

***Deep Health:
Diagnóstico inteligente de
Neumonía Bacteriana o
Vírica***

Copyright

This document is Copyright © 2020 by its contributors as listed below. You may distribute it and/or modify it under the terms of either the GNU General Public License (<http://www.gnu.org/licenses/gpl.html>), version 3 or later, or the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0/>), version 3.0 or later.

All trademarks within this guide belong to their legitimate owners.

Colaboradores

Miguel Salinas: versión español

Realimentación

Por favor, dirija cualquier comentario o sugerencia sobre este documento a: info@thingtrack.com

Fecha de publicación y versión del software

Publicado 21 Marzo 2020.

Contenidos

Copyright.....	2
Presentación.....	4
Fuente de datos: Dataset.....	7
Modelo: Redes Neuronales Convolucionales.....	8
Entrenamiento: Transfer Learning (TL).....	9
Depurando: TensorBoard.....	13
Visualización: Una visión gráfica del CNN.....	16
Predicción: La herramienta Deep Health.....	18
Arquitectura.....	20
Código Fuente: Servicios.....	22
Resumen.....	24

Presentación

Actualmente la pandemia de Covid-19 está provocando una sangría de muertes a nivel mundial por ser un nuevo virus para el cual aún no tenemos ninguna vacuna eficiente que posibilite el control del mismo. Sin embargo los sistemas de detección precoz de este virus si está dando fruto a nivel global. Actualmente el diseño de kits que permitan la detección precoz están teniendo sus frutos.

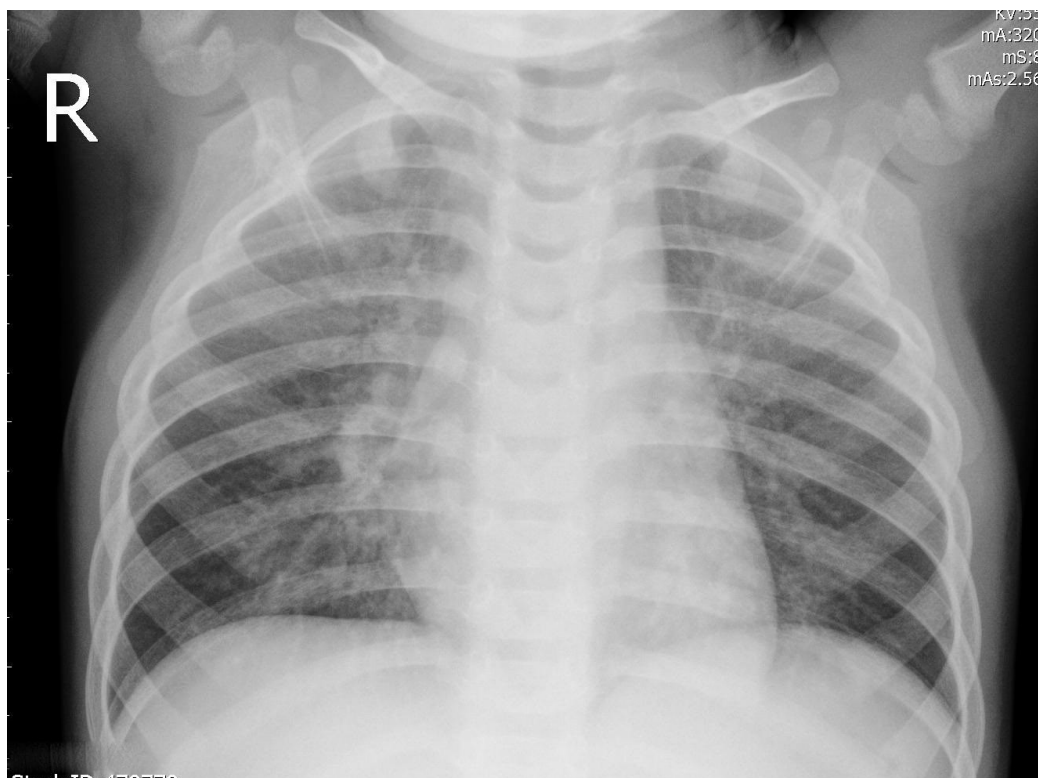
Dentro de la sintomatología de todos los pacientes que han contraído esta enfermedad tiene como denominación común que todos ellos se ven afectados por una neumonía que en algunos casos el agravamiento de la misma, provoca la muerte especialmente en grupos de riesgos: gente mayor o con dolencias previas que puedan agravar este tipo de neumonías de carácter víricas.

Actualmente el uso de kits basándose en técnicas del tipo PCR en tiempo real permiten que la posible detección del virus en fluidos como fosas nasales, o boca, puedan ser detectados a tiempo. Esta técnica que se basa en poder replicar el ARN del virus en grandes cantidades para poder así detectar la presencia de éste con niveles de certeza de al rededor del 98% son las más eficaces a día de hoy. Sin embargo también sabemos que el proceso de fabricación de estos kits, heterogéneos junto a unos mercados saturados por la necesidad global de los mismos, unidos a la necesidad de tener equipos muy especializados en detectar estos virus junto al coste logístico de llevar estas pruebas a los centros de salud u hospitales hacen que esta técnica no pueda ser lo rápido que todos deseamos. Por ello la creación de un sistema capaz de detectar de forma precoz esta enfermedad a una velocidad rápida, es actualmente una necesidad.

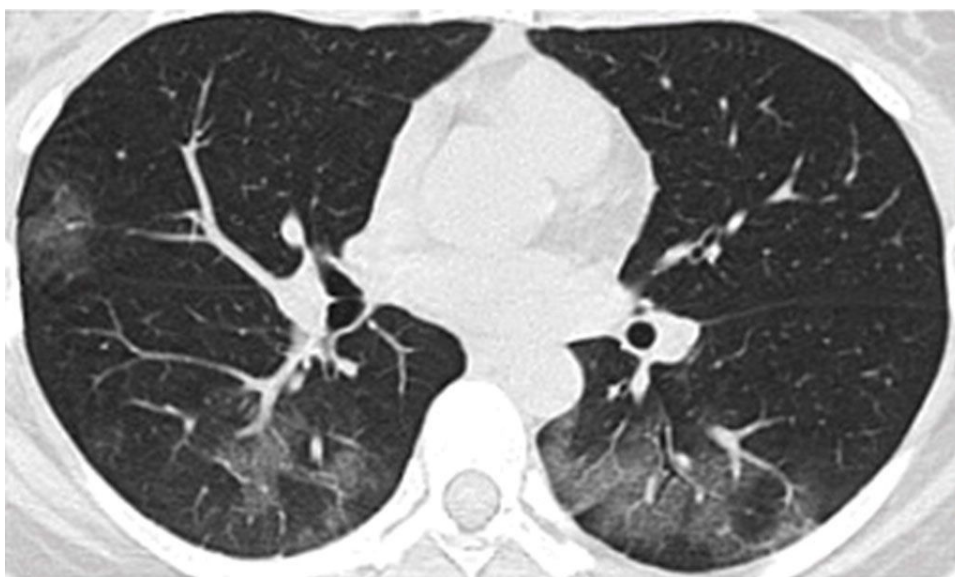
Por estas razones pensamos que apoyarse en técnicas de inteligencia artificial junto al uso de placas de rayos X de torax de fácil obtención en cualquier centro de salud u hospital pueden ser una herramienta de apoyo a los kits que actualmente se están utilizando.

A día de hoy, la detección de neumonías es un proceso manual que

requiere que un clínico capacitado examine y evalúe las placas radiográficas para cada uno de los pacientes afectados.



Placa de Rayos X de torax con neumonía vírica



Placa tomográfica con neumonía Covid-19

La necesidad de un método completo y automatizado de detección de Neumonía, principal causa de mortalidad del virus Covid-19, hacen necesario encontrar otros métodos de detección que ayuden a detectar neumonías provocadas por el Covid-19.

