Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

IMPLEMENTACIÓN DE UN AUTOENCODER BAYESIANO USANDO TRANSFERENCIA DE CONOCIMIENTO PARA LA CLASIFICACIÓN DE LA EPOC

Israel Rivera Zárate
Instituto Politécnico Nacional-CIDETEC
irivera@ipn.mx

Miguel Hernández Bolaños
Instituto Politécnico Nacional-CIDETEC
mbolanos@ipn.mx

Patricia Pérez Romero
Instituto Politécnico Nacional-CIDETEC
promerop@ipn.mx

Abstract

Chronic Obstructive Pulmonary Disease is a condition characterized by airflow obstruction. Smoking contributes to 95% of cases and it is associated with an abnormal inflammatory response of the lungs to harmful particles or gases such as tabaco smoke. In the present work it is shown how a variational autoencoder model based on Bayesian inference can perform the classification of COPD using chest X Ray images of the COVID19 database. It is also shown how Transfer Learning technique is used to make the encoder stage using a pretrained VGG16 convolutional net.

Palabras clave: Enfisema, bronquitis crónica, reparametrización, codificador, decodificador, autoencoder variacional, entrenamiento, sensitividad, especificidad, enfermedad pulmonar obstructiva crónica.

La enfermedad pulmonar obstructiva crónica (EPOC) es una afección caracterizada por la obstrucción al flujo del aire que no es completamente reversible, pero puede ser tratable. El tabaquismo contribuye con el 95% de los casos de EPOC ya que está asociada una respuesta inflamatoria anormal de los pulmones a partículas nocivas o gases como el humo de tabaco. En 2020 se consideró ya la tercera causa de muerte en U.S.A. y la quinta en España. De acuerdo con DL Longo *et al*. (2011), se identifican dos entidades clínicamente diferenciadas: bronquitis crónica y enfisema.

Bronquitis crónica

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

Young AL, *et al*. (2020), indica que la bronquitis crónica es un concepto clínico y se define por tos y expectoración crónica con duración de 3 meses al año reapareciendo consecutivamente durante 2 años. Anatomo-patológicamente se encuentra en los bronquios grandes hiperplasia e hipertrofia de la glándula y submucosa con un índice de Reid > 0.6, cuando el valor normal es de 0.25. El índice de Reid es la relación entre el espesor glandular y el espesor de la pared bronquial. Ver figura 1.

Figura 1. Efecto de la bronquitis crónica.
Fuente: https://topdpctors.mx.

Enfisema

El enfisema se trata de un atrapamiento aéreo distal al bronquiolo terminal con dilatación anormal y destrucción de la pared alveolar. En este proceso intervienen las enzimas elastolíticas. Clínicamente, los enfisematosos presentan disnea y complicaciones que aparecen en fases más avanzadas de la enfermedad como es la insuficiencia cardíaca derecha y la insuficiencia respiratoria global. Ver figura 2.

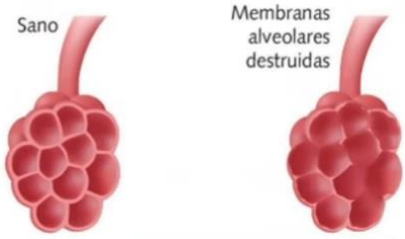
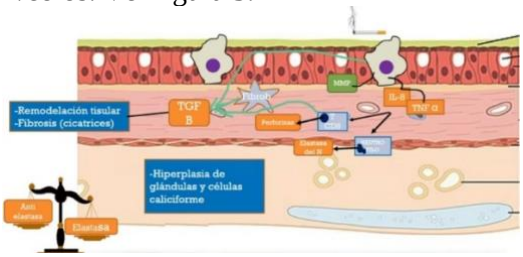


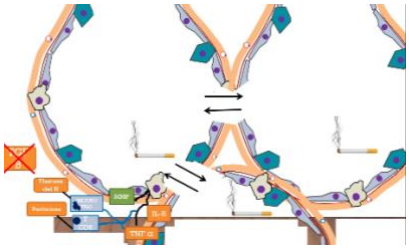
Figura 2. Efecto del Enfisema.
Fuente: https://topdpctors.mx.

