Motivación

En nuestro primer acercamiento al reconocimiento de patrones en imágenes pudimos apreciar el como la combinación correcta de métodos de extracción y selección de características tiene gran incidencia en nuestra función predictora y la efectividad de nuestra solución propuesta.

Si bien fue un buen ejercicio de acercamiento, no nos quedaremos ahí. Debemos explorar nuevas técnicas y escenarios con el fin de saber construir estructuras robustas de análisis de patrones. En esta oportunidad, exploraremos el ejercicio de detección de rayados en paredes.

Por medio de una base de datos constituida de 10.000 patches a color de 64x64 pixeles, correspondientes a porciones de paredes que han sido y que no han sido rayadas, distribuidas en la misma cantidad, es decir 5.000 patches pertenecientes a la clase 1 (con rayas), y 5.000 patches pertenecientes a la clase 0 (sin rayadas), entrenaremos una función predictora que nos ayudará a reconocer a qué clase corresponde cada imagen.

Utilizaremos técnicas de extracción de características de textura, métodos de clasificación supervisada, procesamiento y filtración de caracteristicas altamente correlacionadas y análisis de variación conjunta de variables aleatorias respecto a sus medias para evaluar resultados.

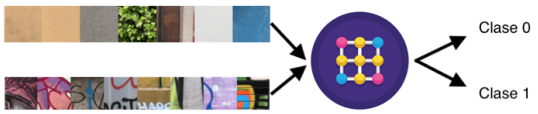

Figure 1: Visualización de muestras de clases y flujo de clasificación.

Solución propuesta

Para enfrentar este problema, hemos divido cada clase en un set de *Training*, compuesto por el 80% de la cantidad total, y el 20% restante de set de *Testing*. Para cada imagen, obtenemos su vector y extraemos sus características LBP en cada canal de colores y su escala de grises.

Originalmente se extraen **236** características. Luego, se procede a limpiarlas, en donde filtramos las altamente correlacionadas, obteniendo **212**. Luego normalizamos los valores y realizamos una selección de las **80** características que nos agregan mayor valor en la clasificación de clases usando el método SFS. Con este último set realizamos una clasificacion supervisada KNN con 3 vecinos, lo que nos entrega un valor máximo de accuracy de **96.45**.

Experimentos

Para lograr obtener el mejor set de caracteristicas a extraer, probamos multiples combinaciones de **filtros de Gabor**, **Haralich**, **HOG**, **LBP**, LBP en gamas de colores y escala de grises [1]. Nuestra hipotesis inicial es que el uso de filtros de Gabor y LBP serán los más indicados, debido a las características de las rayas de las paredes.

[1] Códigos presentados en la función extract_features ()

Una vez realizadas las pruebas de combinación de extractores de características, realizamos una limpieza para eliminar aquellas que fuesen altamente correlacionadas o que no aportasen a nuestro clasificador.

Luego, realizamos una normalización, lo cual mostró ser muy beneficioso, aumentando la efectividad del clasificador en un delta de 10%. Otro experimento que intentamos fue el uso de la técnica **Principal Component Analysis**. Si bien hemos dejado su implementación como herramienta propuesta, no la hemos aplicado, ya que mostró no solo no ser beneficiosa, si no que tener impacto negativo en nuestros resultados. El último paso consistió en utilizar el algoritmo **Sequential Feature Selector**. Este algoritmo obtuvo sustanciales mejoras en nuestro objetivo, ya que por su naturaleza, va generando las combinaciones de descriptores de la forma más conveniente posible. En este experimento, hemos elegido un SFS con número de features igual a 80. Este número fue obtenido luego de múltiples simulaciones, cuyos resultados hemos abreviado en la siguiente Tabla.

# Características	100	85	81	80	79	76	75	70	60	50	30	20
Accuracy	0,956	0,961	0,960	**0,964**	0,960	0,958	0,958	0,957	0,955	0,957	0,940	0,935

Figure 2: Tabla accuracy vs número de características SFS.

Por último, se implemento un algoritmo del tipo K-Nearest Neighbors, el cual computa distancias entre representaciones de datos para realizar una predicción.

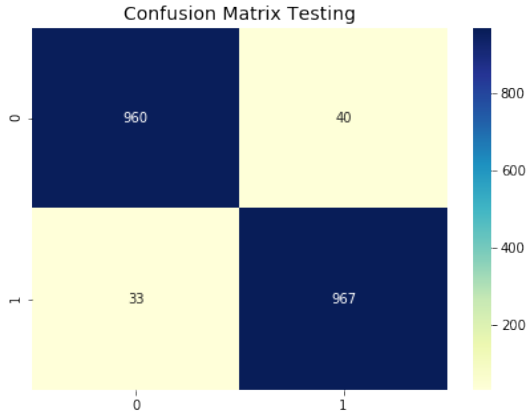

Figure 3: Matriz de confusión.

Conclusión

Luego de haber generado múltiples simulaciones con todas las combinaciones de técnicas posibles, hemos logrado una mejora importante a los resultados de la hipotesis inicial. Algunas enseñanzas claves que nos deja este ejercicio es el como algunas herramientas pueden presentar grandes beneficios en algunos casos, pero tener incidencia negativa en otros, como fue el caso de PCA.

Otra enseñanza es la importancia de generar experimentos de forma iterativa sobre los hiperparametros de cada técnica, lo que nos permitió, por ejemplo, pasar de usar 20 features en SFS a usar 80, lo cuál aumento de 0.9345 a 0.965, máximizando nuestros resultados.