura 5.3.1) al evaluar de forma individual cada una de

las imágenes de las tomografías de los nuevos pacientes esta no puede obtener un resultado elevado de aciertos ya que se generan muchos falsos positivos.

En la práctica sería mejor reportar el tipo de tuberculosis por paciente. En la figura 5.3.3 se muestra la matriz de confusión por paciente. Para obtener el tipo de tuberculosis de cada paciente se clasifican cada una de las imágenes de la tomografía y se obtiene el tipo de tuberculosis determinada por la CNN y se reporta el tipo con mayor número de imágenes clasificadas.

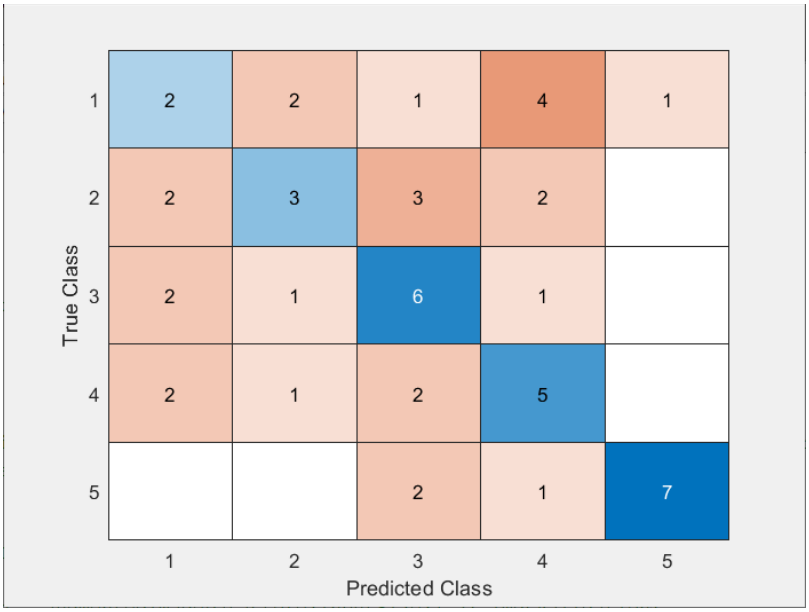


Figura 5.3.3 Matriz de confusión de los 10 nuevos pacientes ingresados al clasificador. La diagonal representa el número de pacientes correctamente clasificados. Cada etiqueta de las filas corresponde al tipo de TBT y en las columnas se muestra el resultado del clasificador.

Al analizar los resultados de esta última prueba se concluye que la CNN tiene problemas para clasificar correctamente los tipos I y II de tuberculosis. Esto se podría deber a que las imágenes asociadas a estos tipos de tuberculosis no son muy distintos en relación a los otros tipos de tuberculosis. Sin embargo, los últimos tipos de tuberculosis son mejor clasificados por la CNN.

Esto hace evidente la necesidad de realizar en un futuro otras pruebas validación del clasificador ya que las pruebas de rendimiento son las que hacen evidentes los resultados reales de los clasificadores.

Al obtener todas las gráficas de los algoritmos utilizados y de todas las CNN entrenadas con los mismos se obtuvo la siguiente tabla comparativa, resaltando los mejores resultados para cada una.

CNN	Algoritmo 1		Algoritmo 2		20 pacientes	
Alexnet	46.53%	8 min 15 seg	**48.18%**	7 min 46 seg	87.75%	91 min 9 seg
Googlenet	46.20%	12 min 59 seg	40.92%	22 min 16 seg	90.04%	214 min 12 seg
Vgg16	**50.17%**	62 min 53 seg	37.29%	62 min 41 seg	91.84%	791 min 45 seg
Vgg19	42.90%	75 min 47 seg	42.57%	82 min 34 seg	96.87%	895 min 28 seg
Resnet	44.22%	13 min 25 seg	46.53%	12 min 3 seg	**96.95%**	156 min 28 seg

Figura 5.3.4 Tabla comparativa entre algoritmo 1 y algoritmo 2 aplicando cada una de las CNN

La información visualizada nos refleja que el mayor porcentaje de clasificación se obtiene por Resnet pero en costo computacional fue de los más altos. El algoritmo 1 Vgg16 fue la que mejor rendimiento mostró a pesar de que es la predecesora de las demás redes junto con Googlenet. Para el algoritmo 2 el que mayor porcentaje tuvo fue Alexnet. En la prueba con 20 pacientes Resnet alcanzo el mejor resultado si no consideramos el costo computacional.