Según lo expresa DL Longo *et al*. (2011), en la Bronquitis Crónica se presenta principalmente por una exposición crónica al humo del tabaco que desencadena una respuesta inflamatoria y atracción de las células inmunitarias a los espacios aéreos terminales del pulmón. Estas células liberan proteinasas elastolíticas (MMF: metaloproteinasa de la matriz) y las citocinas proinflamatorias (IL8: interleucina 8) y el factor de necrosis tumoral alfa (TNF α), las cuales atraen otras células inmunitarias por medio de quimiotaxis como el neutrófilo y los linfocitos (TCD8). El neutrófilo produce la elastasa de neutrófilo y los linfocitos la perforina. Con todo esto, se produce el desequilibrio donde habrá más elastasa que inhibidores de esta. Adicionalmente se presentan fibroblastos que producen el factor de crecimiento beta (TGF β) que va a producir cicatrices fibróticas con baja elastina. Finalmente, las células calciformes favorecen una mayor secreción de moco. En el Enfisema, también se produce una respuesta inflamatoria y atracción de células inmunitarias que van a producir muerte de la estructura celular debido al estrés oxidativo y por último una mala cicatrización de la elastina que va a desencadenar ruptura en las paredes de los alveolos. Ver figura 3.



a)

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957



b)

Figura 3. Fisiopatología del a) Bronquitis y b) Enfisema. https://topdpctors.mx.

Agusti A. *et al*. (2020), señalan que las Imágenes de radiografías de tórax han sido usadas ampliamente en la última década como una herramienta principal en la práctica clínica para la revisión cardiotorácica en la búsqueda de alguna anormalidad. La Fibrosis pulmonar, Neumonía, Enfisema, Bronquitis crónica, cáncer de pulmón son algunos de los padecimientos pulmonares que han sido detectados mediante el uso de imágenes de radiografías de tórax.

La detección de la EPOC es un problema típico de clasificación del aprendizaje máquina y el aprendizaje profundo. El uso de las redes neuronales profundas ha sido ampliamente utilizado en la clasificación e identificación de anormalidades pulmonares. Ver figura 4.

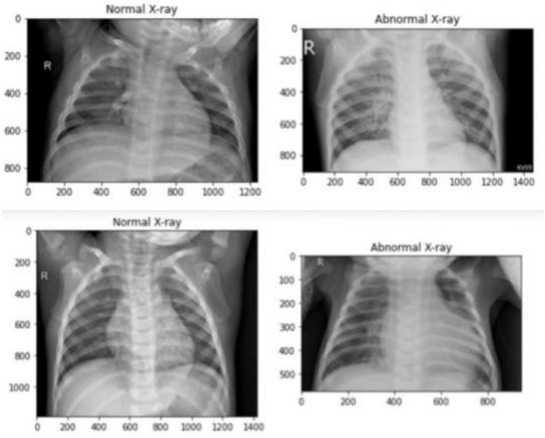


Figura 4. Imágenes de radiografía de tórax normales y anormales (neumonía o enfisema). https://msdmanuals.com/es/professional/.

I. Autoencoder Variacional

Según explica Charte D. *et al*. (2018), el VAE es una red neuronal incompleta ya que varios de sus parámetros son en realidad distribuciones gaussianas. Resulta conveniente abordar la inferencia variacional bayesiana, ya que se observa en el modelo que se cuenta con la distribución a priori de cada clase, $p(z) = \mathcal{N}(0, I)$, la red neuronal en el codificador, que actúa como el posteriori variacional $q_{\Phi}(z|x) = \mathcal{N}(\mu, I\sigma^2)$. Así como, el decodificador, que actúa como la verosimilitud $p_{\theta}(x|z) = \mathcal{N}(\hat{\mu}, I\hat{\sigma}^2)$. Ver figura 5.

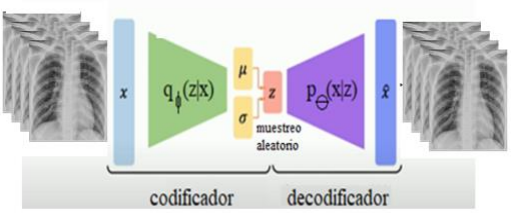


Figura 5. Autoencoder Variacional. Elaboración propia.