Como en todo proceso de Aprendizaje Automático (Machine Learning) debemos de partir de un Dataset o conjunto de datos representativo del problema que queremos modelizar. En nuestro caso hemos partido de un dataset formado por 5863 placas de Rayos X en formato JPEG divididas en tres categorías: No Neumonía, Neumonía bacteriana y Neumonía Vírica.

Se seleccionaron imágenes de rayos X de Tórax (posterior) procedentes de pacientes pediátricos de uno a cinco años procedentes del Centro Médico de Mujeres y Niños de Guangzhou, Guangzhou. Todas las imágenes de rayos X del tórax se realizaron como parte de la atención clínica de rutina de los pacientes.

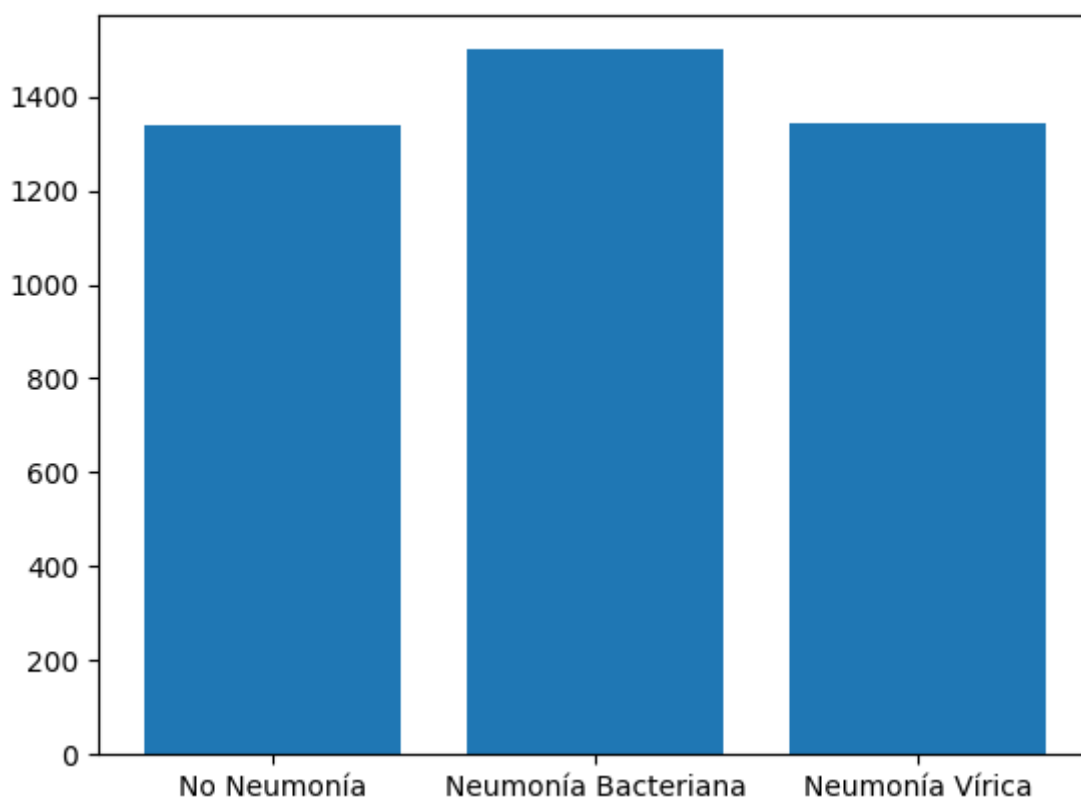
Los diagnósticos de las imágenes fueron calificados por dos médicos expertos. Para tener en cuenta los errores de calificación, un tercer experto también verificó el conjunto de evaluación.

Fuente de datos: Dataset

La fuente de datos de la que hemos partido consta de **5863 imágenes de Rayos X de Torax** con alta resolución, **clasificadas en 3 niveles** dentro de la enfermedad:

- 0: **No_Neumonía**: nivel sano.
- 1: **pneumonia bacteria**: afectado por neumonía bacteriana.
- 2: **pneumonia virus**: afectado por neumonía vírica.

El histograma del dataset utilizado es el siguiente:



Modelo: Redes Neuronales Convolucionales

Dentro del aprendizaje automático (Machine Learning), se ha optado por utilizar modelos inteligentes profundos o también llamados Deep Learning Models, por ser estos los más aptos a la hora de clasificar todo tipo de imágenes. En concreto dentro de los modelos de Deep Learning se ha escogido la familia llamada **Redes Neuronales Convolucionales (CNN)**, por ser estos modelos los que alcanzan niveles de precisión más altos a la hora de reconocer patrones y clasificar imágenes complejas como la que nos hemos encontrado en el diagnóstico de la Neumonía del Covid-19.

El modelo presentado es un modelo que no ha sido entrenado inicialmente con todo el Dataset, dado que el tiempo y recurso necesarios para entrenar una red de estas características sería enorme y el número de ejemplos de los que disponemos nos es muy alto.

Realmente la metodología seguida por nosotros se llama **Transferencia de Conocimiento o Retrain (Reentrenamiento)**, que consiste en seleccionar una red neuronal de **tipo CNN (Convolution Neural Networks)**, ya entrenada capaz de reconocer un gran número de patrones, aprovechando su capacidad cognitiva profunda y sustituir únicamente aquella capa de la red neuronal con menor capacidad de generalización, a la hora de reconocer los patrones de Neumonía que nos ocupa.

La red neuronal base escogida por nosotros ha sido **Inception V3** desarrollada por Google, capaz de clasificar más de un millón de imágenes en cientos de clases. A continuación lo que se va a hacer es reutilizar la capacidad cognitiva, existente en todas las capas profundas de la red neuronal, sustituyendo la penúltima capa del modelo especializada en clasificar solamente imágenes de tipo placas de Rayos X de Torax. Con esto ahorramos muchos recursos de entrenamiento, alcanzando unos niveles de certidumbre muy aceptables si los comparamos igualmente con otras metodologías.

Entrenamiento: Transfer Learning (TL)

En esta fase vamos a desarrollar la técnica de **Transfer Learning (TL)** reentrenando una red CNN con nuestro Dataset de imágenes médicas de Torax.

La técnica de TL representa la capacidad de crear un modelo clasificador CNN de imágenes adquiriendo el conocimiento de otro modelo capaz de clasificar otro conjunto diferente de imágenes. Esta técnica es muy importante, pues partiendo de modelos CNN entrenados con datasets muy grandes utilizando recursos computacionales extensos, somos capaces de aprovechar este trabajo para crear nuevos modelos que sean capaces de aprender sobre nuevos datasets utilizando por contra unos recursos computacionales mucho menores, ya que gran parte del conocimiento adquirido ha sido transferido del modelo base.

Para acelerar el proceso de entrenamiento. Hemos normalizado un nuevo Dataset y lo hemos creado a partir de los tres niveles de Neumonía antes nombrados, pero escogiendo aproximadamente unas 1400 imágenes por grupo, con el fin de que el Dataset resultante esté balanceado en relación al nivel de diabétes.

