UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

COMPARACIÓN DE DESCRIPTORES PARA CLASIFICACIÓN DE MEMES

MEMORIA PARA OPTAR AL TÍTULO DE
INGENIERO CIVIL EN COMPUTACIÓN

ALBERTO IGNACIO SARA ZAROR

PROFESOR GUÍA:
BENJAMÍN BUSTOS CÁRDENAS

MIEMBROS DE LA COMISIÓN:
MAURICIO CERDA VILLABLANCA
BÁRBARA POBLETE LABRA

SANTIAGO DE CHILE
2020

Resumen

El acelerado crecimiento de las redes sociales ha traído muchos nuevos fenómenos que no han sido estudiados completamente. Uno de estos es conocido como meme, que son imágenes cómicas y con relevancia cultural, creadas por usuarios de redes sociales. Son populares por su viralidad y tienen impacto en la sociedad, especialmente por el alto número que se genera. Sin embargo, por la cantidad elevada de información multimedia transmitida a través de redes sociales, es difícil poder identificarlos dentro del gran volumen de imágenes que se publica a diario, lo que dificulta su estudio.

Por esto, se plantea la necesidad de contar con una forma automática para poder clasificar imágenes como meme o no-meme, algo que no ha sido estudiado con anterioridad en la literatura. En esta memoria se proponen diversos métodos que hacen uso de descriptores clásicos y del estado del arte para extraer vectores de características a partir de las imágenes, y se utilizan distintos algoritmos de clasificación para resolver el problema. Además, se propone la búsqueda de imágenes difíciles, es decir, imágenes que corresponden tanto a memes como no-memes, que no puedan ser clasificadas correctamente por ninguno de los métodos aplicados.

Con la realización de los experimentos, se determinó que para la base de datos utilizada, tanto el descriptor profundo del estado del arte DeCAF7 con una máquina de vectores de soporte, como la red neuronal convolucional ResNet152, obtuvieron un 75 % de precisión en el problema de clasificación, por lo que el problema puede ser resuelto pero aún existe espacio para mejorar. También, se pudo determinar experimentalmente que alrededor del 1 % de las imágenes estudiadas no se pueden clasificar con ningún método propuesto, por lo que se establece la aproximación a una cota del 99 % de precisión que podría ser alcanzada si se tuviese un "oráculo" para elegir el método de clasificación apropiado para cada imagen.

Gracias por tanto,
perdón por tan poco.

Saludos

Agradecimientos

Quiero partir agradeciendo al Instituto Milenio Fundamentos de los Datos por el dataset utilizado en esta memoria de título. Junto a esto, quiero agradecerle a Jesús Pérez-Martín por sus consejos y su buena disposición para ayudarme.

Quiero agradecer a mi profesor guía, Benjamín Bustos, por su constante apoyo durante todo el trabajo de título, haciéndola una experiencia de aprendizaje rica y llevadera.

No puedo dejar de lado a mis amigos más cercanos que me dieron tanto ánimo y apoyo desde que les comenté la propuesta inicial de la memoria hasta que vio la luz del día compartiendo sus memes. En especial, quiero agradecerle al Levil, por sus memes que quedaron en este documento, al Oliva, por ser el creador de uno de los memes, y a Carlete y Jajá, por autorizarme a usar memes hechos con sus fotos.

Finalmente quiero agradecer a mi familia, y en particular a mi madre y a mi padre, por todo el sacrificio y la ayuda que me han dado no solo en el transcurso de toda la carrera, sino también de toda la vida.

Tabla de Contenido

Capítulo 1

Introducción

Con el explosivo crecimiento que ha tenido el uso del internet en las últimas décadas, han surgido muchos fenómenos culturales nuevos. La generación y transmisión de contenido multimedia aumenta de manera muy acelerada, pero también es muy impredecible. Hay una evolución constante en como se propaga todo tipo de información, tanto en la forma como en los medios. Estos ya no son limitados a un grupo pequeño de personas, sino más bien están al alcance de todas las personas [20].

(a) Este meme muestra los distintos niveles de felicidad que siente un chileno al ser llamado de forma afectiva o personal por gente típica de Chile.

(b) Este meme muestra el descontento que existe en relación al acontecer político nacional.

Figura 1.1: Memes creados por usuarios de distintas redes sociales.

En particular, una forma de propagar información que se ha hecho muy común en los últimos años son los memes [20]. Estos consisten de imágenes con textos superpuestos, con finalidad humorística y referencias a cultura popular, pero que rápidamente comienzan a cubrir un espectro muy grande de temas. Ejemplos de memes se muestran en la Figura 1.1.

Debido a esto, hemos podido presenciar como empiezan a tener un impacto real en la sociedad, pero no siempre positivo; de manera malintencionada, se puede explotar su viralidad para poder esparcir desinformación y discursos de odio [3] [23].

Dado el gran volumen de memes que se genera en redes sociales, son un perfecto candidato para poder ser estudiado en profundidad y poder determinar la veracidad de la información que despliegan o analizar su contenido en general. Sin embargo, el problema es que en redes sociales se genera una cantidad de datos multimedia muy grande y no existen métodos fáciles que permitan distinguirlos de los memes de manera automática. La literatura existente plantea métodos que permiten realizar categorizar distintos memes según su contenido visual o comunicacional, pero no existen estudios anteriores que permitan diferenciar imágenes meme de no-meme. Por esto, el propósito de esta memoria es poder determinar, en primera instancia, que tan factible es poder clasificar imágenes según sean memes o no, y en segunda instancia, dar una primera aproximación a métodos que permitan realizar esta clasificación.

Para esto, se plantea el uso de descriptores visuales [8]. Estos son distintos tipos de características que pueden ser extraídas de imágenes, y nos permiten obtener una representación más simple de ellas. Estos pueden ser utilizados en conjunto con distintos tipos de clasificadores para poder lograr este objetivo. Además, existen métodos para clasificar imágenes del estado del arte como son las redes neuronales [18]. Aunque estas son más costosas de implementar, cada vez existen más recursos y técnicas para poder utilizarlas con mayor facilidad. En particular, técnicas como *transfer learning* [26] y *fine tuning* [28] permiten utilizar recursos como redes preentrenadas en datasets grandes y de dominio general con alta efectividad.

El objetivo principal de esta memoria consiste en evaluar el desempeño de distintos descriptores visuales y clasificadores para poder determinar que combinaciones logran una mejor caracterización y permiten resolver el problema de clasificación.

1.1. Metodología y Resultados Principales

Para llevar a cabo los objetivos señalados, de identificar memes, sus descriptores y estrategias de clasificación, se plantean tres experimentos principales. El primero es utilizar descriptores visuales y clasificadores estadísticos para realizar la clasificación y ver que combinaciones tienen el mejor desempeño. El segundo consiste en utilizar una red neuronal del estado del arte para realizar la clasificación. El tercero busca encontrar una aproximación a una cota superior de las imágenes que podamos clasificar correctamente. Esto se realiza buscando imágenes difíciles, o sea, imágenes que no puedan ser clasificadas correctamente por ninguno de los métodos probados en los dos experimentos anteriores. La idea de esto es que, si tuviéramos una forma de elegir el mejor método para cada imagen al momento de clasificar, cual es la precisión máxima que podríamos alcanzar.

Para evaluar los resultados, se utilizan una serie de métricas que incluyen la precisión, exactitud, recall, F1 y matrices de confusión. Así se puede tener una evaluación más completa para poder realizar comparaciones más significativas entre los distintos métodos.

Con todos los experimentos realizados, se logra obtener resultados interesantes. Los métodos basados en redes neuronales alcanzan la mayor precisión al clasificar, llegando hasta el 75 % al utilizar la red DeCAF7 con una máquina de vectores de soporte, o al realizar el proceso completo de clasificación con ResNet152. Los descriptores visuales clásicos se desempeñan peor, siendo el mejor la combinación de histograma de grises junto con el clasificador random forest, alcanzando un 70 % de precisión. En el tercer experimento, se obtiene que alrededor del 1 % de las imágenes no puede ser clasificada por ninguna combinación, lo que implica que, en teoría, tenemos una cota superior de alrededor del 99 % de precisión.

Con los resultados obtenidos en esta memoria, se deja un precedente de que es posible realizar clasificación de memes con métodos del estado del arte, y más aún, se plantea una aproximación a una cota superior para la precisión máxima que se podría alcanzar. Queda planteado como trabajo futuro poder encontrar un método que permita alcanzar dicha cota, por ejemplo buscando una heurística que nos ayude a elegir el método apropiado para clasificar cada imagen, o el uso de un grupo de clasificadores que predigan la clase a través de votación.

Este estudio se enmarca dentro de un proyecto emblemático sobre Generación de Estructuras de Información Robusta del Instituto Milenio Fundamentos de los Datos (IMFD), el cual busca estudiar problemas que son influenciados por la falta de información robusta, es decir, conocimiento veraz que puede ser extraído de fuentes no estructuradas.

Capítulo 2

Conceptos Básicos

En este capítulo se cubren los conceptos básicos necesarios para entender el trabajo desarrollado en la memoria. Esto incluye definiciones formales de procesamiento y análisis de imágenes, clasificadores estadísticos e inteligencia computacional.

2.1. Descriptores Visuales

Los descriptores visuales son descripciones de las características visuales de los contenidos de imágenes o vídeos. Dichas características son elementales, como la forma, color, textura u otros.

Los descriptores visuales pueden ser de dominio general o de dominio específico. En el primer caso, proporcionan información acerca de la imagen en su totalidad, siendo de color, formas, regiones, texturas y movimiento entre otros. En el segundo caso, se entrega información acerca de objetos o eventos específicos dentro de la imagen, como por ejemplo en el caso de reconocimiento facial.