Dados el posteriori y la verosimilitud se puede definir la distribución conjunta como se indica en la ecuación (1).

$$q_{\Phi}(z, x) = p_{\theta}(x|z) * p(x) \ldots (1)$$

a. Regularización

Se requiere de una etapa de regularización para forzar que la distribución de aprendizaje permanezca cerca de la distribución normal. Esto se lleva a cabo mediante el cálculo de la divergencia de Kullback-Leibler (D_{KL}). Sumando esto a la función de pérdida, esta etapa actuará como una función de regularización que mide que tan lejos se encuentra la distribución aprendida respecto de la distribución normal y la acerca a la media

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

de 0 y a la desviación estándar de 1. Ver ecuación (2).

$$L(\theta, \Phi) = E_{z\sim q_\Phi(z|x)}[p_\theta(x|z)] - D_{KL}[q_\Phi(z|x)||p_\theta(z)] \ldots (2)$$

b. Entrenamiento

Al tener en el modelo una capa de muestreo estocástico que recopila datos de la distribución de entrada, no es posible obtener su gradiente. Por su parte Charte D., *et al*. (2018) han propuesto que en vez de tomar muestras directamente de la distribución se utilice más bien una expresión para generar el vector latente, ver ecuación (3).

$$z = \mu + \sigma \odot \varepsilon \ldots . (3)$$

Donde μ y σ son parámetros aprendidos durante la retropropagación y $\varepsilon \sim \mathcal{N}(0, I)$ corresponde con una distribución normal estándar que tiene una media de 0 y una desviación estándar de 1. El modelo de variable latente se apoya de la inferencia variacional para obtener la distribución a posteriori de la variable latente z. Cabe señalar que el VAE se entrena maximizando el límite inferior de la evidencia (ELBO) que en este caso corresponde con la función de pérdida establecida en la ecuación (2). El ELBO se optimiza típicamente por el gradiente respecto a θ y Φ. Donde la Divergencia KL tiene una solución analítica y K es la dimensionalidad de la variable latente:

$$D_{KL}[q_\Phi(z|x)||p(z)] = \frac{1}{2}\sum_{j=1}^{K}(\mu_j^2 + \sigma_j^2 - \log\sigma_j^2 - 1) \ldots (4)$$

II. DESARROLLO

La implementación del VAE se desarrolló en Python (https://www.python.org, versión 3.6.1) con la ayuda de las librerías gratuitas PyTorch (https://pytorch.org, versión 2.0) y Tensorflow (https://www.tensorflow.org, versión 1.2.1), ampliamente aceptadas por la comunidad. La biblioteca de programación probabilística Pyro (https://www.pyro.ai, versión 1.2.0). El entorno de desarrollo empleado ha sido el Notebook de Jupyter (http://jupyter.org, versión 5.0.0). El presente trabajo propone un sistema de detección y clasificación de la EPOC con base en un Autoencoder Variacional que recibe una imagen radiológica de tórax como entrada. El sistema cuenta con una etapa codificadora pre entrenada basada en la red VGG16 utilizada para la extracción de características. Posteriormente, con una etapa de reparametrización para el muestreo de los vectores del espacio latente. Adicionalmente, una etapa decodificadora para la reconstrucción de la imagen de entrada. Finalmente, una etapa de clasificación de la EPOC. Ver figura 6.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