Para evitar el desbalanceo provocado por la característica de Neumonía Bacteriana hemos reducido el número de registros de 2530 a 1400. Finalmente nuestro Dataset de entrenamiento esta distribuido como sigue:

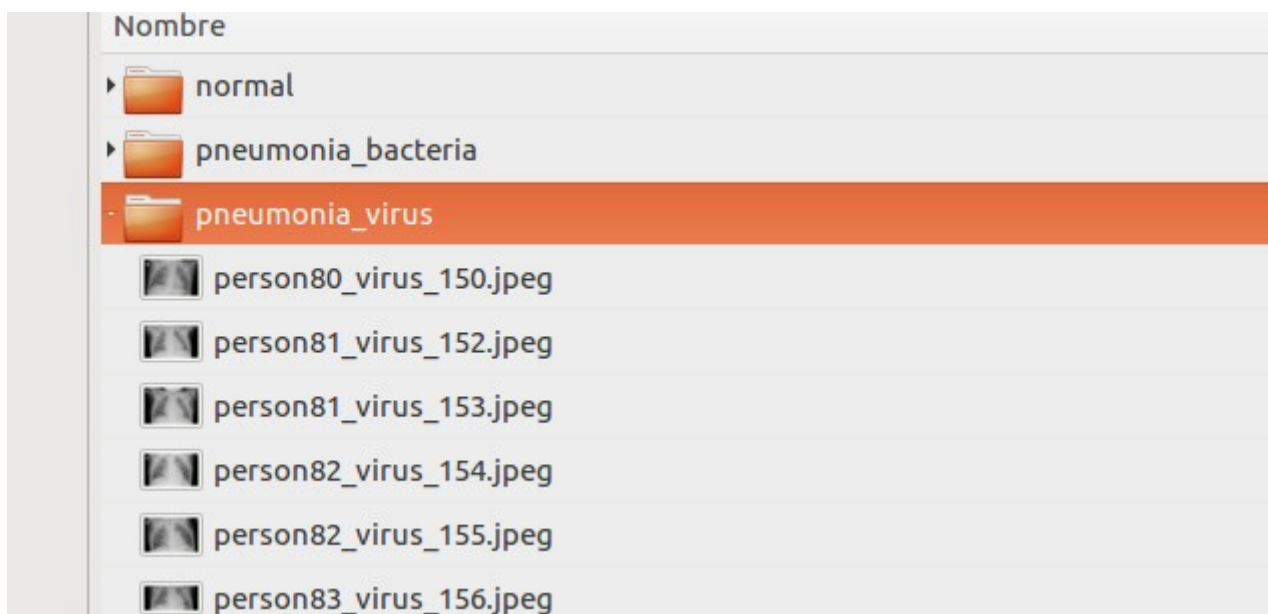
Nivel	Descripción	Número casos
0	Sano	1341
1	Neumonía Bacteriana	1400
2	Neumonía Vírica	1345

Técnicamente para desarrollar este nuevo modelo CNN lo desarrollaremos apartir de un script en python llamado **retrain.py**. Este script permite la

entrada de muchos parámetros, vamos a resumir los más importantes:

- **image_dir:** es el path en donde hemos distribuido previamente nuestras imágenes de entrenamiento, validación y test. El preprocesamiento previo, tiene como misión precisamente en clasificar todas nuestras imágenes en carpetas con nombre como exige en script de generación, es decir. En nuestro caso existirán tres carpetas:
 - **normal:** dataset con imágenes de torax de pacientes sin dolencia neumológica
 - **pneumonia_bacteria:** pacientes con síntomas de neumología bacteriana
 - **pneumonia_virus:** pacientes con síntomas de neumología vírica

Aquí podemos ver todas las carpetas:



- **output_graph:** destino del modelo entrenado. Valor por defecto es /tmp/output_graph.pb
- **how_many_training_steps:** número de etapas de entrenamiento o epochs en el proceso de entrenamiento. Por defecto con 4000
- **output_labels:** destino de las etiquetas del dataset entrenado. Valor por defecto: /tmp/output_labels.txt

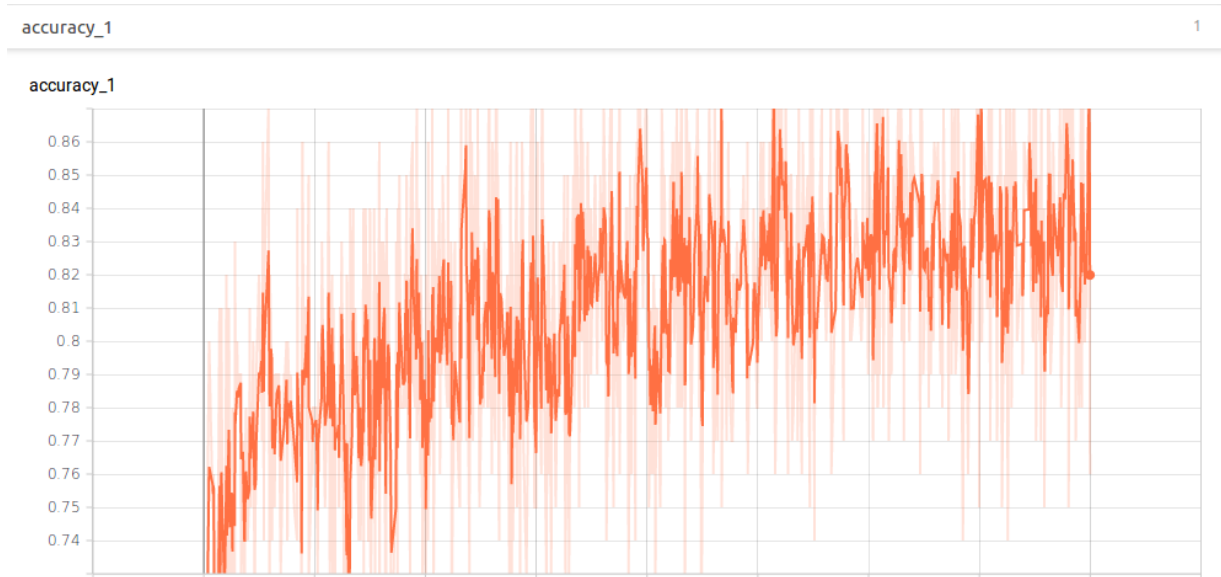
- **learning_rate:** Ratio de entrenamiento utilizado durante el proceso. Valor por defecto es 0.01
- **validation_percentage:** porcentaje de validación utilizado de nuestro dataset. Valor por defecto es 0.1
- **eval_step_interval:** Con que frecuencia evaluar los resultados del entrenamiento. Valor por defecto es 10
- **train_batch_size:** Cuantas imágenes entrenar a la vez. Valor por defecto 100
- **test_batch_size:** En cuántas imágenes probar. Este conjunto de prueba solo se usa una vez, para evaluar. La precisión final del modelo después de completar el entrenamiento. Un valor de -1 hace que se use todo el conjunto de prueba, lo que conduce a más resultados estables en las ejecuciones. Valor por defecto es -1
- **validation_batch_size:** Cuántas imágenes usar en un lote de evaluación. Este conjunto de validación es se usa con mucha más frecuencia que el conjunto de prueba y es un indicador temprano de cómo. La precisión del modelo es durante el entrenamiento. Un valor de -1 hace que se use todo el conjunto de validación, lo que conduce a resultados más estables en las iteraciones de entrenamiento, pero pueden ser más lentos en grandes conjuntos de entrenamiento. Valor por defecto 100
- **architecture:** Qué arquitectura de modelo usar. 'inception_v3' es el más preciso, pero también el más lento. Para modelos más rápidos o más pequeños, elija una MobileNet. Valor por defecto es inception_v3

Para ejecutar el proceso de entrenamiento ejecutamos este comando. Como podemos ver escogeremos la **arquitectura de inception_v3** obteniendo un modelo mas preciso a costa de gastar más recursos computacionales.