Para nuestro objeto de estudio, nos centramos principalmente en descriptores de dominio general. En particular, cubriremos cuatro descriptores visuales:

- Distribución de Color: Utilizando la transformada discreta del coseno en imágenes en el espacio de colores YCbCr, se puede obtener un vector de características para clasificar [27].
- Histograma de Gradientes Orientados: Calculando la magnitud y orientación de los gradientes de una imagen, se puede generar un vector de características utilizando la concatenación de histogramas de su orientación [5].
- Histograma de Bordes: Extrayendo los distintos tipos de bordes, se calculan histogramas basados en los bordes dominantes dentro de distintas sub-regiones de una imagen, que se utiliza para crear un vector de características [17].
- Histograma de Grises: Transformando una imagen a escala de grises, se pueden generar histogramas con la intensidad de cada punto para distintas celdas dentro de esta. La concatenación de estos histogramas genera un vector de características.

2.1.1. Distribución del Color

Sea I una imagen en espacio de color RGB de tamaño $N \times M \times 3$. Se denomina a I la transformación de I al espacio de color YCbCr. Se divide cada canal de I' con una grilla de 8×8, como se aprecia en la Figura 2.1, y se refiere a cada celda como I'_{ijc}, en donde los subíndices i y j representan la fila y columna de la grilla y c el canal.

Sea C una matriz cuyos componentes se definen como

$$C_{jk}^{(N)} = \sqrt{\alpha_j / N}\ cos\left(\frac{\pi\,(2k\ +\ 1)\,j}{2N}\right),$$

se define la transformada discreta del coseno (TDC) de la siguiente forma

$$TDC(I)\ =\ C^{(N)} \cdot I \cdot (C^{(N)})^T,$$

y se define una matriz T cuyos componentes de definen como

$$T_{ijc} =\ prom(TDC(I'_{ijc})),$$

donde prom se refiere al promedio de todos los elementos de la matriz $TDC(I'_{ijc})$. Podemos definir el descriptor de Distribución del Color D como

$$D\ =\ (T_{000}T_{010}...T_{080}T_{180}...\ T_{880}T_{001}...T_{883}).$$

D es un vector de dimensionalidad 192 con la concatenación de la transformada discreta del coseno del valor promedio de todas las celdas producidas por la grilla en I'.

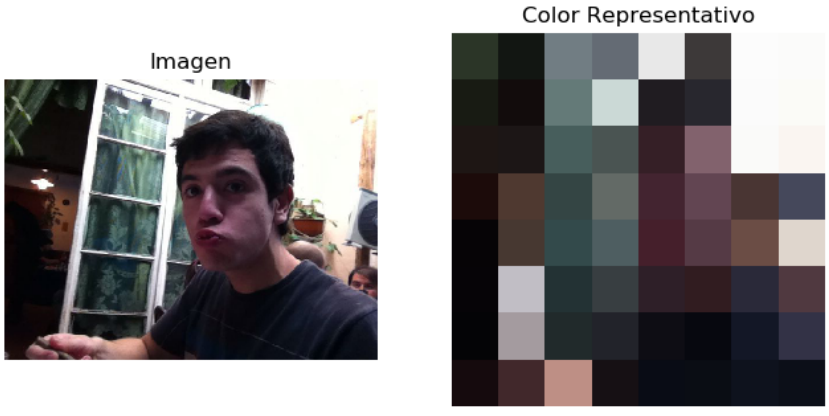


Figura 2.1: Colores dominantes de una imagen antes de aplicarle la transformada discreta del coseno y calcular su histograma.

2.1.2. Histograma de Gradientes Orientados

Sea I una imagen en espacio de color RGB de tamaño $N \times M \times 3$. Se denomina a I' la transformación de la imagen a escala de grises de dimensiones $N \times M$.

Definimos las matrices K_x y K_y como:

$$K_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, \; K_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}.$$

Se definen las matrices I'_x, I'_y como

$$I'_x = K_x * I', I'_y = K_y * I',$$

donde $*$ es la operación convolución. Estas matrices representan las imágenes en donde en cada punto se ha calculado aproximaciones de la derivada parcial horizontal y vertical, respectivamente. Se define la matriz I'_M como

$$I'_M = \sqrt{I'^2_x + I'^2_y},$$

que representa la magnitud de los gradientes en cada punto, y se define también la matriz I'_θ como

$$I'_\theta = atan\left(\frac{I'_y}{I'_x}\right),$$

que representa la dirección del gradiente en cada punto, como se muestra en la Figura 2.2. Se define también un umbral p, de forma tal que solamente se consideran los gradientes cuya magnitud sea mayor a este.

Se define una grilla dimensiones $U \times V$ sobre I', y se refiere a cada celda como I'_{ij}, en donde los subíndices i y j representan la fila y columna de la grilla.

Sobre cada celda I'_ij se calcula un histograma H_{ij} con las orientaciones de los gradientes, con un número B de bins. Las orientaciones pueden estar entre 0° y 180° o 0° y 360°, en caso de que se considere su signo o no. En el caso de que se considere, tenemos 360 direcciones distintas, pero en el caso que no se considere el signo solamente se tienen 180 direcciones distintas pues las con signos opuestos se consideran la misma. El número de bins suele ser 9 si no se considera el signo, o 18 si se considera. En los histogramas, los votos tienen un peso en función de la magnitud del gradiente.

Para tomar en cuenta cambios producidos por iluminación y contraste dentro de la imagen, se define un subconjunto de celdas espacialmente conectadas llamadas bloques, y cada bloque es normalizado. Se puede definir el descriptor de Histogramas Orientados D como la concatenación de estos histogramas:

$$D = (H_{00}H_{10}...H_{U0}H_{01}...H_{0V}...H_{UV}).$$

D es un vector de dimensionalidad $U \cdot V \cdot B$ con la concatenación de los histogramas de gradientes orientados.

Figura 2.2: Representación gráfica del histograma de bordes orientados.

2.1.3. Histograma de Bordes

Sea I una imagen en espacio de color RGB de tamaño $N \times M \times 3$. Se denomina a I' la transformación de la imagen a escala de grises de dimensiones $N \times M$.

Se define además los siguientes kernels

$$K_1 = \begin{pmatrix} 1 & 1 \\ -1 & -1 \end{pmatrix}, K_2 = \begin{pmatrix} 1 & -1 \\ 1 & -1 \end{pmatrix}, K_3 = \begin{pmatrix} \sqrt{2} & 0 \\ 0 & -\sqrt{2} \end{pmatrix}, K_4 = \begin{pmatrix} 0 & \sqrt{2} \\ -\sqrt{2} & 0 \end{pmatrix}, K_5 = \begin{pmatrix} 2 & -2 \\ -2 & 2 \end{pmatrix}.$$

Para cada kernel k se filtra I' realizando una convolución para extraer los bordes. Se puede definir conjunto ${I'}_K$ como

$${I'}_K = \{ i_n \in {I'}_K \mid i_n = I' * K_n \},$$

que representa todos los bordes extraídos de nuestra imagen, donde $*$ es la operación convolución. La Figura 2.3 muestra un ejemplo de los bordes obtenidos.

A partir de ${I'}_K$, se puede definir una matriz de dimensión $N \times M$ que codifique los bordes dominantes:

$$B_D(x,y) = j,\ j = \ max(|i_1(x,y)|, ..., |i_j(x,y)|, ..., |i_5(x,y)|).$$

Es decir, el valor de B_D para el punto ubicado en (x,y) es el índice del kernel que maximiza el valor absoluto de la convolución con I' para dicho punto.

Se divide B_D con una grilla de dimensiones $U \times V$, y se refiere a cada celda como B_{Duv}, en donde los subíndices u y v representan la fila y columna de la grilla. Para cada celda B_{Duv} se calcula un histograma H_{uv}. Se puede definir el descriptor de bordes D como la concatenación de estos histogramas:

$$D \ = \ (H_{00}H_{10}...H_{U0}H_{01}...H_{0V}...H_{UV}).$$

D es un vector de dimensionalidad $U \cdot V \cdot 5$ que codifica los bordes dominantes dentro de nuestra imagen.

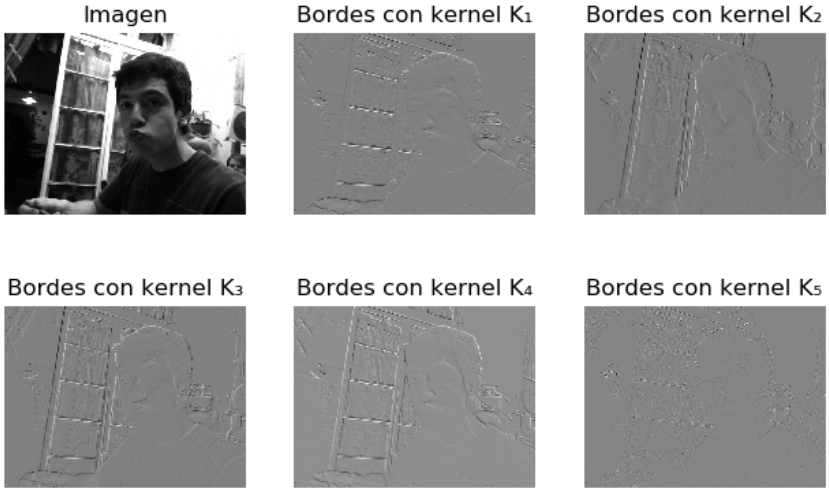


Figura 2.3: Distintos bordes obtenidos al filtrar la imagen con los kernels definidos anteriormente.

2.1.4. Histograma de Grises

Sea I una imagen en espacio de color RGB de tamaño $N \times M \times 3$. Se denomina a I' la transformación de la imagen a escala de grises de dimensiones $N \times M$. Se divide I' con una grilla de $U \times V$ y se refiere a cada celda como ${I'}_{uv}$, en donde los subíndices u y v representan la fila y columna de la grilla.

Para cada celda I'_{uv} se calcula un histograma H_{uv} con un número B de bins. Un ejemplo de esto es en la Figura 2.4. Se puede definir el descriptor de Histograma de Grises D como la concatenación de estos histogramas:

$$D = (H_{00}H_{10}...H_{U0}H_{01}...H_{0V}...H_{UV}).$$

D es un vector de dimensionalidad $U \cdot V \cdot B$ que codifica la información de las intensidades de grises en nuestra imagen.