5.4 Conclusiones y Trabajo Futuro.

En este documento se ha demostrado la utilidad de las CNN para la clasificación de imágenes de tomografías con el fin de desarrollar una herramienta de apoyo para el diagnóstico de los 5 tipos de tuberculosis, además, se han alcanzado los siguientes objetivos:

- Implementar una red neuronal convolucional para la clasificación de cinco tipos (Infiltrativo, Focal, Tuberculoma, Miliar, Fibrocavernoso) de tuberculosis mediante el uso de técnicas de aprendizaje profundo.

  Se ha logrado implementar cinco CNN para la clasificación de cinco clases mediante el uso de la técnica de transferencia de conocimiento y reentrenamiento de las últimas 3 capas, siendo el mejor resultado de 96.95% de precisión del proceso de toma de 20 pacientes por tipo de TBT bajo la arquitectura Resnet, también se pueden tomar como resultados interesantes los vistos en el algoritmo 2 bajo la arquitectura Alexnet teniendo el mejor rendimiento de 7 min 46 seg, tomando en cuenta que el número de imágenes de muestra fueron de 1008 imágenes.

- Experimentar los resultados de utilizar imágenes en escala de grises obtenidas directamente de los datos contenidos en las tomografías y los que al aplicar un tratamiento para obtener una imagen sintética en RGB variaron totalmente en las pruebas con la misma CNN, siendo la más evidente VGG16 con 50.17% para el algoritmo 1 y 37.29% para el algoritmo 2.
- Comparar los resultados obtenidos del clasificador con los resultados vistos en algunos artículos similares. Ya que estos hablan sobre radiografías vistas frontalmente y a los datos obtenidos de tomografías con una cantidad mayor de información para procesar, ya sean generando imágenes sintéticas o tomado por volúmenes de imágenes.

	Algoritmo 1	Algoritmo 2	20 pacientes
Alexnet	46.53	48.18	87.75
Googlenet	46.20	40.92	90.04
Vgg16	50.17	37.29	91.84
Vgg19	42.90	42.57	96.87
ResNet	44.22	46.53	96.95
Alexnet Sermanet. [SEZ + 13]		86.70	
Googlenet Sermanet. [SEZ + 13]		90.20	
Vgg16 Simonyan. [SZ15]		90.00	
Alexnet Hwang. [HKJK16]		78.80	
Alexnet Shin. [SRG + 16]		76	
Googlenet Pascal VOC 2007 [EEZ + 06]		73.9	

Figura 5.4 Tabla comparativa entre representaciones de autores sobre radiografías y resultados obtenidos por tomografías.

Cabe mencionar que en la siguiente tabla los valores analizados son tomados de los artículos que hablan sobre radiografías las cuales, aunque representan el mismo problema a desarrollar es visto desde un enfoque diferente en el cual la información está contenida en una sola imagen.

Por último, al realizar la comparativa entre los 2 algoritmos propuestos y una tercer propuesta que es utilizar únicamente las imágenes del volumen central de las tomografías los resultados fueron notablemente superiores, las CNN trabajan con mayor certeza con grandes cantidades de información y desechando las imágenes con características similares.

Como trabajo futuro sería interesante estudiar técnicas avanzadas de tratamiento de imágenes para resaltar aún más las características principales de cada enfermedad para mejorar las imágenes que se procesan por la CNN con la finalidad de reducir el porcentaje de error que actualmente se obtiene. Además, se podría profundizar en la evaluación del clasificador utilizando otras bases de imágenes que incluyan tomagrafías de sujetos sanos, porque en la práctica se sabe que una red debe ser entrenada y evaluada con miles de imágenes.