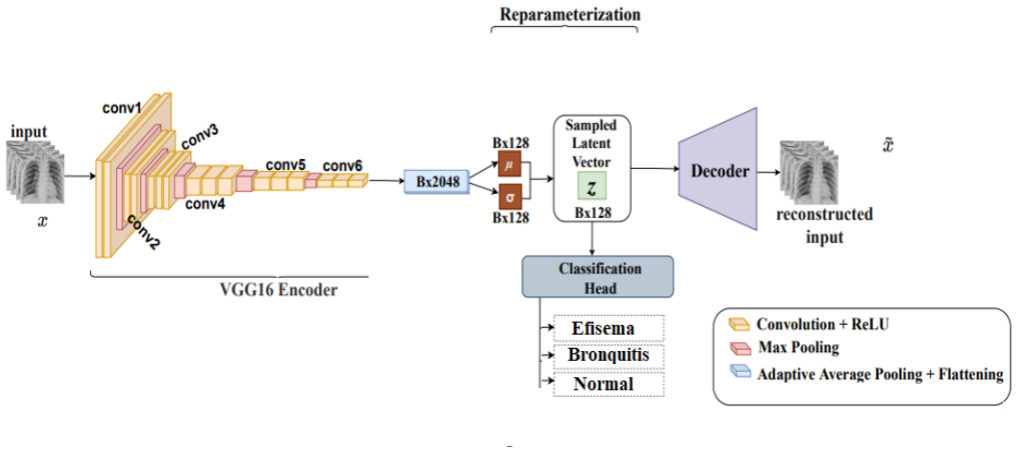


Figura 6. Sistema Autoencoder Variacional propuesto para la detección y clasificación de la EPOC. Adaptado de Addo D. et al. (2022).

Metodología

La metodología adoptada en el presente trabajo consiste en los siguientes pasos:

[Paso 1:] Creación de modelo.

[Paso 2:] Aprendizaje de la red.

[Paso 3:] Clasificación.

A. Creación del modelo

A.1 Codificador VGG16

Como lo expresan Addo D. et al. (2022), la transferencia de conocimiento o Transfer Learning hace referencia al conjunto de métodos que permiten transferir los conocimientos adquiridos gracias a la resolución de problemas previos para resolver problemas de distinta naturaleza. Los modelos utilizados exhiben grandes tiempos de cálculo y consumo elevado de recursos. Utilizando como punto de partida modelos preentrenados el Transfer Learning permite desarrollar rápidamente modelos eficaces y resolver problemas complejos. En la detección y clasificación de la EPOC se ha visto en la literatura relacionada el empleo de modelos preentrenados entre otros tales como: VGGNet, ResNet, EfficientNet, DenseNet y SqueezeNet, entre otros. En el presente trabajo se propone como etapa codificadora, el uso de la red VGG16 cuyas características se resumen en la tabla 1.

VGG16 es un modelo de red convolucional que alcanza un 92.7% de precisión habiéndose entrenado con la popular base de datos ImageNet, la cual cuenta con alrededor de 14 millones de imágenes pertenecientes a unas mil clases de objetos y a unas 22mil categorías.

Tabla 1. Red VGG16 características de sus 16 capas. Elaboración propia.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

No	Convolution	Output Dimension	Pooling	Output Dimension
layer 1&2	convolution layer of 64 channel of 3x3 kernel with padding 1, stride 1	224x224x64	Max pool stride=2, size 2x2	112x112x64
layer3&4	convolution layer of 128 channel of 3x3 kernel	112x112x128	Max pool stride=2, size 2x2	56x56x128
layer5,6,7	convolution layer of 256 channel of 3x3 kernel	56x56x256	Max pool stride=2, size 2x2	28x28x256
layer8,9,10	Convolution layer of 512 channel of 3x3 kernel	28x28x512	Max pool stride=2, size 2x2	14x14x512
layer11,12,13	Convolution layer of 512 channel of 3x3 kernel	14x14x512	Max pool stride=2, size 2x2	7x7x512

Es utilizada principalmente como extractor de características. Por lo tanto, se puede usar el código del modelo de la red VGG16 sin necesidad de volverla a entrenar. Ver figura 7.

```
import numpy as np
import matplotlib.pyplot as plt
import glob
import cv2

from keras.models import Model, Sequential
from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
from keras.layers.normalization import BatchNormalization
import os
import seaborn as sns
from keras.applications.vgg16 import VGG16
##########################
#Load model without classifier layers
VGG_model = VGG16(weights='imagenet', include_top=False, input_shape=(SIZE, SIZE, 3))

#Make loaded layers as non-trainable
for layer in VGG_model.layers:
    layer.trainable = False

VGG_model.summary()  #Trainable parameters will be 0

```

Figura 7. Configuración modelo de la red VGG16. Elaboración propia.