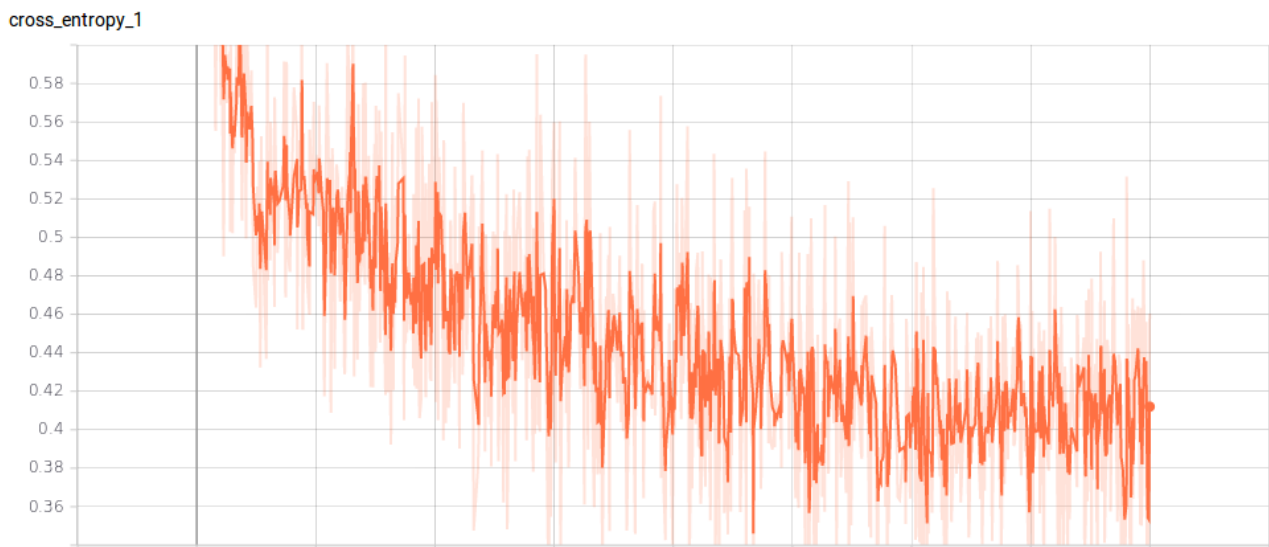
```
python3 retrain.py --image_dir ./Sources/deephealth-covid19/train_classified --architecture inception_v3 --saved_model_dir ./Sources/deephealth-covid19/model
```

Tras el entrenamiento el **nivel de precisión alcanzado niveles entorno al 83%.**

La evolución del **nivel de exactitud** del modelo durante el entrenamiento es:



La evolución de la **Función de coste (Cross Entropy)** del modelo durante el entrenamiento es:



Depurando: TensorBoard

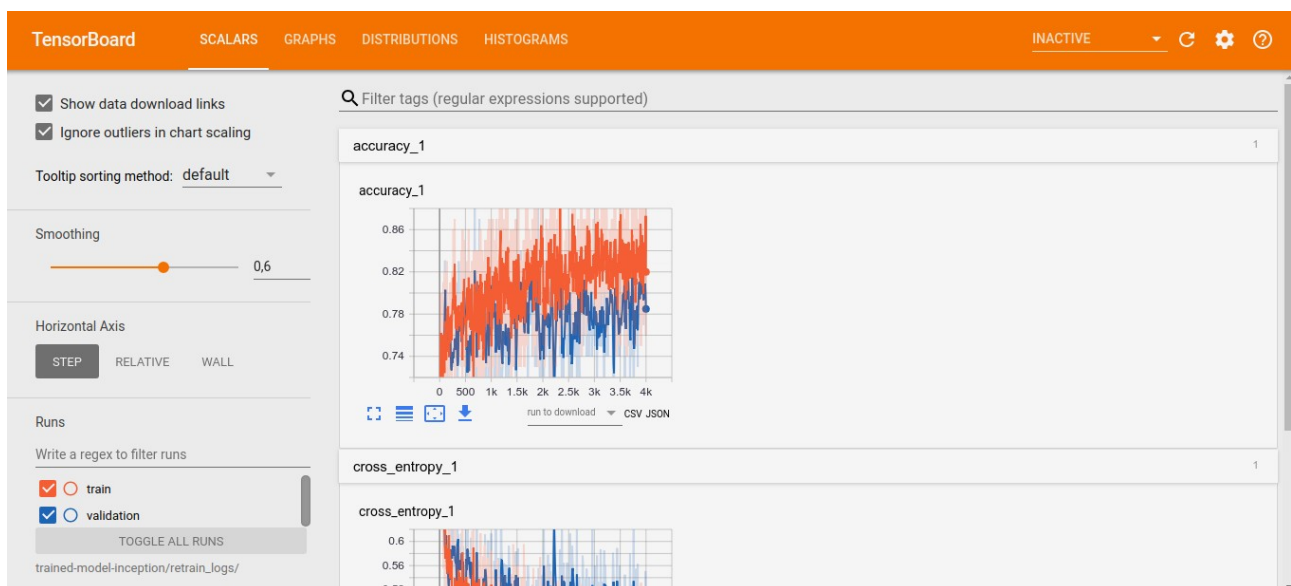
Como hemos comentado previamente, el proceso de entrenamiento por parte de TensorFlow de nuestro modelo CNN, genera muchos logs, que pueden representarse gráficamente para poder depurar el resultado del mismo desde la plataforma de Google llamada TensorBoard.

Para tener acceso a esta plataforma Web, simplemente debemos de instalar TensorBoard en nuestro equipo y ejecutar el siguiente comando pasando la carpeta de logs generada por TensorFlow como sigue:

```
tensorboard --logdir trained-model-inception/retrain_logs
```

Este comando da como resultado una URI, que podremos pegar en nuestro navegador para tener acceso a la plataforma de TensorBoard

TensorBoard 1.13.1 at <http://miguel-Inspiron-5570:6006> (Press CTRL+C to quit)

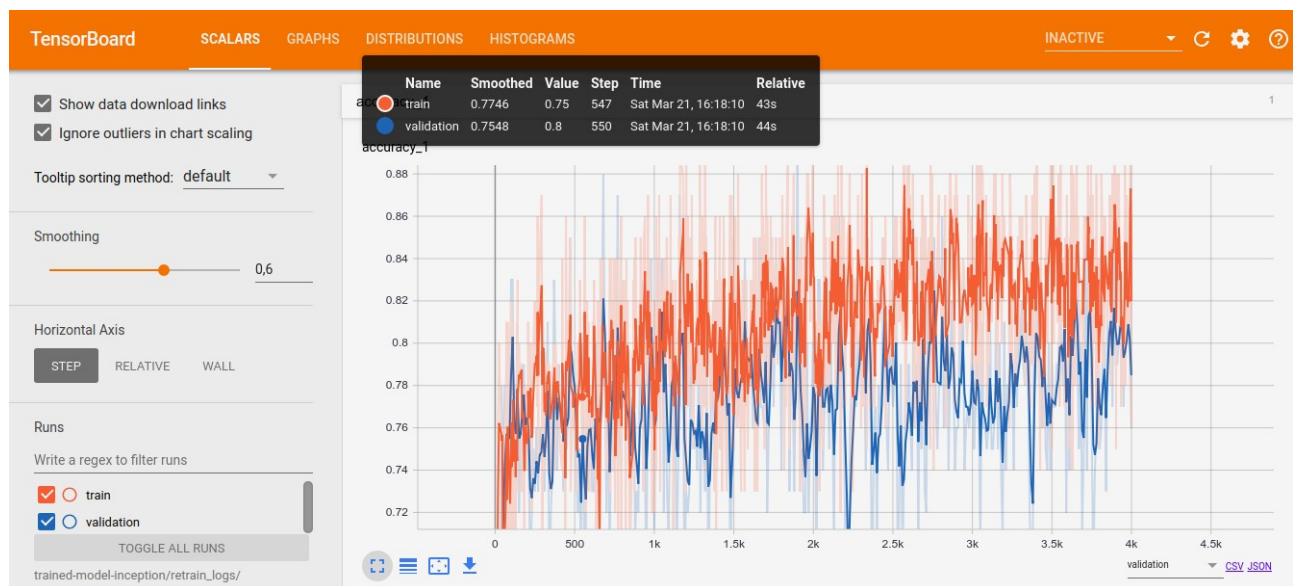


Desde esta plataforma podremos ver como TensorFlow ha llegado a generar nuestro modelo viendo las curvas de aprendizaje y validación durante el proceso de entrenamiento y test. Podremos ver también el nuevo modelo generado a partir del CNN o arquitectura neuronal de la que

partimos, pudiendo navegar de forma interactiva por el mismo de una forma muy visual.