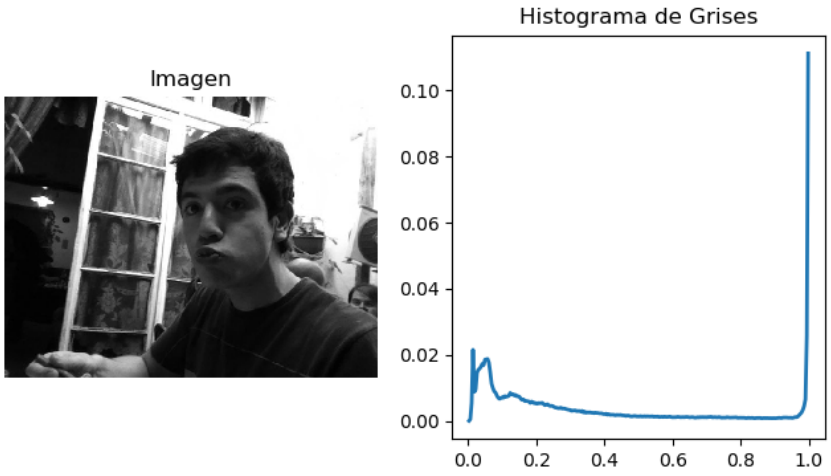


Figura 2.4: Histograma de de intensidades de grises para una imagen.

2.2. Clasificadores

El problema de clasificación corresponde a poder determinar a que clase corresponde un nuevo elemento observado. Un clasificador es, por tanto, un algoritmo que implementa la clasificación.

Estos clasificadores pueden ser supervisados, en cuyo caso el algoritmo aprende basado en ejemplos que relacionan la entrada con una salida determinada o no supervisados, en cuyo caso el algoritmo intenta modelar la densidad de probabilidad sobre las entradas. Para efectos de esta investigación, se utilizan algoritmos de aprendizaje supervisado.

2.2.1. Máquina de Vectores de Soporte

Las máquinas de vectores de soporte (SVM, por su sigla en inglés *support-vector machine*), son modelos de aprendizaje supervisado utilizados para clasificación [7].

En este modelo se consideran los datos como puntos en un espacio de alta dimensionalidad, y se busca generar el mejor hiperplano separador entre los elementos de distintas clases.

Para esto, sea x_i un elemento en nuestro espacio de características y sea $H = (w^T x + b)$ un hiperplano separador, se define $r = \frac{w^T x_i + b}{||w||}$ como la distancia entre x_i y H.

Los elementos x_i que están más cerca de H son los vectores de soporte, y se define el margen ρ como la distancia entre los vectores de soporte. Se define el conjunto de entrenamiento como $(x_i, y_i)_{i=1...n}$, con x_i el elemento en el espacio de características y y_i en $\{-1, 1\}$. Para cada par (x_i, y_i) se tiene que:

$$w^T x_i + b \leq -\rho/2, \quad si\ y_i = -1$$
$$w^T x_i + b \geq \rho/2, \quad si\ y_i = 1.$$

De aquí, se desprende que

$$y_i \left(w^T x_i + b \right) \geq \rho/2.$$

En particular, se tiene la igualdad cuando x_i es un vector de soporte x_s. En este caso, la distancia se puede calcular como $r = \frac{y_s\left(w^T x_s + b\right)}{||w||} = \frac{1}{||w||}$, por lo que podemos expresar el margen en función de w y b como $\rho = 2r = \frac{2}{||w||}$. A partir de esto, se obtiene el siguiente problema de optimización:

Encontrar w y b tal que se maximice $\rho = \frac{2}{||w||}$ y que para todo par en $(x_i, y_i)_{i=1...n}$, se cumpla $y_i \left(w^T x_i + b \right) \geq 1$.

Sin embargo, esto solo nos permite resolver problemas lineales. Para resolver no lineales, se puede mapear el espacio de características a uno no lineal a través de una función Kernel:

Sea $x, x' \in X$ un espacio de entradas arbitrarias. Se definen las funciones Kernel como

$$k: \ X \times X \to \mathbb{R}, \ (x, x') \mapsto k(x, x')$$
$$k\left(x, x'\right) = \left\langle \phi\left(x\right), \phi\left(x'\right) \right\rangle \ ,$$

donde $\phi\left(x\right)$ es un mapeo hacia un espacio $\mathcal{H}$, en donde el problema si puede ser resuelto.

Para que un Kernel represente efectivamente el producto punto de un mapeo $\mathcal{H}$, debe cumplir con las condiciones de Mercer:

- $K\left(x, x'\right)$ es simétrico
- La matriz $\begin{bmatrix} K(x_1, x_2) & \cdots & K\left(x_1, x_N\right) \\ \vdots & \ddots & \vdots \\ K\left(x_N, x_1\right) & \cdots & K\left(x_N, x_N\right) \end{bmatrix}$ es semi-definida positiva para cualquier $x_i \in X$.

2.2.2. Random Forest

Los Random Forest (también conocidos como "Bosques Aleatorios"), son un conjunto de arboles predictores que a través de su voto mayoritario clasifican un elemento. Los arboles, al no ser correlacionados, presentan poca varianza en su conjunto. Esto quiere decir que mientras la predicción de un único árbol es sensible al ruido, no lo es para la predicción conjunta de todo el bosque, siempre que se cumpla que no sean correlacionados [4].

Sea $(x_i, y_i) \in T$ pares del conjunto de entrenamiento. Se selecciona con reposición una cantidad B de veces n pares de elementos pertenecientes a T, denominado T_b. Luego, se entrena un único árbol predictor f_b con T_b. Este proceso se repite para cada $b = 1, ..., B$.

Las predicciones se pueden realizar para un elemento x' como

$$\hat{f} = \frac{1}{B} \sum_{b=1}^{B} f_b\,(x'),$$

o alternativamente, tomando el voto mayoritario de las predicciones de los arboles.

Para asegurar que los arboles no estén correlacionados, a cada árbol predictor se le entrega un subconjunto aleatorio de características del vector de características completo. En caso contrario, si existen algunas características que influyan fuertemente en el valor de y_i, muchos arboles dentro del bosque tenderán a elegir dichas características para predecir, lo que hacen que estén correlacionados.

2.2.3. K Nearest Neighbors

K Nearest Neighbors (K Vecinos Más Próximos) o KNN, es un método de clasificación supervisada, que sirve para estimar la función de densidad de probabilidad de que un elemento x pertenezca a una clase C [19].

Sea $x = (c_1, c_2, ..., c_n)$ un elemento en un espacio de características X. Sea $d(x_i, x_j)$ una función de distancia en X. Para clasificar x, sean $x_1, x_2, ..., x_k$ los elementos más cercanos a x según la función de distancia $d(x_i, x_j)$. Dado esto, x pertenecerá a la clase con mayor frecuencia dentro de $x_1, x_2, ..., x_k$.

La elección del parámetro k dependerá de los datos, por lo cuál es un valor que debe ser optimizado.

2.2.4. Regresión Logística

La regresión logística es un modelo de clasificación [25] que intenta modelar las probabilidades de que nuestro elemento pertenezca a una clase utilizando una función logística:

$$f(w^T x) = \frac{1}{1 + e^{w^T x}}.$$

Se intenta optimizar w de tal forma que la función logística f generalice los datos de mejor forma. Para esto, se busca minimizar la función de costo

$$J\left(w\right) = \frac{1}{N} \sum_{i=1}^{N} \left(z y_i log(\hat{y}_i) + (1 - y_i)\, log(1 - \hat{y}_i)\right),$$

siendo y_i la clase real y $\hat{y}_i$ la estimada.

2.2.5. Gaussian Naive Bayes

Naive Bayes es un algoritmo de aprendizaje basado en el teorema de Bayes [29], con el supuesto "naive" o ingenuo de que todas las características de un vector de características son condicionalmente independientes entre ellas. El teorema de Bayes plantea que, dada una variable de clase y y un vector de características dependientes $(x_1, \ldots, x_n)$, se tiene que

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}.$$

Usando el supuesto de que todas las características son condicionalmente independientes, se tiene que

$$P(x_i | y, x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = P(x_i | y),$$

para todo i, se puede simplificar esta relación como

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}.$$

Como $P(x_1, \ldots, x_n)$ es constanste dado el input, se puede clasificar según

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

se puede utilizar la probabilidad máxima a posteriori para calcular $P(y)$, que es la frecuencia relativa de la clase y en el conjunto de entrenamiento. Para calcular $P(x_i \mid y)$, se asume que que la probabilidad de las características sigue una distribución gaussiana

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right),$$

en donde los parámetros σ_y y μ_y, que representan la varianza y el promedio de la clase y, se estiman utilizando la máxima verosimilitud.

2.3. Redes Neuronales

Las redes neuronales[18] son modelos inspirados en el funcionamiento del cerebro humano. Existen varios tipos de arquitecturas, pero nos centraremos en dos: perceptrones multicapa, y redes convolucionales.

2.3.1. Perceptrón Multicapa

Un perceptrón [9] es el análogo a una neurona, que se entrena como un clasificador binario. Matemáticamente, se define como

$$y_k = \varphi\left(w_k^T \mathbf{x}_k\right) = \begin{cases} 1 & w_k^T \mathbf{x}_k \geq 0 \\ 0 & w_k^T \mathbf{x}_k < 0 \end{cases},$$

donde w_k es un parámetro llamado peso sináptico, cuyo valor se aprende durante el entrenamiento, φ es una función de activación y x_k es un elemento.

Se puede organizar a los perceptrones en capas, de tal forma que en todas las salidas de los perceptrones de una capa se encuentren conectados a los perceptrones de la capa siguiente, y no hayan conexiones de perceptrones dentro de una misma capa. Esto se conoce como perceptrón multicapa, y es la arquitectura más básica de redes neuronales.

Los perceptrones multicapa cuentan con una entrada, una cantidad arbitraria de capas ocultas, y una capa de salida. El método de entrenamiento usualmente utilizado para ajustar los pesos sinápticos de cada perceptrón tiene dos etapas, Forward Pass y Back Propagation:

- Forward Pass: Entregar una entrada del set de entrenamiento y evaluar su salida. Luego, se calcula el error entre la salida real y la de la red.
- Back Propagation: Propagar el error desde la salida hasta la entrada y actualizar los pesos sinápticos.