Bibliografía

Armato, S., Giger, M., & MacMahon., H. (1998). *Automated lung segmentation in digitized posteroanterior chest radiographs.* Academic Radiology.

Boo, D. W., Prokop, M., Uffmann, M., Ginneken, B. v., & Schaefer-Prokop., C. M. (2009). *Computer-aided detection (CAD) of lung nodules and small tumours on chest radiographs.* European Journal of Radiology.

Burges., C. J. (1998). *A Tutorial on Support Vector Machines for Pattern Recognition.* Data mining and knowledge discovery.

Chollet, F. (2015). *Keras Documentation,*. Recuperado el 23 de 09 de 2019, de https://keras.io

Cohen., J. P. (s.f.). *Visualizing CNN architectures side by side with mxnet*. Recuperado el 23 de 09 de 2019, de http://josephpcohen.com/w/visualizing-cnn-architectures-side-by-side-with-mxnet/

Dataset., D. I. (2018). *Diagnostic Imaging Dataset Annual Statistical. Technical report*. Recuperado el 23 de 09 de 2019, de http://www.wjgnet.com/1949-8470/full/v6/i1/1.html

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). *ImageNet: A large-scale hierarchical image database.* IEEE Conference on Computer Vision and Pattern Recognition.

Everingham, M., Zisserman, A., & Williams, C. (2006). *The PASCAL visual object classes challenge 2006 (VOC2006) results. Workshop in ECCV06.* Austria: Graz.

Fellbaum, C. (1998). WordNet: an electronic lexical database. *MIT Press*.

Frangi, A., Niessen, W., Vincken, K., & Viergever, M. (1998). *Multiscale vessel enhancement filtering Medical Image Computing and Computer- Assisted Intervention.* MICCAI.

Ginneken, B. v., Setio, A. A., Jacobs, C., & Ciompi, F. (2015). *Off-the-shelf convolutional neural network features for pulmonary nodule detection in computed tomography scans.* In 2015 IEEE 12th International Symposium on Biomedical Imaging (ISBI).

Ginneken, B. v., Stegmann, M. B., & Loog, M. (2006). *Segmentation of anatomical structures in chest radiographs using supervised methods: A comparative study on a public database.* Medical Image Analysis.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning–book. *MIT Press*, 521(7553):800.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition.* IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Hogeweg, L., Sánchez, C. I., Maduskar, P., Philipsen, R., Story, A., Dawson, R., . . . Ginneken, B. V. (2015). *Automatic detection of tuberculosis in chest radiographs using a combination of textural and shape abnormality analysis.* IEEE Transactions on Medical Imaging.

Huang, F. J., & LeCun, Y. (2006). *Large-scale learning with SVM and convolutional nets for generic object categorization* (Vol. 1). In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern.

Hwang, S., Kim, H.-E., Jeong, J., & Kim, H.-J. (2016). *A novel approach for tuberculosis screening based on deep convolutional neural networks.* SPIE Medical Imaging.

INSP, W. (2019). *Día Mundial de la Tuberculosis 2019*. Recuperado el 22 de 11 de 2019, de https://insp.mx/avisos/4904-tuberculosis-dia-mundial.html

Jaeger, S., Candemir, S., Antani, S., Wang, Y.-X. J., Lu, P.-X., & Thoma, G. (2014). *Two public chest X-ray datasets for computer-aided screening of pulmonary diseases.* Quantitative imaging in medicine and surgery.

Jaeger, S., Karargyris, A., Candemir, S., Folio, L., Siegelman, J., Callaghan, F., . . . McDonald, C. J. (2014). *Automatic tuberculosis screening using chest radiographs.* IEEE Transactions on Medical Imaging.

Jones, E., Oliphant, T., & Peterson, P. (2001). *SciPy: Open source scientific tools for Python*. Recuperado el 23 de 09 de 2019

Krizhevsky, A., Sutskever, I., & (NIPS2012), H. G. (2012). *ImageNet Classification with Deep Convolutional Neural Networks.* Advances in Neural Information Processing Systems.

