Trabajo Final

*Curso: Machine Learning

Piero Herrera
Ciencias de la Computación ,UPC
Lima, Perú
u201718536@upc.edu.pe

Diego Urrutia
Ciencias de la Computación ,UPC
Lima, Perú
u201515749@upc.edu.pe

Camilo Silva
Ciencias de la Computación ,UPC
Lima, Perú
u201511769@upc.edu.pe

***Abstract*—En el presente documento se describirá el procedimiento que se ha llevado a cabo para clasificar las distintas imágenes del dataset SARS-COV-2, con el fin de comparar las medidas de precisión, accuracy, f1 y recall, y concluir cual de los modelos implementados ofrece mejores resultados frente al problema propuesto. Se realizarán experimentos con cada modelo donde se variarán los hiperparametros con el fin de encontrar los ajustes pertinentes que den un resultado favorable.**

***Index Terms*—accuracy, PCA, recall, f1 score, precisión, Random Forest, Super Vector Machine, K-neighbors, K-means, SOM**

I. Introducción

El objetivo del trabajo es realizar un análisis de hiperparámetros con metricas de desempeño en 6 modelos de clasificadores distintos (3 de aprendizaje supervisado y 3 de aprendizaje no supervizado). Debemos de experimentar y comparar resultados con el fin de reconocer cual de los modelos es el que produce mejores resultados respecto a la clasifiación de imágenes tomografías computarizadas de SARS-COV-2, las cuales se dividen entre infectados (1252 img) y no infectados (1230). El conjunto de imágenes es una recopilación de resultados de pacientes en los hospitales de Sao Paulo, Brasil.

II. Marco Teórico

A. Técnicas de extracción de Keypoints

1) SIFT: Es un algoritmo que permite detectar y describir características locales en imágenes. Para ello se sigue una serie de pasos:

- Selección de picos de espacio de escala: ubicación potencial para encontrar entidades.
- Localización de puntos clave: localización precisa de los puntos clave de funciones.
- Asignación de orientación: asignación de orientación a puntos clave.
- Descriptor de puntos clave: describe los puntos clave como un vector.
- Coincidencia de puntos clave

Donde el thread es equivalente a una iteración del bucle en el caso serializado

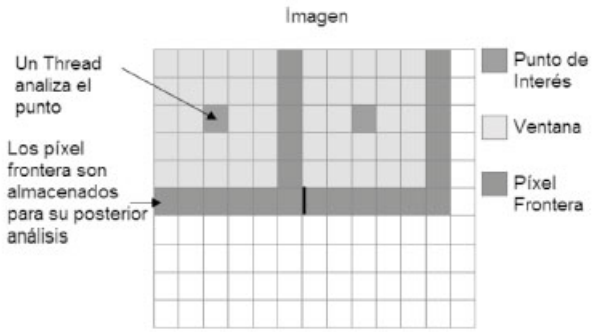


Fig. 1. Estrategia para encontrar los puntos clave

2) SURF: Es un algoritmo para la representación y la comparación local de imágenes. De manera similar a otros métetodos también basados en descriptores locales, los keyponits de una imagen dada se definen como características sobresalientes de una representación invariante de escala. Dicho análisis se obtiene mediante la convolución de la imagen inicial con granos discretos a varias escalas (filtros de caja) seguido del uso de gradientes locales (intensidad y orientación).

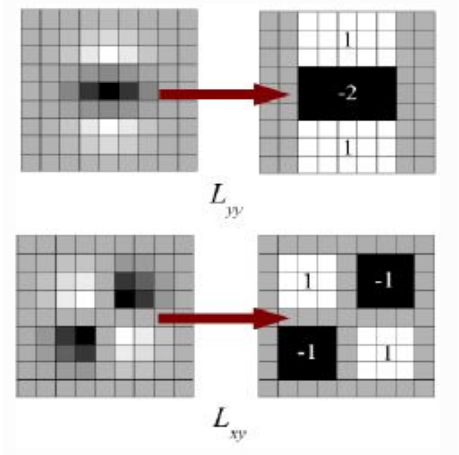


Fig. 2. Reconocimiento de pixeles SURF

SURF es bueno para manejar imágenes que presentan desenfoque y poca iluminación; sin embargo, no es tan bueno para cuando hay cambios de vista y de iluminación