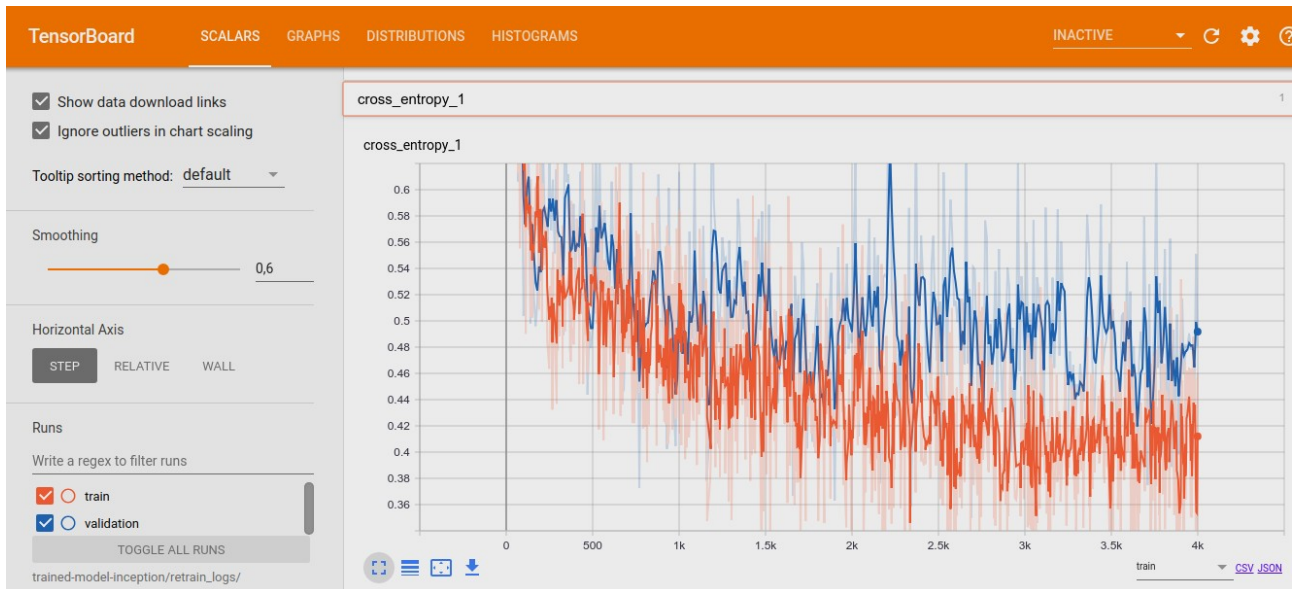
Voy a destacar las curvas de aprendizaje y validación para poder ver los niveles de precisión alcanzados. Para mayor detalle de la plataforma TensorBoard podemos consultar la documentación de la misma en [Google TensorBoard](#)

Vamos a ver las curvas generadas por TensorFlow durante el proceso de entrenamiento y validación desde TensorBoard:



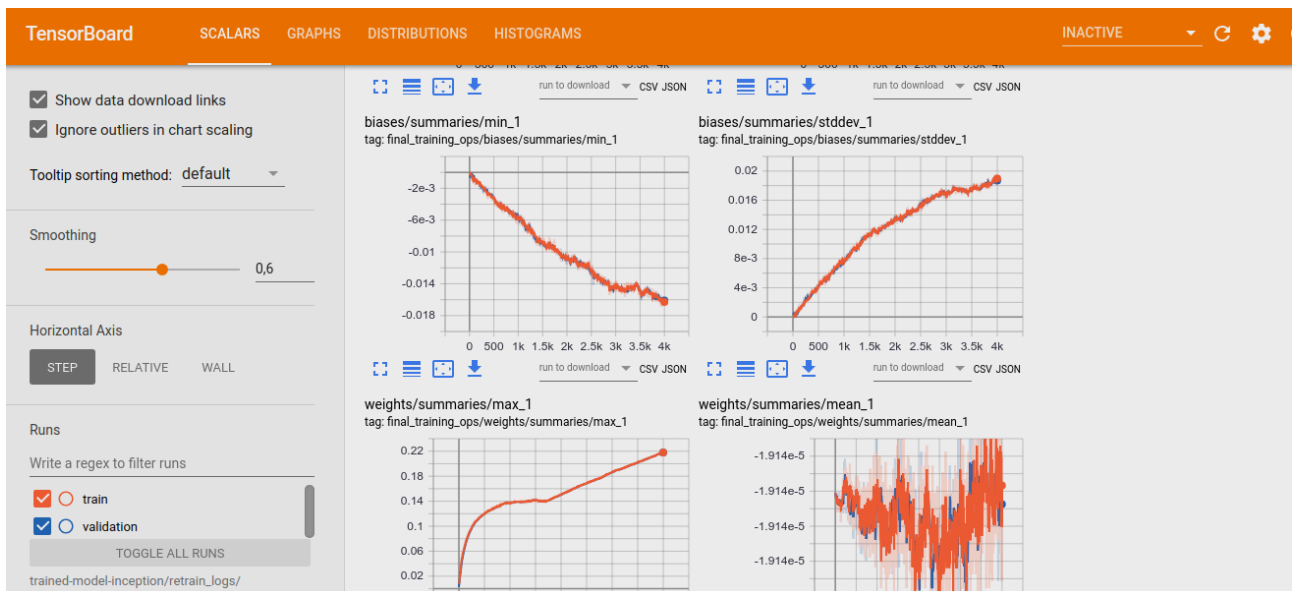
Se puede ver claramente las dos curvas de entrenamiento y validación en rojo y azul como van aumentando a medida que entrenamos el modelo. Se ve como el entrenamiento alcanza valores muy prometedores **del orden de 83% y el de validación unos 78% inferiores como es lógico**

Igualmente podemos ver las curvas de la función de pérdida utilizada en el proceso de entrenamiento y validación de TensorFlow que sigue como es lógico un proceso decreciente como podemos ver en la siguiente imagen:



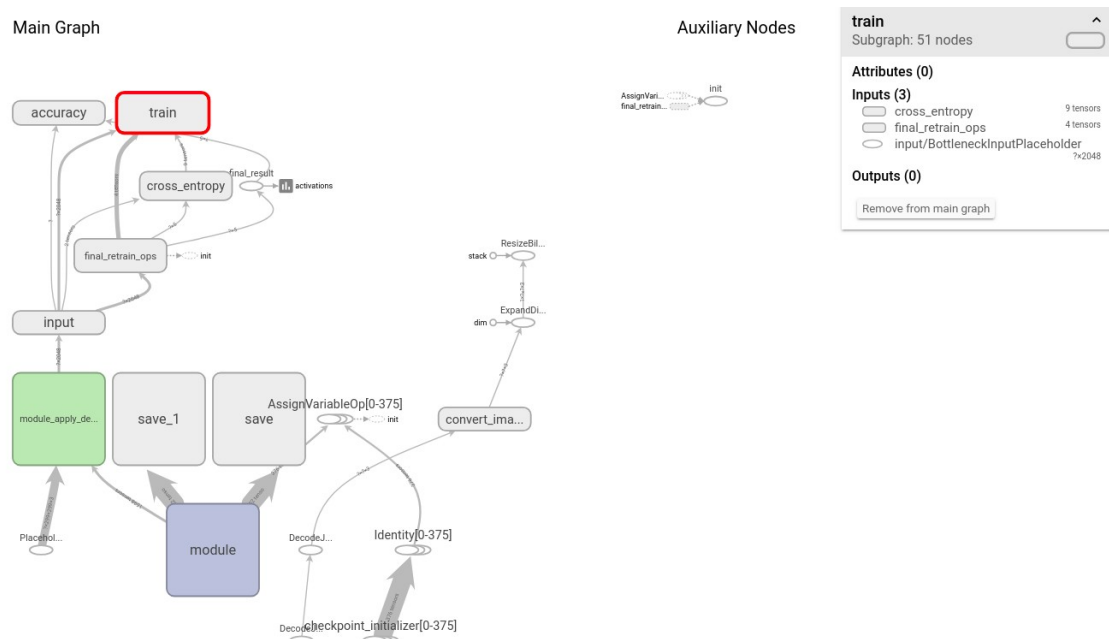
Se puede ver como ambas curvas entrenamiento y validación van disminuyendo a medida que entrenamos el modelo