A.1.1 Interfaz Codificador – Espacio latente

En el lado del codificador, la capa de entrada recibe la imagen con tamaño "input_dim", se conecta hacia dos capas ocultas de tamaño "hidden_dim" (128 elementos). Las capas ocultas establecen conexiones hacia el bottleneck con tamaño "latent_dim" (2 elementos). Ver figura 8. Como salida, el codificador va a producir dos parámetros gaussianos: la media en "self.z_loc" y la desviación estándar de la variable latente en "self.z_scale".

```
class EncoderDual(torch.nn.Module):
    def __init__(self, latent_dim, input_dim=128*128, hidden_dim=128):
        super(EncoderDual, self).__init__()
        self.hidden1 = torch.nn.Linear(input_dim, hidden_dim)
        self.hidden2 = torch.nn.Linear(hidden_dim, hidden_dim)
        self.z_loc = torch.nn.Linear(hidden_dim, latent_dim)
        self.z_scale = torch.nn.Linear(hidden_dim, latent_dim)
        self.activation = torch.nn.Softplus()
```

Figura 8. Código etapa codificador. Elaboración propia.

A.2 Reparametrización

Las salidas del encoder son transformadas usando una capa lineal de tamaño del espacio latente R^{Bx128}, donde B es el tamaño del batch. Posteriormente se lleva a cabo un muestreo aleatorio a partir de una distribución normal gaussiana caracterizada por una media y una varianza. Por su parte, dos capas lineales son usadas para implementar la media y la varianza respectivamente, las cuales tienen la misma dimensión que la del espacio latente. La desviación estándar se calcula a partir de la varianza y la ecuación 5 se utiliza para muestrear z.

A.3 Decodificador

El decodificador realiza la operación inversa del codificador. Esta etapa toma a z como entrada y lleva a cabo la $ConvTranspose2d$ para producir la salida

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

$\tilde{X} \sim X$. Por su parte, el decodificador recibe como entrada la variable latente de tamaño "latent_dim" y es conducida a dos capas ocultas de tamaño "hidden_dim" (128 elementos). Ver figura 9.

```
class Decoder(torch.nn.Module):
    def __init__(self, latent_dim, output_dim= 128*128 hidden_dim=128):
        super(Decoder, self).__init__()
        self.hidden1 = torch.nn.Linear(latent_dim, hidden_dim)
        self.hidden2 = torch.nn.Linear(hidden_dim, hidden_dim)
        self.output = torch.nn.Linear(hidden_dim, output_dim)
        self.activation = torch.nn.Softplus()
```

Figura 9. Código etapa decodificador. Elaboración propia.

En la etapa de entrenamiento, el autoencoder variacional que es una clase o módulo en python se debe modificar, tanto el modelo como la guía. Para esto, se indica en el constructor la salida dual del codificador. Ver figura 10. Los modelos probabilísticos en Pyro son funciones que mapean diferentes elementos: 1. Variables latentes: z (pyro.sample). 2. Variables aleatorias observadas: x (pyro.sample) con el argumento "obs". 3. Parámetros aprendibles θ (pyro.param). 4. Manejadores de contexto: plates (pyro.plates).

```
import pyro.distributions as dists

class VariationalAutoEncoder(torch.nn.Module):

    def __init__(self, latent_dim, input_dim=128*128, hidden_dim=128):
        super(VariationalAutoEncoder, self).__init__()
        self.encoder = EncoderDual(latent_dim, input_dim=data_dim, hidden_dim=hidden_dim)
        self.decoder = Decoder(latent_dim, output_dim=data_dim, hidden_dim=hidden_dim)
        self.latent_dim = latent_dim
```

Figura 10. Salida dual codificador. Elaboración propia.

A.4 Modelo

En el lado del modelo se especifican los parámetros del decodificador: el priori y la verosimilitud. El priori p(z) se obtiene a partir de la media "z_loc" que es una distribución normal en cero y la desviación estándar "z_scale" en uno. Finalmente se toma una muestra de la variable latente "z=pyro.sample". La verosimilitud p(x|z) se obtiene pasando directamente primero por la red en "self.decoder.forward(z)" y tomando una muestra de salida "pyro.sample", Ver figura 11.

```
def model(self, x):
    pyro.module("decoder", self.decoder)
    with pyro.plate("data", size=x.shape[0]):
        # p(z)
        z_loc = torch.zeros(x.shape[0], self.latent_dim, device=x.device)
        z_scale = torch.ones(x.shape[0], self.latent_dim, device=x.device)
        z = pyro.sample("latent", dists.Normal(z_loc, z_scale).to_event(1))
        # p(x|z)
        p_logits = self.decoder.forward(z)
```

Figura 11. Programación modelo. Elaboración propia.