3) ORB: Es un algoritmo que proviene de la fusión del detector de puntos clave FAST y el descriptor BRIEF con modificaciones para mejorar el rendimiento. Primero usa FAST para encontrar puntos clave, luego aplica la medida de esquina de Harris para encontrar los N puntos principales entre ellos. También utiliza la pirámide para producir características multiescala. Además, ya que FAST no calcula la orientación los autores propusieron que se calcule centroide ponderado por intensidad del parche con la esquina ubicada en el centro. La dirección del vector desde este punto de esquina al centroide da la orientación. Para mejorar la invariancia de rotación, los momentos se calculan con x e y que deben estar en una región circular de radio r, dónde r es el tamaño del parche.

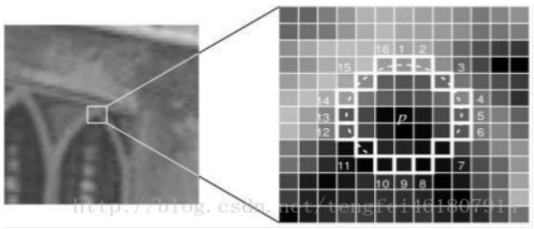


Fig. 3. Reconocimiento de pixeles alrededor de p (keypoint)

B. Bag of Words

Es una técnica que sirve para la extracción de características, ya sean de documentos o imágenes. Las características constan de puntos clave y descriptores. Los puntos clave son los puntos "sobresalientes" en una imagen, por lo que no importa que la imagen se gire, encoja o expanda, sus puntos clave siempre serán los mismos. El descriptor es la descripción del punto clave. Se hace uso de estos puntos clave y descriptores para representar cada imagen como un histograma de frecuencia de características que se encuentran en la imagen. A partir de este histograma, posteriormente, podemos encontrar otras imágenes similares o predecir la categoría de la imagen.

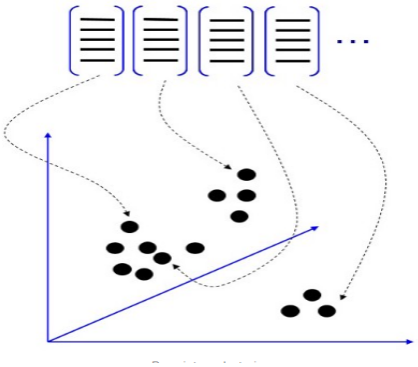


Fig. 4. Agrupación de descriptores

C. PCA

Principal Component Analysis (PCA) es un método estadístico que permite simplificar la complejidad de espacios muestrales con muchas dimensiones a la vez que conserva su información

Supongamos que existe una muestra con n individuos, cada uno con p variables (X1, X2, ..., Xp), es decir, el espacio muestral tiene p dimensiones. PCA permite encontrar un número de factores subyacentes

$$(z < p) \tag{1}$$

que llega a explicar lo mismo que las variables originales. Donde antes se necesitaban p valores para caracterizar a cada individuo, ahora bastan z valores. Cada una de estas z nuevas variables recibe el nombre de componente principal.

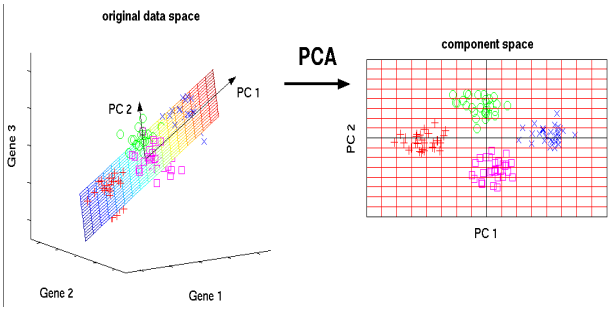


Fig. 5. Aplicación de PCA

D. Aprendizaje supervizado

En el aprendizaje supervisado, los algoritmos trabajan con datos ya etiquetados, con el fin de encontrar una función que, dados los datos de entrada (variables), les asigne la etiqueta adecuada. El algoritmo se entrena con un conjunto de datos y así aprende a asignar la etiqueta de salida adecuada a un nuevo valor.