Este método busca minimizar el error a través de la red, lo que hace que cada perceptrón converja a sus parámetros correspondientes.

2.3.2. Red Convolucional

Para el análisis de imágenes, los perceptrones multicapa presentan dos problemas fundamentales. Como cada capa es completamente conectada, se requieren muchos pesos sinápticos entre cada neurona, lo que hace el entrenamiento muy lento y en ocasiones incapaz de converger; además, no se aprovecha la estructura espacial que tienen los datos.

Con esto en mente, se introducen las redes convolucionales [1]. La diferencia principal entre estas redes es que se introducen neuronas convolucionales. Estas en lugar de tener un vector con un tamaño igual a la entrada como peso sináptico, tienen una matriz llamada kernel. Los kernels suelen tener dimensiones muy pequeñas, y se operan con su entrada a través del producto convolución. Esto soluciona ambos problemas: en lugar de aprender un peso para cada entrada se debe aprender solamente el kernel, y el producto convolución genera una salida utilizando la entrada de forma local, beneficiandose de la estructura espacial de los datos. Así, la red aprende una serie de kernels que sirve para extraer características.

Además, se introducen las neuronas de muestreo o pooling. Su función es realizar un muestreo de la entrada para poder reducir su resolución. Esto le da a la red tolerancia ante el ruido, además de ir reduciendo el tamaño de la entrada, lo que hace a la red más rápida.

También se cuenta con neuronas de clasificación, que son perceptrones. Usualmente las últimas capas de la red son capas de clasificación, las cuales toman las características extraídas por las neuronas convolucionales y de muestreo, clasificando la entrada.

Sin embargo, uno puede hacer uso de de la red sin las capas de clasificación para poder extraer las características que la red aprende para poder clasificar elementos, y utilizar dicho vector con otros métodos de clasificación; de esta forma, una red convolucional puede ser utilizada tanto como un descriptor o como un clasificador.

2.3.3. DeCAF7

DeCAF [6] (por su nombre en inglés, *Deep Convolutional Activation Features*) es una red neuronal convolucional diseñada específicamente para extraer características de imágenes. Su propósito es ser entrenada en un dataset grande de reconocimiento de objetos para luego poder ser reutilizada en tareas genéricas. Para lograrlo, fue entrenada como un clasificador sobre el ImageNet Large Scale Visual Recognition Challenge 2012 [24], que consta con más de 10 millones de imágenes a través de más de 10,000 clases.

DeCAF puede ser utilizada directamente como un clasificador, pero se propone utilizar las características profundas extraídas por las distintas capas de la red. Se denota como DeCAF_n a la capa cuyas características se extraen, con n entre 1 y 7. Por la arquitectura de la red, se sugiere utilizar las capas DeCAF_5, DeCAF_6 o DeCAF_7, pues las anteriores aún no han pasado por todas las capas convolucionales de la redes, por lo que no es probable que contengan características interesantes.

Para constatar que la red es capaz de ser reutilizada en otros dominios, se entrenó una má-

quina de vectores de soporte con las características de DeCAF_6 en el dataset Caltech-101 [14], obteniendo una precisión de 86.9 %. El buen desempeño que tiene DeCAF es atribuido en parte al dataset de gran escala con el que fue entrenado [6].

2.3.4. ResNet152

ResNet (por su nombre en inglés, *Residual Neural Network*) [10], es una arquitectura de red neuronal que busca encontrar una forma de optimizar redes muy profundas sin perder precisión.

Las arquitecturas tradicionales sufren degradación en su precisión al aumentar la cantidad de capas debido a lo complejo que se vuelve optimizarlas, por lo que no se puede utilizar el máximo potencial de redes profundas. ResNet, sin embargo, puede tener una profundidad mucho mayor sin sufrir degradación de precisión por su diseño fácil de optimizar. De esta forma, logra alcanzar un desempeño substancialmente mejor que otras redes, ya que puede obtener todo el beneficio de su profundidad.

Para mejorar la optimización, se hace uso del residual de una función. Sea $\mathcal{H}(x)$ el mapeo que generará un subconjunto de capas. Con la hipótesis de que varias capas pueden aproximar asintóticamente funciones complicadas, entonces también puede aproximar asintóticamente la función residual, es decir, $\mathcal{H}(x)-x$. Así, se puede buscar aproximar directamente la función residual, que podemos definir como $\mathcal{F} := \mathcal{H}(x)-x$. De esta forma, la función original se puede escribir como $\mathcal{F} + x$.

Con lo anterior, se puede definir formalmente un bloque dentro de la red como

$$y = \mathcal{F}(x, W_i) + x,$$

donde x e y son las entradas y salidas del bloque, y $\mathcal{F}(x, W_i)$ representa el mapeo residual a aprender. La operación $\mathcal{F} + x$ se puede realizar con un .ªtajo"dentro de la red, que conecta directamente la entrada a la salida correspondiente. Esta operación no introduce parámetros nuevos ni complejidad adicional, por lo que es una opción atractiva.

Al igual que en el caso de DeCAF, se puede utilizar la red completa para clasificar, o se puede extraer las características profundas de ResNet para luego entrenar otro tipo de clasificador. De la misma forma de lo anterior, se puede obtener una ventaja del hecho que fue entrenada con un dataset de gran escala, por lo que se pueden obtener buenas características para clasificar.

Capítulo 3

Métodos de clasificación de Memes

Para trabajar el problema de clasificación, se propusieron dos líneas principales a seguir. La primera consta en utilizar los descriptores visuales presentados en el capítulo anterior para extraer vectores de características de cada imagen y ajustar distintos clasificadores con ellas. La segunda utiliza las imágenes directamente para entrenar una red neuronal para realizar la clasificación. De esta forma se puede realizar una comparación entre el estado del arte de métodos clásicos para clasificar imágenes junto con el de redes neuronales.

Además, en este capítulo se detalla la generación del dataset utilizado a lo largo de los experimentos.

3.1. Memes

En la actualidad, las redes sociales se han convertido en una parte cada vez más importante de la vida diaria. Lo que inicialmente surgió como una forma de poder comunicar a grupos de personas evolucionó de manera dramática y sin precedentes, convirtiéndose en una fuente de entretenimiento y difusión [20]. Pero dentro de este espacio digital surgen nuevas dificultades que tienen impacto en la sociedad, pero que aún no han sido estudiadas completamente.

El impacto social y político que tienen las redes sociales es claro. Sabemos que estas permiten a la ciudadanía poder coordinarse y poder generar movimientos sociales con un alcance a nivel nacional, desde movilizaciones feministas hasta protestas contra el sistema de pensiones. Sin embargo, sabemos que estas también han logrado generar un impacto negativo [3] [23]. Por ejemplo, la credibilidad de distintas instituciones se ve dañada debido al esparcimiento de noticias falsas, o se promueven discursos de odio con facilidad, incluso de manera no intencional por usuarios. [3]

El estudio de este complejo problema es abordado por el Instituto Milenio Fundamentos de los Datos (IMFD), bajo un proyecto emblemático que trata sobre Generación de Estructuras de Información Robusta. Lo que se busca en este proyecto es poder estudiar y enfocarse en problemas que son influenciados por la falta de información robusta, entendida como conocimiento veraz que puede ser extraído de fuentes no estructuradas. Dado el volumen masivo de fuentes no estructuradas como también su variada composición, se busca crear sistemas automáticos para estudiarlos.

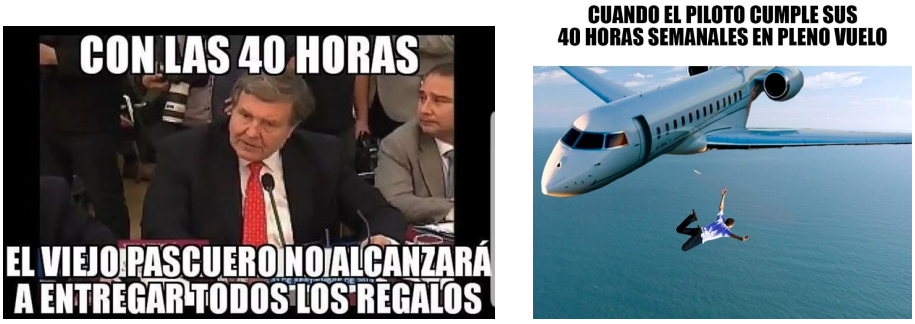
Uno de estos tipos de fuente que ha cobrado mucha predominancia son los denominados memes [20]. Estos consisten en información multimedia, usualmente imágenes acompañadas de texto para poder expresar una idea, que suele tener un tono humorístico o sarcástico. Son de muy variada índole, se viralizan con facilidad y pueden ser creados por cualquier persona utilizando herramientas hechas con este fin. Por esto su estudio resulta muy relevante dentro del proyecto del IMFD.

Otros investigadores ya han realizado estudios sobre memes. Frecuentemente se busca, por ejemplo, poder categorizar la opinión que entrega un meme [2] para poder determinar si contiene mensajes de odio, o en general, si contiene un mensaje negativo o positivo, y si está dirigido hacia algún grupo. Otros estudios más complejos buscan poder determinar como los memes se viralizan, influencian la mentalidad de comunidades de Internet, y cómo afectarán sus tendencias a futuro [11]. Resulta ser bastante interesante, pues muchos memes serán los mismos o muy similares, pero con mensajes totalmente opuestos, y se distribuirán en sus comunidades respectivas.

Sin embargo, estos estudios trabajan sobre memes considerándolos como elementos ya extraídos, pero la literatura existente en su obtención es muy precaria. Aunque existen servicios que permiten distinguir memes entre imágenes [16], suele ser más común obtenerlos de manera manual, ya sea extrayéndolos de repositorios de memes, en el cuál se pierde mucho contexto e información acerca del entorno del meme, o clasificándolos desde redes sociales, lo que es muy lento, especialmente considerando el volumen de datos que se genera hoy en día a través de todas las redes sociales.