A.5 Guía

La guía se encarga de la inferencia aproximada, debe tener los mismos argumentos que el modelo y las mismas variables latentes. En el lado de la guía se especifica el parámetro del codificador: el posteriori. El posteriori q(z|x) se obtiene pasando directamente primero por la red en "self.encoder.forward" y tomando una muestra de salida "pyro.sample", en la que se va a obtener la media "z_loc" y la desviación estándar de la variable latente "z_scale". Ver figura 12.

```
def guide(self, x):
    pyro.module("encoder", self.encoder)
    with pyro.plate("data", size=x.shape[0]):
        # q(z|x)
        z_loc, z_scale  = self.encoder.forward(x.reshape(-1,128*128))
        pyro.sample("latent",
                    dists.Normal(z_loc, z_scale).to_event(1))
```

Figura 12. Programación guía. Elaboración propia.

A.6 Entrenamiento

Para el entrenamiento, se realiza una instancia del modelo y se aplica el descenso del gradiente con base en la función de pérdida. Se declara un objeto de inferencia variacional estocástica (SVI) y se declaran los atributos del modelo en "pyro.infer.SVI". La función de pérdida será por maximización del ELBO en "pyro.infer.TraceMean_ELBO". La optimización será Adam. Ver figura 13.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

```
pyro.enable_validation(True)
pyro.clear_param_store()

vae = VariationalAutoEncoder(latent_dim=2)

use_gpu = False
if use_gpu:
    vae = vae.cuda()

svi = pyro.infer.SVI(model=vae.model,
     guide=vae.guide,
     optim=pyro.optim.ClippedAdam({"lr": 1e-2}),
     loss=pyro.infer.TraceMeanField_ELBO(num_particles=5,
                                         vectorize_particles=True))

for nepoch in tqdm(range(50)):
    epoch_loss = 0.0
    for x, y in train_loader:
        if use_gpu:
            x = x.cuda()
        epoch_loss += svi.step(x)
    print(f"{nepoch}: {epoch_loss/(len(train_loader)*train_loader.batch_size):0.4f}")
```

Figura 13. Entrenamiento del objeto SVI. Elaboración propia.

B. Aprendizaje de la red

B.1 Base de datos

En el presente trabajo se utilizó la base datos "COVID-19 Radiography" que está disponible en el repositorio de Kaggle. Como se indica en Radiological Society of North America (2019) la base de datos está compuesta por 21,165 imágenes de rayos x de tórax, con un total de 10,192 casos normales, 6012 casos de opacidad de pulmón y 1345 casos de neumonía. Con la intención de llevar a cabo la clasificación de las imágenes en tres categorías (enfisema, bronquitis y normal). El conjunto de datos se dividió en una sección para efectos de entrenamiento del orden del 70% de la base de datos, para validación 25% y un 5% restante para prueba.

C. Clasificación

La etapa de clasificación toma la z como entrada y lleva a cabo una transformación lineal. La dimensión de salida es de R^{BxC}, donde B es el tamaño de batch, y $C = 3$ es el número de clases.

III. Pruebas y Resultados

Se propuso durante la etapa de prueba, evaluar los resultados obtenidos por el VAE propuesto en el presente trabajo respecto de los principales modelos reportados por Addo D. et al. (2022), en los que se tienen:

a) **Modelo Uno:** Se propone una arquitectura tipo autoencoder donde se concatenan los mapas de características de dos codificadores, el primero corresponde a la red ResNet50 y el segundo a la red VGG16. Esta información es luego dirigida a la etapa de reparametrización y posteriormente a la de muestreo de la variable latente z. Este modelo permite unir en alto nivel, las características individuales de ambos codificadores produciendo de este modo un mapa de características enriquecido. El vector latente muestreado es pasado luego como entrada al decodificador para reconstruir la entrada y llevar a cabo la clasificación de diferentes tipos de neumonía.