1) Super Vector Machine: Uno de los algoritmos supervisados más poderosos que sirven tanto para clasificación como para regresión. Las máquinas de soporte vectorial ofrecen una forma de mejorar el problema del discriminativo lineal. En de simplemente trazar una línea de grosor nulo entre las clases, podríamos trazar un par de márgenes a ambos lados de la línea, trazado hasta el punto mas cercano. El modelo representa a los puntos de muestra en el espacio, separa las clases en 2 espacios lo más amplios posibles mediante un vector entre los 2 puntos más cercanos de las dos clases al que se llama vector soporte .

2) Random Forest: Es un algoritmo de aprendizaje conjunto que sirve para clasificación, regresión y otras tareas que funcionan mediante la construcción de arboles de decisión durante el entrenamiento y la salida de la clase (clasificación) o predicción media / promedio (regresión) de los árboles individuales

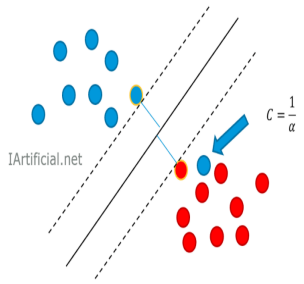


Fig. 6. Vector Soporte

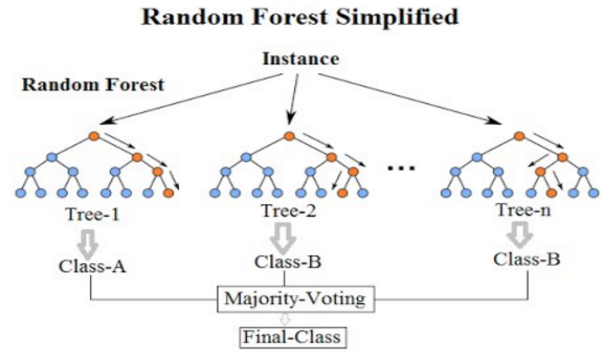


Fig. 7. Representación de random forest

3) KNeighbors Classifier: El algoritmo k-means busca un número predeterminado de agrupamientos (clusters) dentro de un conjunto de datos multidimensionales sin etiquetar. Lo logra utilizando una concepción simple de cómo se ve la agrupación óptima:

- El ”centro del grupo” es la media aritmética de todos los puntos que pertenecen al grupo.
- Cada punto está más cerca de su propio centro de agrupación que de otros centros de agrupación.

Esos dos supuestos son la base del modelo de k-means.

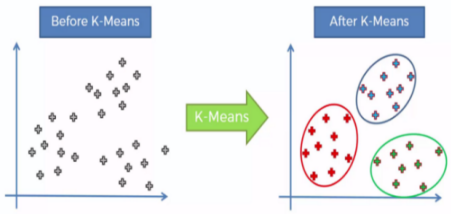


Fig. 8. Aplicación de K-Means

4) Gaussian Naive Bayes: Naive Bayes es un algoritmo de clasificación para problemas de clasificación binarios (de dos clases) y de clases múltiples. La técnica es más fácil de entender cuando se describe utilizando valores de entrada binarios o categóricos.

Se le llama Bayes ingenuo porque el cálculo de las probabilidades de cada hipótesis es simplificado para manejar comodamente el calculo. En lugar de intentar calcular los valores de cada valor de atributo p (d1, d2, d3 — h), se supone que son condicionalmente independientes dado el valor objetivo y se calculan como p(d1 — h) * p(d2 — H).