Es así que el primer paso que se daría para el análisis de esta fuente correspondería a poder clasificar una imagen como un meme de manera automática. Para esto se propone el uso de descriptores visuales que permitan caracterizarlos de forma óptima para su posterior clasificación, pudiendo determinar si una imagen es o no un meme. Sin embargo, esta es una tarea compleja, pues los memes se encuentran en constante evolución, lo que abre varias interrogantes. Por ejemplo, dada la gran variedad de estos ¿existirán memes que se puedan clasificar de forma más precisa? ¿Serán todos los descriptores visuales efectivos para todas las variedades de memes?

En la Figura 3.1 podemos ver un ejemplo de las interrogantes anteriores, planteadas pensando en la gran variedad que tiene nuestro elemento a estudiar. Ambas imágenes son memes que tratan acerca de la misma temática, pero visualmente son bastante diferentes; aunque comparten un texto similar, el contenido de las imágenes no guardan relación alguna, el texto no tiene el mismo color, y aunque estén en posiciones similares dentro del meme, en la Figura 3.1 (a) este se encuentra directamente sobre la imagen, mientras que en la Figura 3.1 (b) este se encuentra fuera de esta, ubicado sobre un fondo plano.



(a) (b)

Figura 3.1: Memes con la misma temática pero con distinta estructura visual.

Esto hace que el problema sea bastante desafiante e interesante de abordar desde una perspectiva clásica, refiriéndonos con esto al uso de descriptores visuales en lugar de redes neuronales profundas. Nuestro objeto de estudio tiene elementos comunes que suelen repetirse, independiente de la variedad en que se presente, desde que comenzaron a popularizarse hasta el día de hoy. Gracias a estos patrones presentes en los memes, nosotros podemos aprovecharlos para generar métodos de clasificación que requieran de un volumen de datos menos masivo y de menor capacidad de procesamiento para obtener resultados precisos.

3.2. Métodos de clasificación

3.2.1. Descriptores y clasificadores

En esta línea de clasificación, se hace uso de descriptores visuales para extraer un vector de características para luego entrenar un clasificador con ellos. Podemos separar nuestros descriptores en dos grupos:

- Clásicos: Estos descriptores realizan cálculos sobre la imagen, generalmente después dividirla en varias celdas. Luego los valores de interés para cada celda son acumulados en un histograma o en otros tipos de representaciones, que posteriormente se utilizan para generar su vector de características. A este grupo pertenecen los descriptores de bordes, distribución de color, histograma de grises e histograma de gradientes orientados.
- Deep Features: Estos descriptores provienen de redes neuronales convolucionales profundas. A través de su entrenamiento, estas redes aprenden distintos filtros que permiten extraer características específicas de cada imagen que corresponden a una combinación de varios componentes de estas, como colores, formas y texturas. Para utilizar este tipo de descriptor, se utiliza la imagen como input de la red, y esta entrega un vector de características. A este grupo pertenecen los descriptores DeCAF7 y ResNet152.

Cada clasificador estadístico debe ser ajustado para cada uno de los descriptores en ambos grupos para poder determinar el desempeño de cada par descriptor-clasificador posible.

Además de esto, en algunos clasificadores es necesario encontrar los parámetros óptimos para cada descriptor. En caso contrario, el desempeño de estos se puede ver perjudicado. Esto se logra haciendo una búsqueda en el espacio de parámetros. Para cada clasificador, se define un rango discreto de parámetros sobre los cuales se realiza dicha búsqueda. Luego, para cada combinación de parámetros se entrena nuestro clasificador N veces, obteniendo el promedio y la varianza de cada métrica. Así, nosotros podemos encontrar los parámetros óptimos dentro de los rangos establecidos para cada clasificador al ser entrenado en cada uno de los descriptores. En caso de que sea necesario, se puede ampliar el rango de búsqueda en los parámetros posibles para buscar un mejor óptimo.

Clasificador	**Parámetros**
Perceptrón Multicapa	Arquitectura de capas ocultas
	Función de activación
	Solver
	Learning Rate
	Alpha
KNN	Algoritmo
	Número de vecinos
	Tipo de distancia
Máquina de vectores de soporte	Kernel
	Parámetro de regularización
	Grado

Tabla 3.1: En esta tabla se muestran los distintos parámetros que deben ser optimizados para los distintos clasificadores. Estos deben ser optimizados cada vez que se entrenan los modelos para los distintos descriptores.

Clasificador	**Parámetros**
Regresión Logística	Solver: SAG
Gaussian Naive Bayes	-
Random Forest	Número de estimadores: 300

Tabla 3.2: En esta tabla se muestran los parámetros utilizados para los clasificadores que no requerían realizar una optimización de estos.

3.2.2. Redes Neuronales

En esta línea de clasificación, se hace uso de redes neuronales. A diferencia del caso anterior, la red neuronal se encarga del proceso completo, es decir, extrae las características e inmediatamente las utiliza para realizar la clasificación. Para los experimentos, se utiliza la red neuronal ResNet152.

Una diferencia fundamental entre el caso de los clasificadores anteriores y las redes neuronales es que la red necesita un volumen de datos muy grande. Sin embargo, es posible implementar técnicas que mejoran el desempeño de nuestra red incluso con datos limitados. Estas técnicas son *transfer learning* [26] y *fine tuning* [28].

Transfer Learning

Esta técnica consiste en utilizar el conocimiento que se aprende al solucionar un problema y utilizarlo para resolver un problema diferente, pero que aún se encuentre relacionado [26]. Por ejemplo, el conocimiento adquirido para poder reconocer gatos puede ser utilizado para reconocer tigres.

Para implementar esto, se puede utilizar la red neuronal pre-entrenada sobre la base de datos ImageNet. Esta base de datos cuenta con sobre 14 millones de imágenes de sobre 1000 clases distintas, que incluye personas, animales, plantas y objetos entre otras cosas. Como nuestro problema presenta imágenes de muy variado contenido, esto presenta una ventaja para la red pues los filtros aprendidos podrán interpretarlas apropiadamente.

Fine Tuning

Esta técnica es una aplicación del caso anterior para redes neuronales. Como se explicó anteriormente, las redes neuronales convolucionales constan de varias capas, siendo la capa de salida una capa completamente conectada entrenada para clasificar el output de las capas convolucionales anteriores [28].

Estas capas están entrenadas para clasificar las clases específicas de cada problema. Por lo tanto, es posible reemplazar esta capa para que aprenda a clasificar las clases de un problema nuevo. Esto requerirá reentrenar esta nueva capa, pero sin necesidad de reentrenar el resto de la red, pues gracias a la técnica de *transfer learning*, se reutilizan los filtros ya aprendidos, y solamente es necesario ajustar la última capa encargada de realizar la clasificación.

3.3. Dataset

El dataset utilizado fue generado para el IMFD por la Facultad de Comunicaciones de la Pontificia Universidad Católica [12]. Este dataset se construyó recolectando todos los mensajes de twitter con imágenes de usuarios chilenos publicados desde mayo hasta octubre de 2019. Luego, cuatro expertos clasificaron manualmente una muestra de 52,000 imágenes de un total de más de 618,284.

3.3.1. Medidas de fiabilidad

Para la anotación se consideran dos medidas de fiabilidad, ICR y alfa de Krippendorf.

ICR

ICR (por su nombre en inglés, *inter-coder reliability*) [15] es el grado en que dos o más codificadores independientes acuerdan la codificación del contenido de interés con una aplicación del mismo esquema de codificación.

Alfa de Krippendorf

El Alfa de Krippendorf [13] es un coeficiente de fiabilidad calculado para medir el acuerdo entre dos o más codificadores independientes, utilizado para fenómenos típicamente no estructurados. Se calcula como

$$a = 1 - \frac{D_o}{D_e},$$

donde D_o es el desacuerdo de los observadores, y D_e es el desacuerdo esperado cuando la codificación de los elementos es atribuible al azar y no a las propiedades de los elementos en sí. Cuando $D_o = 0$ tenemos que $a = 1$, correspondiente al caso en que no existe desacuerdo entre los observadores y la fiabilidad es perfecta. Cuando $D_o = D_e$ tenemos que $a = 0$, correspondiente al caso en que los observadores no pueden distinguir entre los elementos a codificar, o se codificaran al azar.

3.3.2. Metodología de anotación

Para el propósito de clasificar la muestra, los expertos utilizaron la siguiente metodología de anotación:

1. Los expertos fueron entrenados hasta alcanzar un ICR de 90 % o superior, y un alfa de Krippendorff de 0.7 o superior. El ICR, fue calculado con una muestra de 2,000 imágenes no incluidas en la muestra final de 52,000 casos.

2. Cada experto clasificó 13,000 imágenes distintas en cuatro clases: Meme, No-Meme, Sticker y Dudoso. Además, para imágenes clasificadas como Memes y Stickers, los expertos transcribieron el texto en la imagen, entregando una descripción de los contenidos de la imagen (sujetos, lugares, acciones, etc.) y una interpretación semántica del contexto y la situación mostrada.

Detallando cada una de las cuatro clases señaladas anteriormente:

- Meme: Imagen (foto, ilustración, esquema, etc.) con texto sobrepuesto. Dicho texto debe ser un mensaje o lema humorístico con alto nivel de intertextualidad. Responden a la realidad social de las personas que los crean y comparten.

(a)

(b)

Figura 3.2: Ejemplos de memes.

- Sticker: Imagen que puede o no contener texto, pero que sirve para dar una respuesta o un mensaje conciso como “entendido”, “gracias” o “te amo”. No necesariamente presenta intertextualidad o humor, por lo que se trata de forma separada a la clase anterior.

(a)

(b)

Figura 3.3: Ejemplos de stickers.

- No-Meme: Imagen que no es meme ni sticker.
- Dudoso: Imagen ambigua que no se puede determinar a cual de las clases anteriores pertenece.

El dataset consta de 52,000 imágenes: 1,194 Meme, 1,443 Sticker, 49,347 No-Meme y 16 Dudoso. Las transcripciones, descripciones y anotaciones de las imágenes conforman un vocabulario de 8,669 palabras.