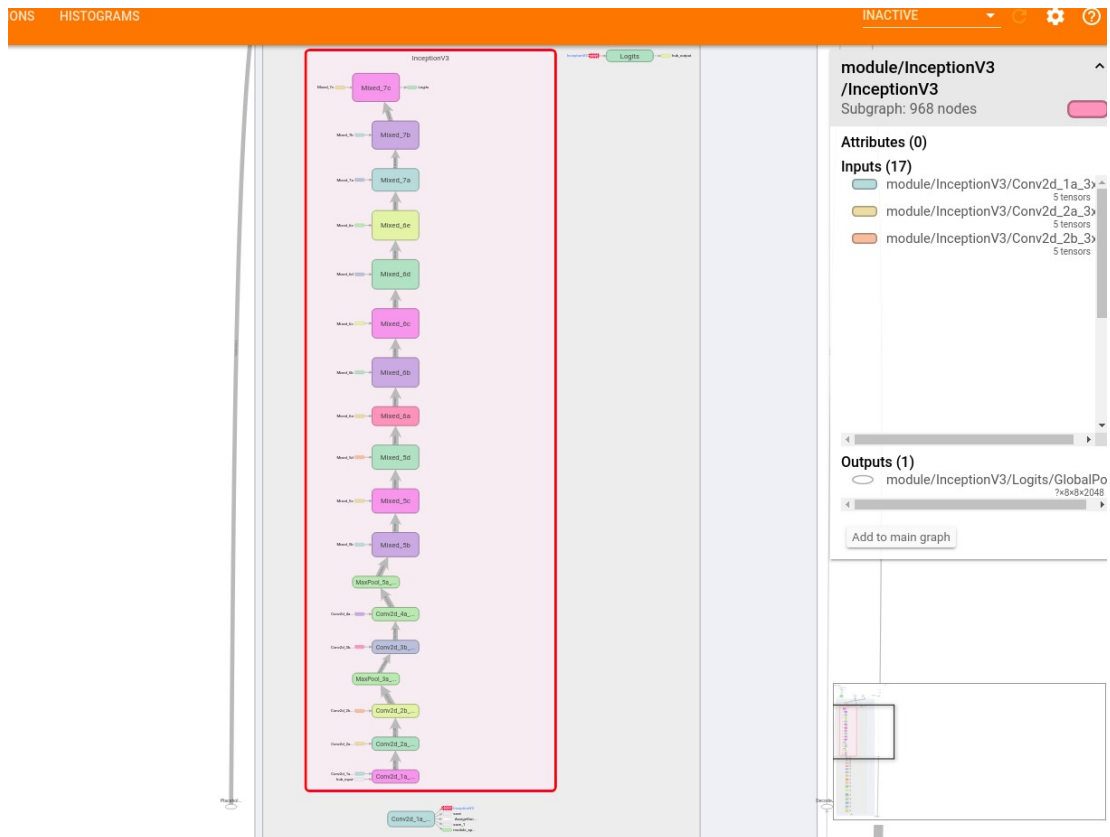
Otras curvas del proceso de entrenamiento pueden ser por ejemplo las relacionadas con la evolución de los pesos, bias como se puede ver en la imágenes inferior, todas ellas ofrecidas por los logs generados por el proceso de entrenamiento:



Visualización: Una visión gráfica del CNN

Como hemos comentado en el apartado anterior, TensorBoard nos permite visualizar con un máximo nivel de detalle el nuevo modelo generado durante el proceso de TL. En la siguiente imagen se puede ver una gráfica de nuestro modelo neuronal basado en **Inception V3** de Google. Aunque recomiendo arrancar **TensorBoard** y navegar por el modelo para poder entender en profundidad las capas de nuestro modelo CNN





Predicción: La herramienta Deep Health

Una vez escogido el modelo de Red Neuronal y siendo este ya entrenado. Deep Health permite que el acceso a este modelo sea de forma online, a través de un navegador con una simple conexión a Internet. Esta herramienta sumamente sencilla, permite que el especialista médico solo tenga que escoger la imagen Rayos X del Torax del paciente que quiere diagnosticar y dejar que la Inteligencia Artificial del modelo clasifique al paciente dentro de uno de los 3 rangos posibles de Neumonía.

Para realizar el test de Deep Health, contamos con más de 720 imágenes repartidas en 234 sin dolencias y 390 con problemas de neumología. Se ha escogido al azar una imagen para cada uno de los posibles niveles de diabetes y se han obtenido los siguientes datos:

- Paciente con **nivel sano (No_Neumonía)**:

Power by [Thingtrack](#)

The predictor rated the presence of Pneumonia in each image on a scale of 0 to 3, according to the following scale:

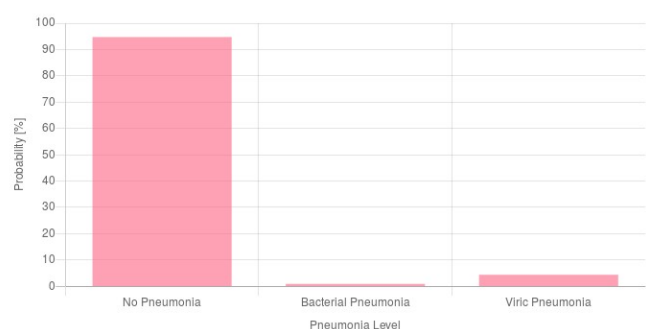
- 0 - No Pneumonia
- 1 - Bacterial Pneumonia like Streptococcus
- 2 - Viric Pneumonia like Influenza A, Covid-19

Upload image

No file selected.



Image:NORMAL2-IM-0370-0001.jpeg



• Paciente nivel Neumonía Bacteriana

Power by [Thingtrack](#)

The predictor rated the presence of Pneumonia in each image on a scale of 0 to 3, according to the following scale:

- 0 - No Pneumonia
- 1 - Bacterial Pneumonia like Streptococcus
- 2 - Viric Pneumonia like Influenza A, Covid-19

Upload image

No file selected.

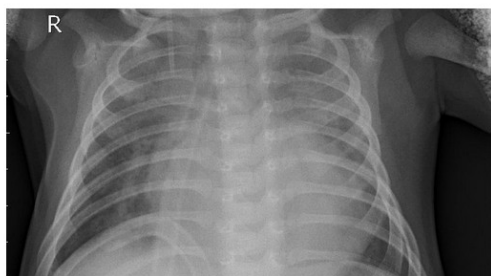
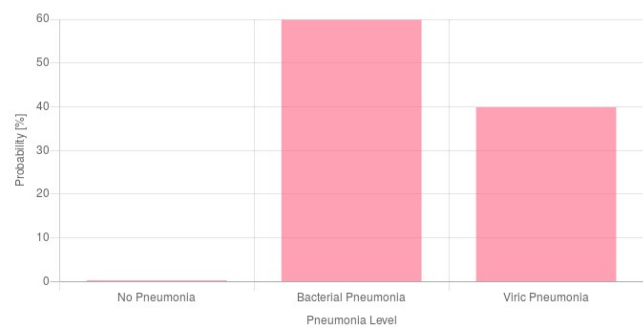


Image:person82 bacteria 404.jpeg



• Paciente con nivel Neumonía Vírica

Power by [Thingtrack](#)

The predictor rated the presence of Pneumonia in each image on a scale of 0 to 3, according to the following scale:

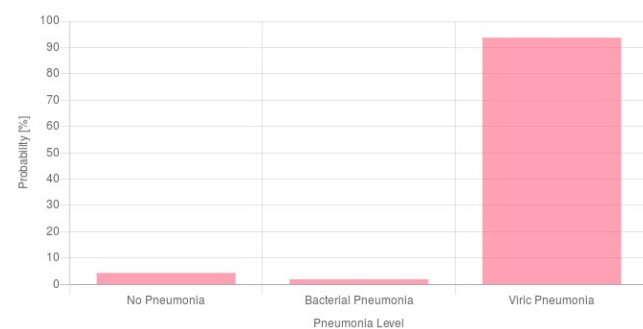
- 0 - No Pneumonia
- 1 - Bacterial Pneumonia like Streptococcus
- 2 - Viric Pneumonia like Influenza A, Covid-19

Upload image

No file selected.