Lakhani, P., & Sundaram, B. (2017). *Deep Learning at Chest Radiography: Automated Classification of Pulmonary Tuberculosis by Using Convolutional Neural Networks.* Radiology.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). *Gradientbased learning applied to document recognition.* Proceedings of the IEEE.

Leung, A. N. (1999). *Pulmonary tuberculosis: the essentials.* Radiology.

Li, F.-F., Karpathy, A., & Johnson, J. (2008). *CS231n Convolutional Neural Networks for Visual Recognition.* Recuperado el 23 de 09 de 2019, de http://cs231n.github.io/

Maduskar, P., Muyoyeta, M., Ayles, H., Hogeweg, L., Peters-Bax, L., & Ginneken, B. v. (2013). *Detection of tuberculosis using digital chest radiography: automated reading vs. interpretation by clinical officers.* The international journal of tuberculosis and lung disease : the official journal of the International Union against Tuberculosis and Lung Disease.

Mason, D. (2011). *Pydicom: An Open Source DICOM Library.* Medical Physics.

McConnell, R. K. (1986). *Method of and apparatus for pattern recognition*. Recuperado el 23 de 09 de 2019, de http://www.google.co.uk/patents/US4567610

Melendez, J., Ginneken, B. v., Maduskar, P., Philipsen, R. H., Ayles, H., & Sanchez, C. I. (2016). *On Combining Multiple- Instance Learning and Active Learning for Computer-Aided Detection of Tuberculosis.* IEEE Transactions on Medical Imagin.

Mildenberger, P., Eichelberg, M., & Martin, E. (2002). *Introduction to the DICOM standard.* European Radiology.

Ojala, T., Pietikainen, M., & Harwood, D. (1994). *Performance evaluation of texture measures with classification based on Kullback discrimination of distributions* (Vol. 1). In Proceedings of 12th International Conference on Pattern Recognition.

Oliphant, T. E. (2007). *Python for scientific computing.* Computing in Science and Engineering.

Pages, U. N. (2016). *The Sustainable Development Goals Report*. Recuperado el 23 de 09 de 2019, de http://www.un.org/sustainabledevelopment/sustainable-development-goals

Pedregosa, F., & Varoquaux, G. (2011). *Scikit-learn: Machine learning in Python, volume 12*. Recuperado el 23 de 09 de 2019, de http://dl.acm.org/citation.cfm?id=2078195

Razavian, A. S., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). *CNN features off-the-shelf: An astounding baseline for recognition.* In IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). *OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.* arXiv preprint arXiv.

Shin, H. C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., . . . Summers, R. M. (2016). *Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning.* IEEE Transactions on Medical Imaging.

Shlens, J. (2018). *Research Blog: Train your own image classifier with Inception in TensorFlow*. Recuperado el 23 de 09 de 2019, de https://research.googleblog.com/2016/03/train-your-own-image-classifier-with.html

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2015). *Going deeper with convolutions* (Vols. 07-12). In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). *Rethinking the Inception Architecture for Computer Vision.*

Tang, Y. (2013). *Deep Learning using Linear Support Vector Machines*. Recuperado el 23 de 09 de 2019, de Deeplearning.Net: http://deeplearning.net/wp-content/uploads/2013/03/dlsvm.pdf

Team, M. (2018). *Pretrained Deep Neural Networks*. Recuperado el 23 de 09 de 2019, de https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html

Team, T. D. (2016). *Theano: A Python framework for fast computation of mathematical expressions.* arXiv e-prints.

Walt, S. V., Colbert, S. C., & Varoquaux, G. (2011). *The NumPy array: A structure for efficient numerical computation.* Computing in Science and Engineering.

WHO. (2016). *WHO Global tuberculosis report 2016. Technical report*. Recuperado el 23 de 09 de 2019, de http: //www.who.int/tb/publications/global{_}report/en/

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). *How transferable are features in deep neural networks?* (Vol. 2). Proceedings of the 27th International Conference on Neural Information Processing Systems.