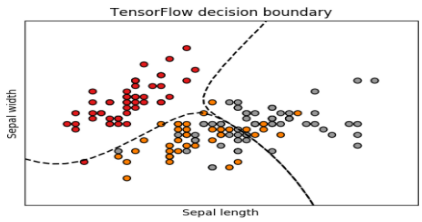


Fig. 9. Aplicación de Guassian

E. Aprendizaje no supervizado

En este tipo de aprendizaje los datos no se encuentran etiquetados para el entrenamiento. Se le considera de caracter exploratorio.Solo se conocen los datos de entrada, pero no existen datos de salida que correspondan al input. Es por ello que solo podemos describir la estrutura particular de los datos y encontrar alguna organización respecto a estas, con el fin de simplificar el estudio.

1) Cluster Jerarquico: Es un algoritmo que sirve para agrupar datos (conocidos como clusters en data mining). Agrupa los datos basándose en la distancia entre cada uno y buscando que los datos que están dentro de un clúster sean los más similares entre sí.

Los pasos del algoritmo de clustering jerárquico son:

- Hacer de cada punto un cluster
- Agrupar los dos clusters mas cercános en un nuevo cluster
- Repetir 2 hasta que sólo quede un cluster global.

Una parte importante de este algoritmo son los dendogramas. Un dendrograma es un tipo de representación gráfica o diagrama de datos en forma de árbol que se encarga de organizar los datos en subcategorías que se van dividiendo en otros hasta llegar al nivel de detalle deseado, muy similar a un árbol. Este tipo de representación permite apreciar claramente las relaciones de agrupación entre los datos e incluso entre grupos de ellos aunque no las relaciones de similitud o cercanía entre categorías. Observando las sucesivas subdivisiones podemos hacernos una idea sobre los criterios de agrupación de los mismos, la distancia entre los datos según las relaciones establecidas, etc.

2) SOM: Es un tipo particular de red neuronal artificial que se entrena mediante el aprendizaje no supervisado para producir una representación discretizada de baja dimensión (casi siempre bidimensional) del espacio de entrada de las muestras de entrenamiento, el cual es llamado mapa, y por lo tanto es un método para hacer la reducción de dimensionalidad. Difieren de otras redes debido a que aplican el aprendizaje competitivo en oposición al aprendizaje con corrección de errores (Backpropagation y gradiente descendiente). Las ventajas de este método que se se puede usar para clusterización en la misma aplicación.

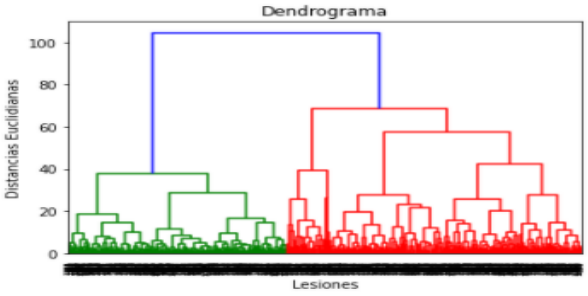


Fig. 10. Dendograma

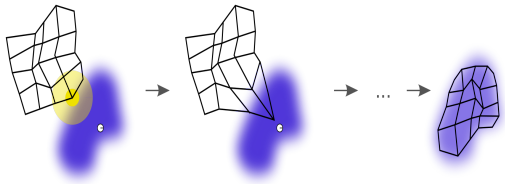


Fig. 11. Entrenamiento de un modelo SOM

III. Método

El primer paso es decargar el dataset del siguiente link https://www.kaggle.com/plameneduardo/sarscov2-ctscan-dataset. Como podemos observar se dividen en dos carpetas, la primera encontramos las imagenes de las tomografías computarizadas referentes a infectados, mientras en la otra carpeta se encuentran las de los no infectados. Se realiza un conteo de las imagenes y obtenemos la siguiente tabla:

class	img
0	1229
1	1252

Fig. 12. Tabla de imagenes

Lo siguiente es realizar un ploteo de las imagenes obtenidas:

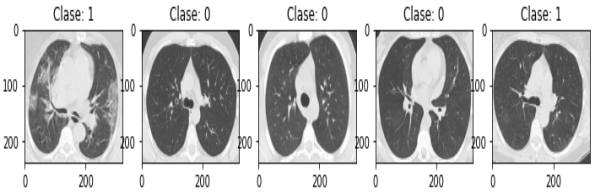


Fig. 13. Imagenes

Luego se aplican los metodos de SIFT, SURF y ORB para obtener los keypoints de cada imágen.