Capítulo 4

Evaluación Experimental

En este capítulo se detalla el diseño de los experimentos, como también varios resultados de interés y un análisis de estos. Solamente se incluyen los resultados más interesantes dentro de las secciones siguientes, pero el resto de ellos se encuentran en el anexo.

4.1. Métricas

Para poder evaluar el desempeño de los clasificadores y descriptores, es necesario primero establecer las métricas según las cuales estos serán evaluados.

4.1.1. Matriz de confusión

Una matriz de confusión es una matriz que nos permite visualizar el desempeño de un algoritmo de clasificación. Denominamos los elementos según su clasificación como se indica en la Tabla 4.1.

		Clase predicha	
		1	0
Clase real	1	Verdadero Positivo (VP)	Falso Negativo (FN)
	0	Falso Positivo (FP)	Verdadero Negativo (VN)

Tabla 4.1: Matriz de confusión. Las columnas representan la clase predicha, y las filas la clase real.

Con esta tabla se puede obtener una idea intuitiva de cómo están siendo clasificadas las clases.

4.1.2. Precisión

La precisión es la razón entre las observaciones positivas y el total de las predicciones positivas. En términos de la matriz de confusión, se puede expresar como

$$Precisión = \frac{VP}{VP + FP}.$$

Su valor va entre 0 y 1, siendo mejor mientras más cercano a 1 sea. Intuitivamente, nos indica cuantos elementos fueron correctamente clasificados de los que se predijeron como positivos.

4.1.3. Exactitud

La exactitud es la razón entre las observaciones correctas del total de observaciones hechas. En términos de la matriz de confusión, se puede expresar como

$$Accuracy = \frac{VP + VN}{VP + FP + VN + FN}.$$

Su valor va entre 0 y 1, siendo mejor mientras más cercano a 1 sea. Intuitivamente, nos indica cuantos elementos fueron correctamente clasificados del total. La diferencia entre precisión y exactitud recae en que la exactitud son las correctamente clasificadas para ambas clases, mientras que la precisión solamente se preocupa de la clase positiva.

4.1.4. Exhaustividad (Recall)

La exhaustividad es la razón entre las observaciones correctas positivas y el total de observaciones positivas. En términos de la matriz de confusión, se puede expresar como

$$Recall = \frac{VP}{VP + FN}.$$

Su valor va entre 0 y 1, siendo mejor mientras más cercano a 1 sea. Intuitivamente, indica cuantos elementos positivos fueron correctamente clasificados del total de elementos positivos. Es decir, nos indica que tan completa fue la clasificación para elementos positivos.

4.1.5. F1

La puntuación F1 es el promedio armónico entre la precisión y la exhaustividad, por lo que toma en cuenta tanto los falsos positivos como negativos. Se puede expresar como

$$F1 = 2 \times \frac{Precision \times Recall}{Precisión + Recall} = \frac{VP}{VP + \frac{1}{2}(FP + FN)}.$$

Su valor va entre 0 y 1, siendo mejor mientras más cercano a 1 sea. Es más difícil de entender intuitivamente, pero ayuda a medir el compromiso entre la precisión y la exhaustividad, lo que es una medida útil de conocer cuando las clases no están distribuidas uniformemente.

4.1.6. Validación Cruzada (Cross-Validation)

La validación cruzada es una técnica utilizada para evaluar los resultados de un análisis estadístico. Esta sirve para garantizar que dichos resultados son independientes de la partición que se haga entre el conjunto de entrenamiento y validación.

Todo nuestro dataset es dividido en dos particiones: entrenamiento y prueba. Al utilizar validación cruzada, durante el entrenamiento, se genera una nueva partición dentro del conjunto de entrenamiento, obteniendo el conjunto de entrenamiento propiamente tal, y un conjunto de validación. Esta partición entre entrenamiento y validación se realiza un número K de iteraciones, y para cada iteración, se entrena el clasificador y se valida con las particiones dadas. Una vez finalizadas las iteraciones, se procede a calcular una media aritmética entre las métricas obtenidas, además de la varianza, para obtener una idea precisa de como funciona el clasificador independiente de los datos.

4.1.7. Combinaciones de Descriptores y Clasificadores

En la Tabla 4.2 se muestran todos los descriptores y clasificadores a utilizar. Cada descriptor se utiliza con cada clasificador, por lo que se cuenta con un total de 36 combinaciones totales. Estas combinaciones se utilizan en el primer y tercer experimento que se detallan en la sección siguiente. No se realiza una combinación de los mismos descriptores pues nos interesa estudiar el desempeño de cada descriptor de forma individual. Para el segundo experimento, se utiliza ResNet152 como clasificador directamente con las imágenes.

Descriptores	**Clasificadores**
Distribución del Color	Máquina de Vectores de Soporte
Histograma de Gradientes Orientados	Random Forest
Histograma de Bordes	K Nearest Neighbors
Histograma de Grises	Regresión Logística
DeCAF7	Naive Bayes Gaussiano
ResNet152	Perceptrón Multicapa

Tabla 4.2: Tabla de resumen con las combinaciones de descriptores y clasificadores utilizados en los experimentos.

4.2. Diseño Experimental

Se definen tres experimentos principales:

1. Evaluar el desempeño de las distintas combinaciones de clasificadores y descriptores con las métricas señaladas en la sección anterior.
2. Modificar una implementación de ResNet152 y reentrenarla para evaluar su desempeño.
3. Clasificar las imágenes con todas las combinaciones posibles y analizar la posible existencia de memes que sean más difíciles de clasificar, es decir, que no sean clasificables por ninguna de ellas.

Los resultados de los experimentos se encuentran en la sección de resultados experimentales.

4.2.1. Primer Experimento

El primer experimento se realiza en tres etapas principales:

1. Extracción de características.
2. Optimización de parámetros para clasificadores.
3. Entrenamiento y evaluación.

La primera etapa se realiza iterando a través de todas las imágenes y generando una base de datos que contiene las características extraídas con cada uno de los descriptores [8]. Así, las características quedan disponibles para poder trabajar en los experimentos.

La segunda etapa consta de la optimización de los parámetros de cada clasificador, para cada descriptor. Para esto, primero se realiza un submuestreo aleatorio de los datos, para poder generar un subconjunto de nuestras clases que tengan la misma cantidad de elementos. De esta forma podemos balancear las clases. Luego, utilizando validación cruzada, se entrena cada combinación de descriptor y clasificador una cierta cantidad de veces, que dependerá de los parámetros a explorar. Una vez obtenidos los resultados, se puede evaluar el uso de distintos valores para los parámetros, o se pueden seleccionar los que obtuvieron mejores resultados.

La tercera etapa consiste en entrenar las combinaciones de clasificadores con los parámetros encontrados en la etapa anterior. Una vez finalizado el entrenamiento, se procede a evaluar todas las métricas establecidas. Todo lo anterior se realiza principalmente con la librería scikit-learn [22].

4.2.2. Segundo Experimento

En este experimento se toma una implementación existente [21] y preentrenada de la red neuronal convolucional profunda ResNet152, y se modifica su arquitectura para la clasificación de imágenes.

Para esto se reemplaza la última capa completamente conectada de la red, por una capa que solamente tiene tres elementos en la salida, que representan a cada clase. Como la red se encuentra preentrenada con ImageNet, se congelan los pesos de todas las capas salvo la de la salida, pues estas ya tienen las características profundas aprendidas. Esto además acelera el proceso de convergencia. Se utiliza como función de error entropía cruzada, y se utiliza el descenso de gradiente estocástico para entrenar la red.

Al igual que en el experimento anterior, se realiza un submuestreo aleatorio de las imágenes para poder generar un subconjunto balanceado de las clases. Luego se preprocesan las imágenes y se entrena la red utilizando validación cruzada por 300 épocas o hasta que converja.

4.2.3. Tercer Experimento

En este último experimento se busca poder determinar la existencia de memes difíciles de clasificar. Esto nos permite encontrar una aproximación a una cota superior que, en teoría, si se tuviese un oráculo o método que nos permitiera elegir el clasificador y descriptor correcto para cada meme, se podría alcanzar al realizar clasificaciones.

Para esto, se realizó un submuestreo aleatorio de los datos para poder generar un subconjunto balanceado de las clases. Luego, se entrenan todas las combinaciones de clasificadores y descriptores con los parámetros obtenidos en el primer experimento, y con la red entrenada en el segundo experimento. Finalmente, para el conjunto de prueba, cada imagen y todos sus respectivos descriptores se entregan a los clasificadores correspondientes para realizar la predicción. En caso de que ningún clasificador pueda predecir correctamente la clase de la imagen, esta es considerada difícil.

Así, queda propuesto como trabajo futuro encontrar un método que permita alcanzar esta cota.

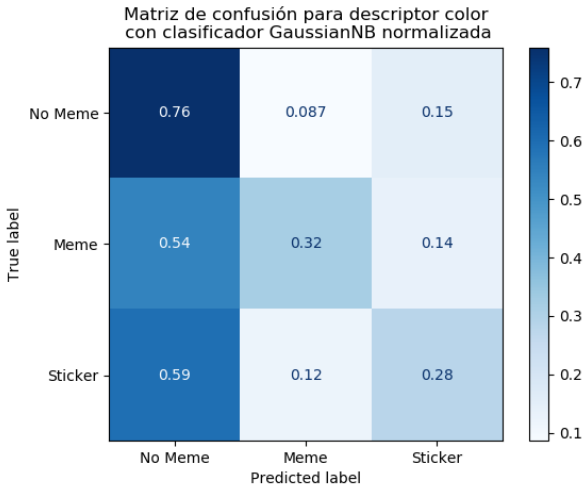
4.3. Resultados Experimentales

A continuación se presentan los resultados experimentales obtenidos. No se hace uso de la clase "dudoso"del dataset.