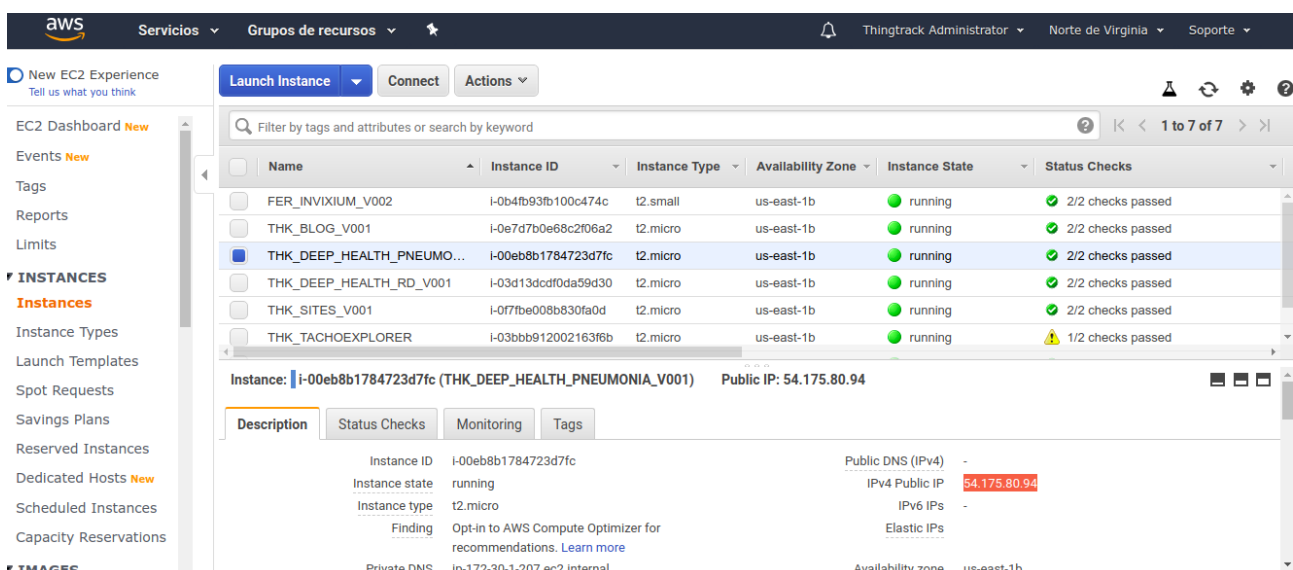
Image:person22 virus 55.jpeg



Arquitectura: Plataforma microservicios

El servicio Deep Health sigue una arquitectura de microservicios, altamente escalable orquestada por tecnología de PM2 en un sola máquina Virtual en Amazon AWS. Los recursos disponibles para esta PoC son limitados con 1vCPU y no más de 1GB de memoria y 10 GB de SSD. Aquí se puede ver la imagen VM corriendo en Amazon AWS y los servicios de backend y frontend controlados por PM2

Instancia de amazon AWS corriendo



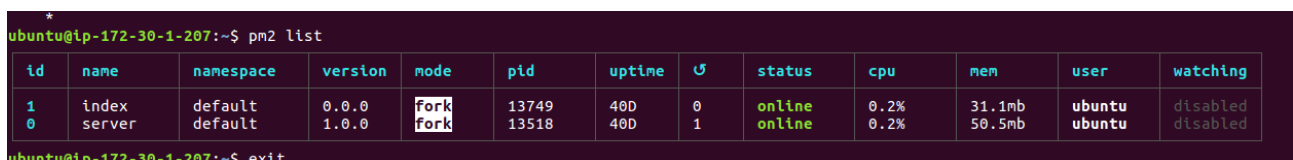
The screenshot shows the AWS Management Console interface. On the left, there's a navigation menu with options like 'New EC2 Experience', 'Events', 'Tags', 'Reports', 'Limits', 'INSTANCES', 'Images', etc. The main area displays a list of EC2 instances. The instance 'THK_DEEP_HEALTH_PNEUMONIA_V001' is selected, and its details are shown below the list. The instance is in a 'running' state, has a public IP of 54.175.80.94, and is located in the us-east-1b availability zone.

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
FER_INVIXIUM_V002	i-0b4fb93fb100c474c	t2.small	us-east-1b	running	2/2 checks passed
THK_BLOG_V001	i-0e7d7b0e68c2f06a2	t2.micro	us-east-1b	running	2/2 checks passed
THK_DEEP_HEALTH_PNEUMONIA_V001	i-00eb8b1784723d7fc	t2.micro	us-east-1b	running	2/2 checks passed
THK_DEEP_HEALTH_RD_V001	i-03d13dcdf0da59d30	t2.micro	us-east-1b	running	2/2 checks passed
THK_SITES_V001	i-077be008b830fa0d	t2.micro	us-east-1b	running	2/2 checks passed
THK_TACHOEXPLORER	i-03bbb912002163f6b	t2.micro	us-east-1b	running	1/2 checks passed

Instance: i-00eb8b1784723d7fc (THK_DEEP_HEALTH_PNEUMONIA_V001) Public IP: 54.175.80.94

Property	Value
Instance ID	i-00eb8b1784723d7fc
Instance state	running
Instance type	t2.micro
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Private DNS	in-172-30-1-207.ec2.internal
Public DNS (IPv4)	-
IPv4 Public IP	54.175.80.94
IPv6 IPs	-
Elastic IPs	-
Availability zone	us-east-1b