Estos keypoints son extraidos y pasan directamente al BOW para extraer las características de cada imagen. Estas se guardan en un dataframe donde cada registro es una imagen y cada columna representa a un valor del total de características de la imagen.

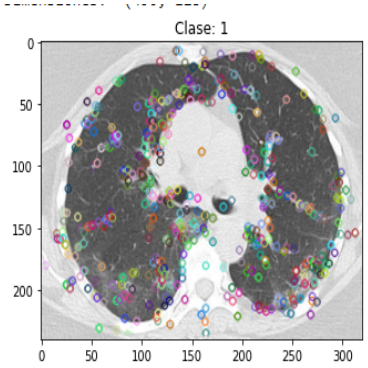


Fig. 14. Keypoints de SIFT

```
def applyBow(fFile, df):
  start_time = time.time()
  data = []
  for kp, kp_des in fFile:
    for kpd in kp_des:
      data.append(kpd)
  kmeans = MiniBatchKMeans(n_clusters=20, verbose=0).fit(data)
  kmeans.verbose = False
  features_hist = []
  for kp, kp_des in fFile:
    hist = np.zeros(20)
    for kpd in kp_des:
      feature = kmeans.predict([kpd])
      hist[feature] += 1
    features_hist.append(hist)
  print("BoW generado en %s segundos"  % (time.time() - start_time))
  for i, row in df.iterrows():
    features_hist[i] = np.append(features_hist[i], row['class'])
  cols = [ 'X'+str(i) for i in range(20) ]
  cols.append('class')
  df_bow = pd.DataFrame(data=features_hist, columns=cols)
  return df_bow
```

Fig. 15. Algoritmo BOW

	X0	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13	X14	X15	X16	X17	X18	X19	class
0	16.0	5.0	16.0	41.0	105.0	3.0	17.0	23.0	24.0	30.0	23.0	62.0	33.0	17.0	31.0	12.0	37.0	18.0	11.0	18.0	1.0
1	19.0	2.0	19.0	30.0	78.0	5.0	12.0	23.0	14.0	13.0	5.0	50.0	27.0	5.0	29.0	11.0	32.0	12.0	12.0	12.0	1.0
2	5.0	2.0	27.0	28.0	81.0	1.0	8.0	17.0	25.0	34.0	11.0	57.0	13.0	32.0	16.0	17.0	28.0	14.0	13.0	21.0	1.0
3	2.0	26.0	5.0	5.0	74.0	2.0	8.0	14.0	22.0	4.0	1.0	48.0	9.0	6.0	4.0	2.0	5.0	5.0	3.0	4.0	1.0
4	24.0	22.0	25.0	22.0	82.0	3.0	6.0	22.0	56.0	30.0	39.0	67.0	10.0	41.0	21.0	29.0	11.0	9.0	39.0	30.0	1.0

Fig. 16. DF de características

Se realiza la estandarización de de los datasets de vectores característicos reducidos y se aplica PCA.

Luego se divide la data de entrenamiento y test:

Aplicamos Kfold, para las pruebas usamos un numero de divisiones igual a 15 y activamos el parametro randomstate para que estas divisiones sean aleatorias.

$$KFold(n_splits = 15, random_state = 10, shuffle = True) \quad (2)$$

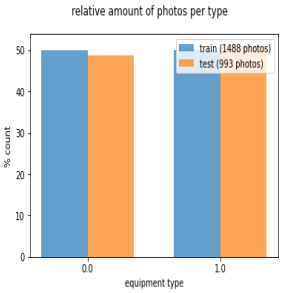
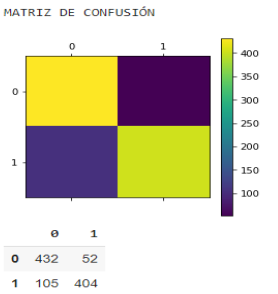


Fig. 17. División de la data

A partir de aquí se implementan los algoritmos de clasificación, tanto de aprendizaje supervisado como no supervisado. Cabe resaltar que cada uno es probado con los DF de características de SIFT, SURF y ORB.