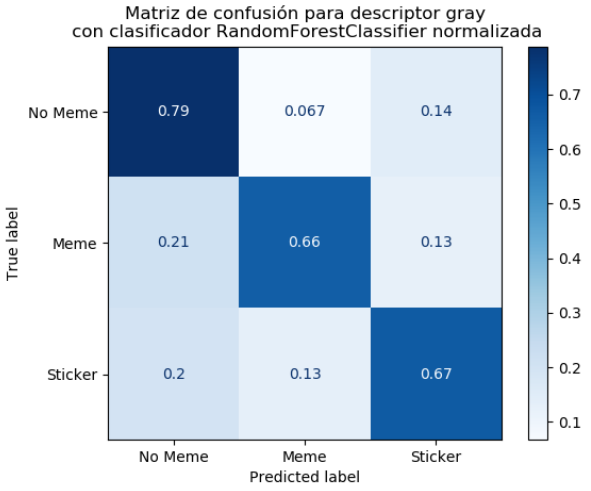
4.3.1. Matrices de confusión

Primer Experimento

Las matrices de confusión de la Figura 4.1 nos muestra tanto el mejor como el peor resultado obtenido al utilizar descriptores clásicos. En general, hay una variación en los resultados no despreciable dentro de los mismos descriptores cuando se usan distintos clasificadores. El resto de las matrices se encuentran en el anexo.



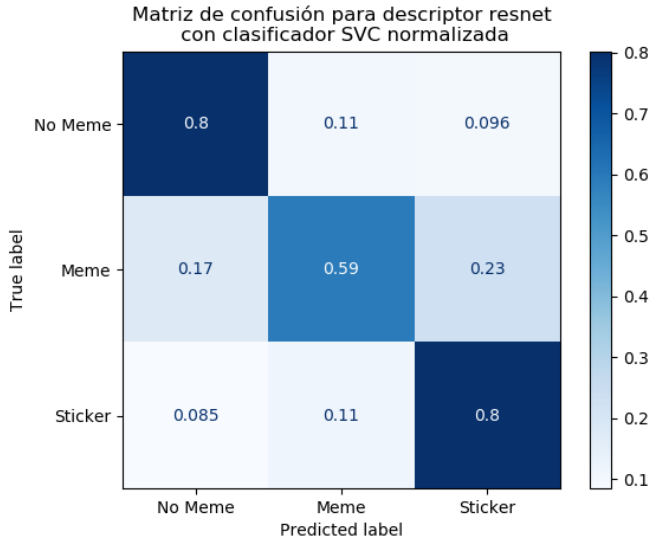
(a)



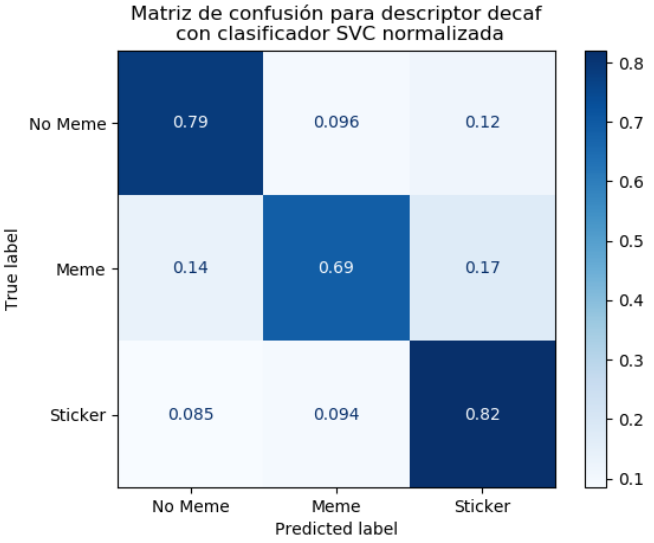
(b)

Figura 4.1: Matrices de confusión. (a) Es la combinación de clasificador y descriptor con la precisión más baja de los descriptores clásicos. (b) Es la combinación de clasificador y descriptor con la precisión más alta de los descriptores clásicos.

Las matrices de confusión de la Figura 4.2 nos muestra los resultados obtenidos al utilizar descriptores del estado del arte. Existe una variación en los resultados dependiendo de los clasificadores que se usen. El resto de las matrices se encuentran en el anexo.



(a)

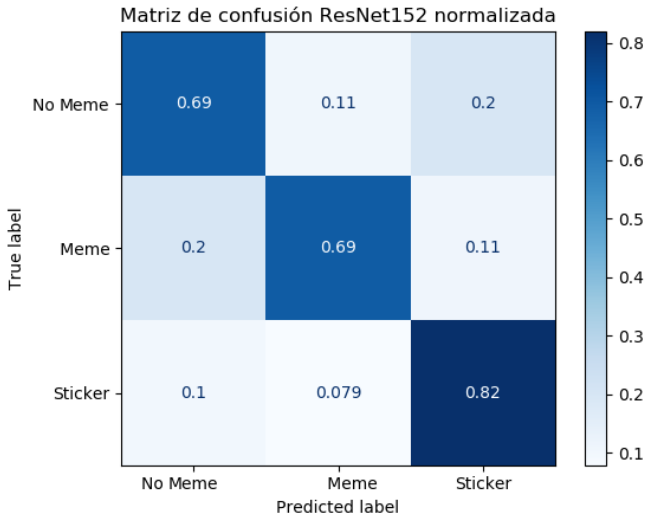


(b)

Figura 4.2: Matrices de confusión. Ambas utilizan deep features como descriptores, extraídos de redes neuronales.

Segundo Experimento

Las matrices de confusión de la Figura 4.3 nos muestra los resultados obtenidos al utilizar ResNet152 para realizar la clasificación.

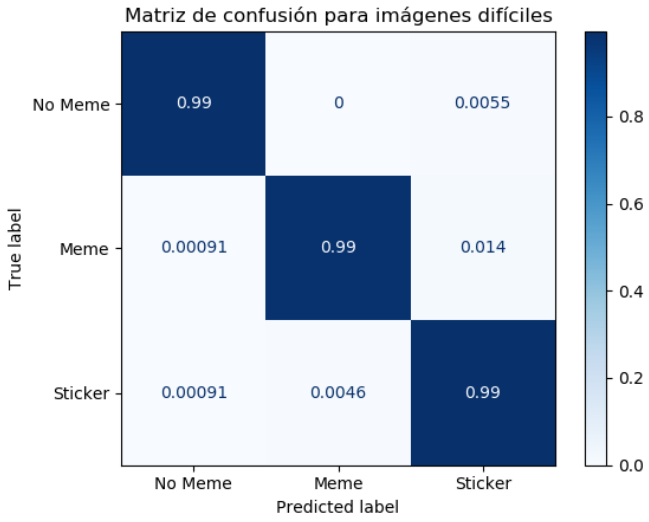


(a)

Figura 4.3: Matriz de confusión para ResNet152 al finalizar su entrenamiento.

Tercer Experimento

La matriz de confusión de la Figura 4.4 nos muestra como fue la clasificación para las imágenes difíciles, realizadas con todas las combinaciones de descriptores y clasificadores. En general, se obtuvo un 99 % de precisión a través de todas las clases.

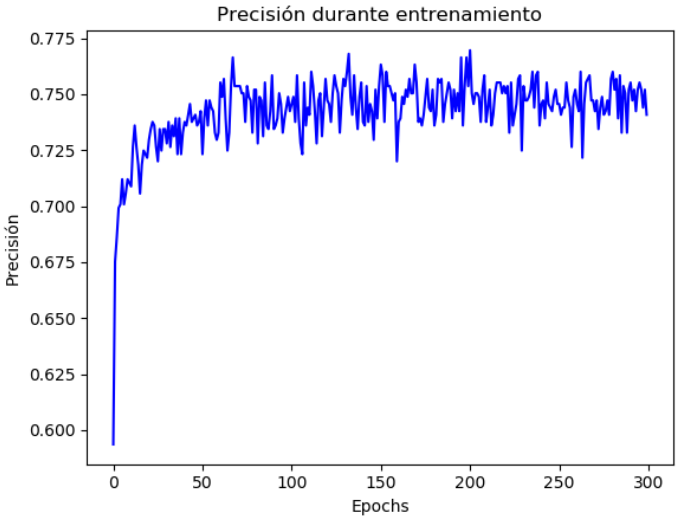


(a)

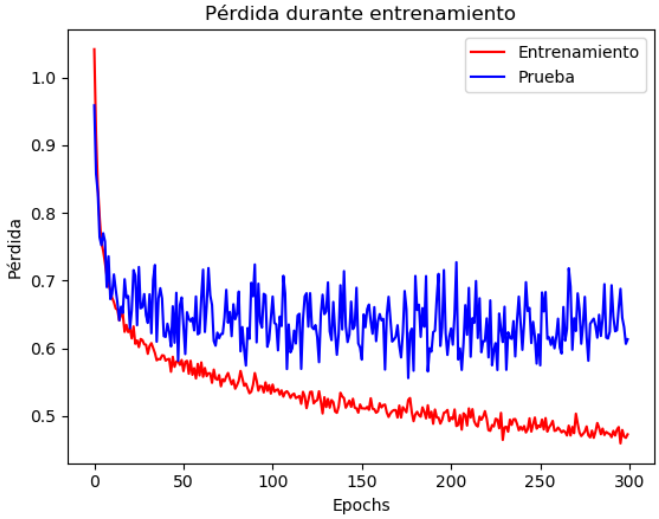
Figura 4.4: Matriz de confusión para imágenes difíciles a través del “oráculo”.

4.3.2. Evolución de ResNet152

Los gráficos de la Figura 4.5 nos muestra como fue el entrenamiento de la red para el segundo experimento. Se entrenó durante 300 épocas, donde la red deja de hacer avances significativos en su entrenamiento.



(a)



(b)

Figura 4.5: (a) Es la Precisión de la red durante el entrenamiento. (b) Es la evolución de la pérdida durante el entrenamiento..

4.3.3. Métricas

A continuación se presentan las métricas detalladas anteriormente con su respectivo error. Las Tablas 4.3, 4.4, 4.5 y 4.6 siguen la estructura de un descriptor y un clasificador, correspondientes al primer experimento. La Tabla 4.7 muestra las métricas para ResNet152 realizando el proceso completo de clasificación, correspondiente al segundo experimento.