Servicios de Deep Helth controlados y escalados por PM2



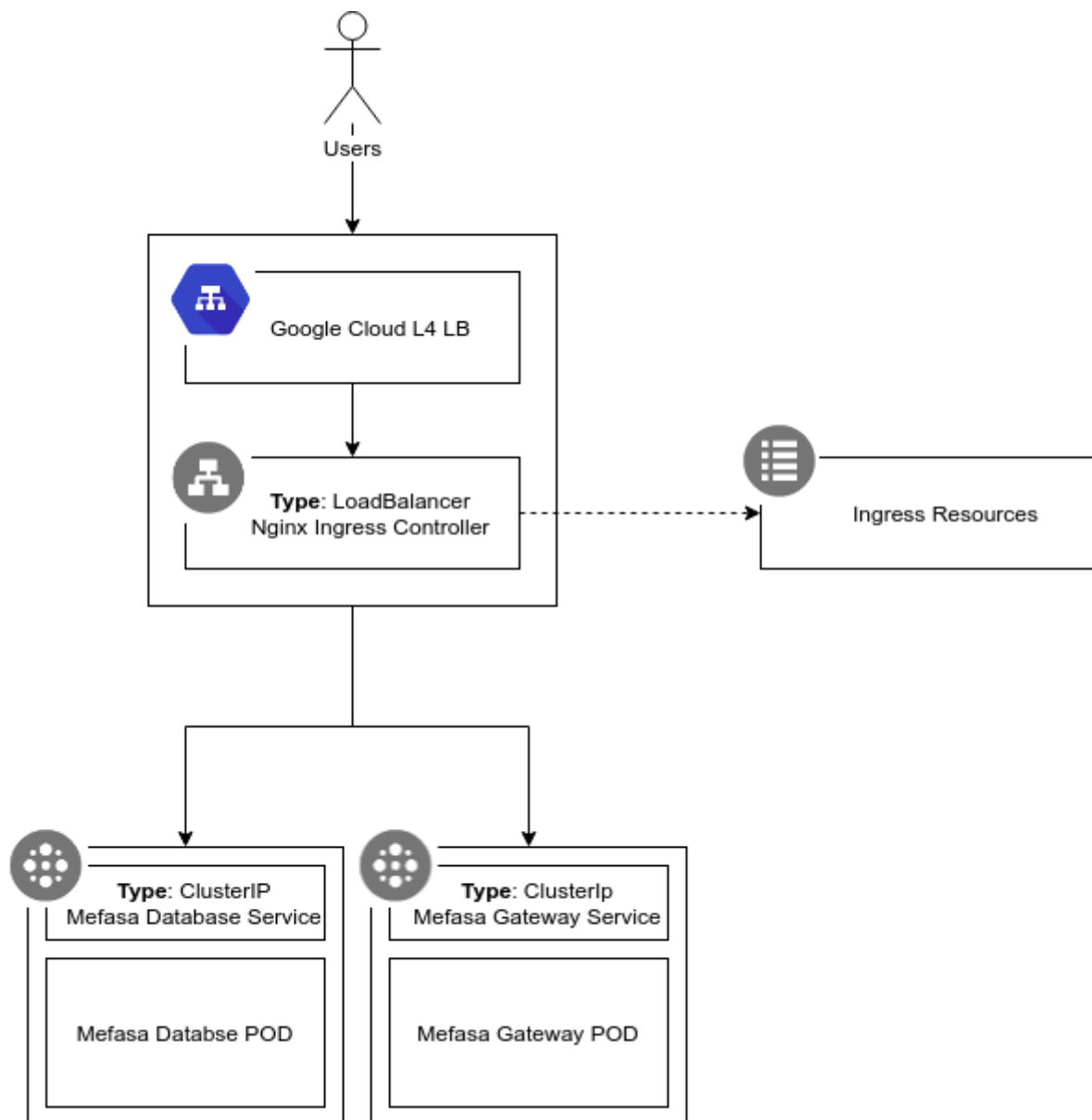
```
ubuntu@ip-172-30-1-207:~$ pm2 list
```

id	name	namespace	version	mode	pid	uptime	status	cpu	mem	user	watching
1	index server	default	0.0.0	fork	13749	400	online	0.2%	31.1mb	ubuntu	disabled
0	default	default	1.0.0	fork	13518	400	online	0.2%	50.5mb	ubuntu	disabled

```
ubuntu@ip-172-30-1-207:~$ exit
```

La arquitectura de microservicios permite que la PoC se convierta realmente en una aplicación en producción bajo el control de KGE (Kubernetes Google Engine) de Google Cloud Platform. Permitiendo escalar la aplicación sea cual sea la demanda del mercado

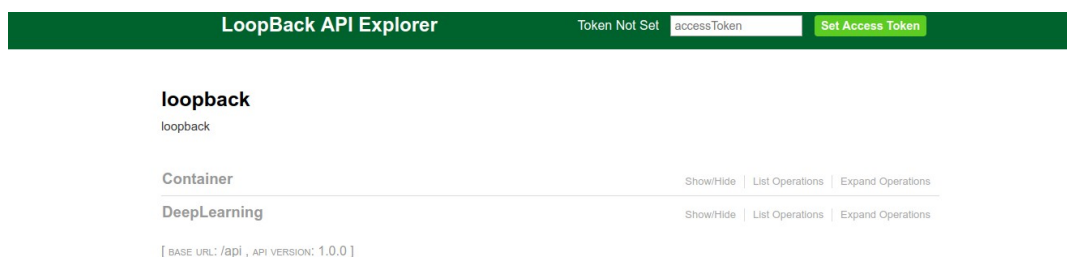
El diagrama de Kubernetes que representa el sistema escalable es:



Código Fuente: Servicios

Como se ha comentado en el apartado de la arquitectura la solución Deep Health está compuesta de varios servicios, que paso a explicar:

- **deephealth-pneumonia-model:** este proyecto representa el repositorio de imágenes utilizadas en el proceso de entrenamiento y test del modelo deeplearning de convolución.
 - **train_classified:** carpeta con las imágenes de entrenamiento y test del modelo.
 - **dataset_preprocesor.py:** Script en Python para el preprocesamiento en la siguiente fase de entrenamiento
 - **retrain.py:** script en Python para generar el modelo deeplearning utilizando la técnica de transferencia de conocimiento a partir del modelo Inception V3 Deep Concolution tercera versión de este modelo ampliamente conocido Google
- **deephealth-pneumonia-api:** este proyecto representa el servicio backend o fachas de Webservices del sistema basado en IBM Loopback, en el se encuentra embebido el modelo de deeplearning previamente creado. Una pasarela python/node nos permite realizar llamadas de predicción al modelo anterior. Se diseña un protocolo JSON para que el resultado pueda ser enviado a la capa de presentación implementada bajo el servicio **deephealth-pneumonia-ui**.



Deep Health Backend

- **deephealth-pneumonia-ui:** este proyecto representa el servicio frontend del sistema: Se utiliza angular 8 para la lógica de UI y Bootstrap 4 para el UX y representación de la información de forma sencilla al personal médico

Deep Health

Pneumonia Predictor

Author [Miguel Salinas Gancedo](#)

Power by [Thingtrack](#)

The predictor rated the presence of Pneumonia in each image on a scale of 0 to 3, according to the following scale:

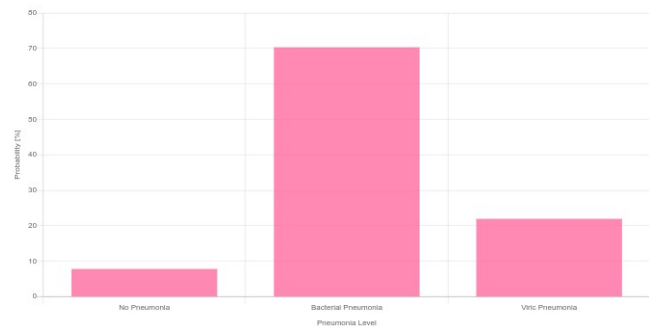
- 0 - No Pneumonia
- 1 - Bacterial Pneumonia like Streptococcus
- 2 - Viric Pneumonia like Influenza A, Covid-19

Upload image

Ningún archivo seleccionado



Image:person1_virus_13.jpeg



Deep Health Frontend

Todos los proyectos son públicos y disponibles en Github:

- **Proyecto DeepHealth Model:** thingtrack/deephealth-pneumonia-model

<https://github.com/thingtrack/deephealth-pneumonia-model.git>

- **Proyecto DeepHealth Backend:** thingtrack/deephealth-pneumonia-api

<https://github.com/thingtrack/deephealth-pneumonia-api.git>

- **Proyecto DeepHealth Frontend:** thingtrack/deephealth-pneumonia-ui

<https://github.com/thingtrack/deephealth-pneumonia-ui.git>

Resumen

Para finalizar podemos concluir como Deep Health es un predictor que puede asistir al especialista médico a la hora de diagnosticar una enfermedad como la Neumonía de forma ágil y sencilla.

Igualmente la capacidad de poder acceder a este **asistente de forma online**, permite que este diagnóstico sea instantáneo y accesible desde cualquier lugar a través de una simple conexión a Internet e incluso poder automatizar el diagnóstico. Estos sistemas inteligentes de diagnóstico abren las puertas a que la Inteligencia Artificial ayuden en otras áreas médicas donde el tiempo es un factor clave a la hora de evitar todo tipo de enfermedades.