b) **Modelo Dos:** Se propone la combinación de dos VAEs de bajo nivel, unidos para conformar un solo VAE de alto nivel. El primer VAE corresponde a la red ResNet50 y el segundo a la red VGG16. Cada VAE de bajo nivel se entrena para aprender las representaciones de los datos de entrada para crear su propio vector latente. Los vectores latentes independientes cuentan con su propia distribución normal independiente y son dirigidos luego a su propia etapa de reparametrización individual. Finalmente, la información pasa a un único decodificador y clasificador. Por lo tanto, el desempeño del modelo dependerá de la sensibilidad de cada codificador para extraer características relevantes de la entrada de datos.

c) **Modelo Tres:** Se propone la combinación de dos VAEs, cada uno incluyendo sus propias etapas codificadoras, de reparametrización y decodificadoras. El

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

primer VAE corresponde a la red ResNet50 y el segundo a la red VGG16. Después de las etapas de reparametrización, el vector latente de cada codificador individual es unido para conformar un único vector latente que es luego pasado a su respectivo decodificador para reconstruir la entrada de datos y a un único clasificador.

Cabe señalar que los parámetros utilizados en los modelos de prueba fueron los siguientes: Tasa de aprendizaje de 0.00003, algoritmo optimizador Adam, número de épocas 20, tamaño de batch 4, dimensión del espacio latente 128, y dimensión de la imagen de entrada 224x224.

Métricas de Evaluación

Como lo señala Vizcaíno Salazar G. (2002) la "exactitud" (Accuracy) mide la fracción de predicciones que el modelo realizó correctamente. Es una medida indirecta de la calidad de los clasificadores.

$$exactitud = \frac{TP+TN}{TP+TN+FP+FN} \ldots (12)$$

La "sensitividad" mide la proporción de casos anormales que fueron detectados correctamente como anormales. Una sensitividad del 100% detectará a todos los pacientes enfermos:

$$sensitividad = \frac{TP}{TP+FN} \ldots (13)$$

La "especificidad" mide la proporción de casos normales que fueron detectados correctamente como normales. Una especificidad del 100% detectará a todos los pacientes sanos:

$$especificidad = \frac{TN}{TN+FP} \ldots (14)$$

Donde: Verdaderos positivos (TP): casos anormales que han sido correctamente clasificados como anormales. Falsos negativos (FN): anormales que han sido clasificados erróneamente como normales. Verdaderos negativos (TN): normales que han sido correctamente clasificados como normales. Falsos positivos (FP): normales que han sido clasificados erróneamente como anormales. En la tabla 2 se muestran los resultados obtenidos por las métricas de evaluación de los modelos reportados por Addo D. et al. (2022), durante la etapa de validación.

Tabla 2. Métricas de Evaluación de los tres modelos. Elaboración propia.

Model	Exactitud (%)	Sensitividad (%)	Especificidad (%)
Model One	96.44	95.18	96.48
Model Two	97.12	96.18	96.23
Model Three	98.24	98.24	98.32

Por su parte, la tabla 3 muestra los resultados obtenidos por la arquitectura autoencoder propuesta en el presente proyecto utilizando los datos de validación.

Tabla 3. Resultados obtenidos por el autoencoder. Elaboración propia.

Tipo prueba	Enfermedad Presente	Enfermedad Ausente (Casos Normales)	Total
Prueba positiva	900	81	981
Prueba negativa	39	1740	1779
	939	1821	2760

Por lo tanto, con base en los datos de la tabla 4 se obtienen los valores de las métricas de evaluación del modelo propuesto en el proyecto. Ver tabla 4.

Humanidades, Tecnología y Ciencia, del Instituto Politécnico Nacional
Certificado de Reserva de Derechos al Uso Exclusivo del Título: 04-2010-0326124-14000-203 ISSN: 2007-1957

Tabla 4. Métricas de Evaluación del modelo Autoencoder propuesto. Elaboración propia.

Model	Exactitud (%)	Sensitividad (%)	Especificidad (%)
Autoencoder VGG16	95.55	95.84	95.55

Con base en los resultados de la tabla 4 mostrada, se puede observar que la exactitud es de 2640/2760=95.55%. Asimismo, el valor de la sensitividad es de 900/939=95.84%, así como la especificidad es de 1740/1821=95.55%. Del mismo modo se puede apreciar que el número de falsos negativos es de 100-95.84=4.16%, así como el número de falsos positivos es de 100-95.55=4.45%.