Calculamos las métricas de accuaracy, recall, f1 y support, al igual que su matriz de confusión que en este caso es unicamente de 2x2.



	0	1
0	432	52
1	105	404

Fig. 18. Matriz de confución SURF + SVM

	precision	recall	f1-score	support
0.0	0.80	0.89	0.85	484
1.0	0.89	0.79	0.84	509
accuracy			0.84	993
macro avg	0.85	0.84	0.84	993
weighted avg	0.85	0.84	0.84	993

Accuracy Calculado: 0.8418932527693856
F1 Calculado 0.8416551133724821
Accuracy CrossValidation: 0.8581209687233783

Fig. 19. Métricas

Finalmente se hace una comparación entre las métricas que obtenemos para sacar resultados y conclusiones.

Experimentos y resultados

Obtenemos de la comparación entre diagramas de cajas de los modelos de aprendizaje supervizado en SURF lo siguiente:

El mejor modelo de todos sigue siendo el SVM

En el caso de SIFT tenemos los mismo resultados:

El mejor modelo ha sido SVM junto a Voting.

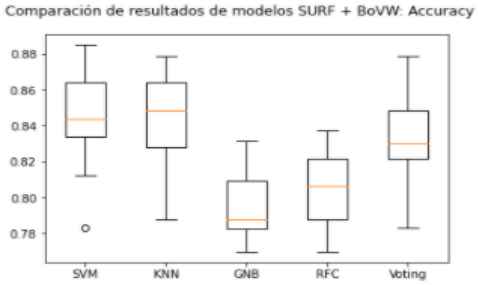


Fig. 20. SURF

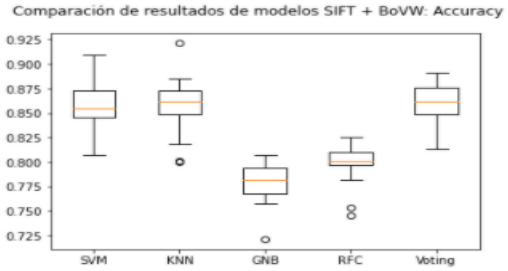


Fig. 21. SIFT

Mientras que en ORB si que obtenemos valores mucho menores comparados a los anteriores. Aún así el modelo que mejor se desempeña es el SVM

IV. Conclusiones

- Usar PCA para obtener una vector de características mucho más consiso pero sin perder data agiliza el tiempo de compilación y permite que los resultados obtenidos en las métricas sean realmente óptimos.
- Hemos visto que el resultado de las métricas, en especial f1, no distan mucho entre los diversos modelos. Esto puede deberse a que al solo clasificar dos clases, la diferencia de eficiencia no es tan notoria. Sin embargo, es posible que si se necesitaran clasificar aún más clases algunos métodos como SVM tendrían un mejor desempeño.
- Tener un data set de aproximadamente 2000 img sigue siendo pequeño, por lo que los rendimientos de ciertos modelos no son tan notorios.
- Otra de las razones por las que no hay tanta direfencia en las métricas es debido a la graduación de hiperparámetros. Se debe de hacer un análisis exahustivo de cada modelo y con cada caso de extracción de caracteríscas par obtener los mejores resultados posibles.

References

[1] Bag of Visual Words in a Nutshell, //towardsdatascience.com/bag-of-visual-words-in-a-nutshell-9ceea97ce0f.

[2] SARS-COV-2 Ct-Scan Dataset ttps://www.kaggle.com/plameneduardo/sarscov2-ctscan-datset

[3] Naive Bayes Classification in Tensor Flow https://nicolovaligi.com/naive-bayes-tensorflow.html

[4] Clustering and classification using a self-organizing MAP: The main flaw and the improvement perspectives, https://ieeexplore.ieee.org/document/7556150