De tal manera que se puede decir que la identificación de casos de enfermedad presente, muestra una alta sensibilidad (95.84%), por lo que ante un resultado negativo es muy probable que se descarte el diagnóstico de enfermedad pulmonar obstructiva dado que hay un número bajo de falsos negativos (4.16%) y una especificidad alta (95.55%), que demuestra que un resultado positivo indica o confirma la posibilidad de tener enfermedad pulmonar obstructiva añadiendo que hay un 4.45% de falsos positivos.

IV. Conclusiones

Se pudo comprobar que el problema de clasificación de la enfermedad pulmonar crónica por medio de imágenes de radiografías torácicas bien puede modelarse como una aproximación de probabilidad condicional bayesiana. Se observó que la arquitectura del autoencoder cuenta con la distribución a priori de cada clase $p(z)$, la red neuronal en el codificador actúa como el posteriori variacional $q_{\Phi}(z|x)$, en tanto que el decodificador actúa como la verosimilitud $p_{\theta}(x|z)$. También se pudo dar constancia de la eficacia en la etapa de entrenamiento del autoencoder, el cual se entrenó minimizando el error entre la entrada y la salida, que resultó equivalente a maximizar la verosimilitud gaussiana, añadiendo además un regularizador L2 en θ y Φ.

Por su parte, la arquitectura propuesta del tipo autoencoder variacional con un espacio latente regular de dimensión 2, basado en una media centrada en 0 y desviación estándar unitaria, con muestreo estocástico, permitió establecer una adecuada clasificación de las imágenes normalizadas de 3x224x224 de prueba obtenidos de la base de datos "COVID-19 Radiography" que está disponible en el repositorio de Kaggle.

Finalmente, con base en las mediciones efectuadas, se pudo observar que la arquitectura correspondiente al "Modelo Tres" es la que muestra mejores características de desempeño entre los modelos reportados en Addo D. et al. (2022). El modelo tres presenta una exactitud, sensitividad y especificidad por encima del 98%, en tanto que la arquitectura propuesta en el presente trabajo reportó unos valores en las métricas de evaluación por encima del 95%. Esta condición es comprensible entendiendo que los sistemas reportados cuentan con dos etapas codificadoras que realizan la extracción de características de una forma más enriquecida. Sin embargo, en la arquitectura propuesta basada en la red preentrenada VGG16 gracias a la cualidad de la Transferencia de conocimiento (Transfer Learning) y que se apoya de un VAE con inferencia bayesiana programado mediante las funciones de la biblioteca Pyro, permitió realizar representaciones eficientes de los datos de entrada, así como una adecuada reconstrucción de estos.

V. Referencias

Young AL, Bragman FJS, Rangelov B, Han MK, Galbán CJ, Galbán G, Lynch D, et al. (2020). Disease Progression Modeling in Chronic *Obstructive Pulmonary Disease*. Am J Respir Crit Care Med.; 201:294–302.

DL Longo, AS Fauci, DL Kasper, SL Hauser, JL Jameson & E. Braunwald (2011). Harrison's: *Principies of Internal Medicine*. (18a. Ed.) McGraw Hill.

Agusti A., Faner R., When Harry & Met Sally, (2020*). When machine learning met chronic obstructive pulmonary disease*. Am J Respir Crit Care Med.; 201(3): 263–75.

Charte D., Charte F., García S., Del Jesus M. J., & Herrera F. (2018). *A practical tutorial on autoencoders for nonlinear feature fusion: Taxonomy, models, software, and guidelines*. Information Fusion.

Addo D., Zhou S., Jackson J.K., Nneji G.U., Monday H.N., Sarpong K., Patamia R.A., Ekong F., Owusu-Agyei C.A. (2022). EVAE-Net: *An Ensemble Variational Autoencoder Deep Learning Network for COVID-19 Classification Based on Chest X-ray Images*. Diagnostics, 12, 2569.

Radiological Society of North America (2019). *RSNA pneumonia detection challenge*. (Recuperado el 9 de junio de 2023). https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data

Vizcaíno Salazar G. (2002). *Sensibilidad y Especificidad. Medicina basada en la evidencia y análisis de diseños de investigación clínica.* Maracaibo, Venezuela: EDILUZ; 57-71.