	SVM	Random Forest	Naive Bayes Gausiano	Regresión Logística	Perceptrón Multicapa	K-Nearest Neighbors
Gris	64 % ± 6 %	70 % ± 5 %	51 % ± 5 %	53 % ± 8 %	59 % ± 5 %	54 % ± 6 %
Color	53 % ± 5 %	56 % ± 5 %	44 % ± 7 %	46 % ± 5 %	52 % ± 4 %	47 % ± 5 %
HoG	61 % ± 8 %	61 % ± 9 %	53 % ± 7 %	58 % ± 9 %	58 % ± 6 %	53 % ± 7 %
Bordes	57 % ± 7 %	56 % ± 5 %	49 % ± 5 %	53 % ± 6 %	55 % ± 6 %	53 % ± 5 %
ResNet152	71 % ± 8 %	63 % ± 5 %	64 % ± 7 %	62 % ± 8 %	69 % ± 7 %	60 % ± 7 %
DeCAF7	75 % ± 4 %	74 % ± 3 %	65 % ± 4 %	72 % ± 5 %	72 % ± 4 %	65 % ± 5 %

Tabla 4.3: Tabla de precisiones.

	SVM	Random Forest	Naive Bayes Gausiano	Regresión Logística	Perceptrón Multicapa	K-Nearest Neighbors
Gris	64 % ± 7 %	71 % ± 5 %	52 % ± 5 %	52 % ± 8 %	59 % ± 6 %	53 % ± 5 %
Color	58 % ± 7 %	57 % ± 7 %	45 % ± 8 %	44 % ± 5 %	51 % ± 5 %	46 % ± 6 %
HoG	62 % ± 9 %	60 % ± 11 %	53 % ± 8 %	58 % ± 8 %	59 % ± 6 %	52 % ± 7 %
Bordes	56 % ± 7 %	57 % ± 5 %	50 % ± 5 %	51 % ± 6 %	54 % ± 6 %	52 % ± 5 %
ResNet152	72 % ± 5 %	63 % ± 3 %	65 % ± 7 %	63 % ± 8 %	68 % ± 7 %	62 % ± 5 %
DeCAF7	75 % ± 4 %	73 % ± 3 %	66 % ± 5 %	72 % ± 5 %	73 % ± 4 %	65 % ± 5 %

Tabla 4.4: Tabla de exactitud (accuracy).

	SVM	Random Forest	Naive Bayes Gausiano	Regresión Logística	Perceptrón Multicapa	K-Nearest Neighbors
Gris	63 % ± 7 %	70 % ± 5 %	50 % ± 5 %	53 % ± 7 %	59 % ± 6 %	55 % ± 6 %
Color	52 % ± 4 %	56 % ± 6 %	45 % ± 8 %	46 % ± 6 %	52 % ± 5 %	47 % ± 6 %
HoG	62 % ± 8 %	62 % ± 8 %	53 % ± 5 %	59 % ± 8 %	58 % ± 5 %	54 % ± 7 %
Bordes	56 % ± 7 %	57 % ± 6 %	49 % ± 6 %	51 % ± 7 %	55 % ± 8 %	54 % ± 6 %
ResNet152	71 % ± 7 %	63 % ± 6 %	63 % ± 6 %	64 % ± 7 %	70 % ± 7 %	61 % ± 7 %
DeCAF7	75 % ± 3 %	74 % ± 4 %	65 % ± 5 %	71 % ± 6 %	73 % ± 4 %	65 % ± 6 %

Tabla 4.5: Tabla de recall.

	SVM	Random Forest	Naive Bayes Gausiano	Regresión Logística	Perceptrón Multicapa	K-Nearest Neighbors
Gris	65 % ± 5 %	71 % ± 4 %	50 % ± 5 %	53 % ± 8 %	60 % ± 5 %	55 % ± 5 %
Color	54 % ± 5 %	55 % ± 5 %	45 % ± 7 %	45 % ± 5 %	53 % ± 3 %	48 % ± 5 %
HoG	60 % ± 7 %	62 % ± 8 %	54 % ± 7 %	59 % ± 8 %	58 % ± 7 %	54 % ± 8 %
Bordes	56 % ± 7 %	57 % ± 5 %	50 % ± 5 %	52 % ± 5 %	55 % ± 5 %	54 % ± 5 %
ResNet152	72 % ± 8 %	64 % ± 5 %	65 % ± 7 %	61 % ± 6 %	70 % ± 6 %	62 % ± 6 %
DeCAF7	74 % ± 5 %	74 % ± 2 %	66 % ± 4 %	71 % ± 4 %	73 % ± 5 %	66 % ± 4 %

Tabla 4.6: Tabla de F1.

	Precision	Accuracy	Recall	F1
ResNet152	74 % ± 3 %	74 % ± 3 %	74 % ± 2 %	74 % ± 2 %

Tabla 4.7: Métricas para ResNet152.

4.4. Análisis de los resultados

Podemos ver en las distintas métricas en las tablas de la sección 4.1.3, la combinación que se desempeña mejor es DeCAF7 con una máquina de vector de soporte con kernel RBF. También, podemos ver que la que se desempeñó peor es el descriptor de distribución de color con el clasificador Naive Bayes Gausiano. La diferencia entre ambas combinaciones en las métricas es cercana al 30 %, siendo DeCAF7 casi un 33 % más precisa.

El resto de las combinaciones se encuentra con valores intermedios. Cabe destacar que los descriptores clásicos se desempeñan peor en general salvo por el histograma de grises, que alcanza valores cercanos al 70 % de precisión en el mejor caso con el clasificador Random Forest. En cambio, los descriptores del estado del arte de tipo deep feature alcanzan un porcentaje superior: ResNet con una máquina de vectores de soporte con kernel RBF llega a alrededor del 72 % de precisión, y DeCAF7 con SVM con kernel RBF, que se sitúa como la mejor combinación con 74 %.

Sin embargo, al aplicar los métodos descritos anteriormente de *transfer learning* y *fine tuning*, es posible alcanzar una precisión del 74 % al modificar y re-entrenar las últimas capas de ResNet152, desempeñándose con la misma precisión que DeCAF7 con SVM.

4.4.1. Imágenes difíciles

Dentro de los resultados obtenidos, hubo imágenes que fueron más difíciles de clasificar que otras, es decir, hubo imágenes que fueron clasificadas correctamente por menos combinaciones. En particular, alrededor del 1 % de las clases no pudo ser clasificada correctamente por ninguna clasificación.

(a) (b)

Figura 4.6: Imágenes difíciles de clasificar. (a) Es un sticker incorrectamente clasificado como meme. (b) Es un meme incorrectamente clasificado como sticker.

Las imágenes en la Figura 4.6 son ejemplos de elementos que ninguna combinación pudo clasificar correctamente. Sin embargo, cabe notar que ambas tienen características de las dos clases, pero su estructura visual se asemeja más a la de la clase de la cual fue incorrectamente clasificada.

Capítulo 5

Conclusiones

Los resultados que se obtuvieron son bastante interesantes, y se puede analizar desde distintos puntos de vista.

Clasificación de Memes

Efectivamente es posible realizar la clasificación de memes, pudiendo distinguir entre un meme y un no-meme. Este es un resultado importante, pues no es un concepto muy estudiado en la literatura. Sin embargo, con un resultado del 75 % de precisión en el mejor caso, podemos ver que aún existe espacio para mejorar.

Es interesante notar que, dentro de las imágenes más difíciles de clasificar correctamente, en particular dentro de las categorías de memes y sticker, varios de los ejemplares resultan ser un tanto ambiguos incluso para personas, pues pueden cumplir varios de los criterios para ambos. En este sentido, aún falta un avance dentro del área de la comunicación para poder obtener mayor claridad en como definir cada concepto.

Descriptores clásicos y el estado del arte

Dentro de los descriptores clásicos hubo mucha varianza en los resultados obtenidos, mientras que los métodos del estado del arte fueron más consistentes. De todas formas, el mejor desempeño de un descriptor clásico (histograma de grises con random forest) obtuvo una precisión del 70 % ±5 %, resultado que no dista mucho con el mejor resultado obtenido del estado del arte (DeCAF7 con SVM) con 75 % ±4 %.

A pesar de esto, cada descriptor también puede tener resultados muy variados dependiendo del clasificador con el que se utilice. Es necesario poder trabajar cada uno de ellos con varios clasificadores, pues se puede evidenciar que no todos los descriptores pueden ser utilizados por los clasificadores con la misma efectividad.

Siguiendo la idea anterior, es difícil poder comparar directamente los descriptores entre ellos; es necesario realizar la comparación correspondiente a través de sus desempeños con los clasificadores.

Limitaciones

Al momento de clasificar, los descriptores extraen componentes de la estructura visual de la imagen. Un problema con esto es que para que una imagen sea o no meme, va a depender del contenido del texto en sí, además de su estructura visual. Este tipo de información no es algo que se pueda extraer con este tipo de métodos, y es una limitación importante pues el contenido del texto juega un rol fundamental para poder determinar si una imagen es un meme o no.

5.1. Trabajo Futuro

Imágenes difíciles de clasificar y elección de combinaciones apropiadas

Durante la realización de los experimentos surgen distintas dudas, como por ejemplo, si existen memes que no pueden ser clasificados correctamente por ninguno de los métodos estudiados. Tras realizar pruebas preliminares, efectivamente existen ejemplares que no pueden ser clasificados correctamente por ninguna combinación anterior, y corresponden a un subconjunto del total estudiado de aproximadamente el 1 % de los memes.

Esto significa que, si se pudiera elegir una combinación de descriptor y clasificador apropiada para cada meme, supongamos a través de un oráculo, se podría mejorar el proceso de clasificación hasta alcanzar una precisión de al menos 99 %, lo cuál es una cota que se podría alcanzar en teoría, y que es substancialmente superior al 75 % obtenido en los experimentos realizados.

Debido a lo anterior, como trabajo futuro se podría plantear un método que permitiera llegar a esa cota. Por ejemplo, encontrar una heurística para elegir un tipo de descriptor y clasificador dependiendo de la imagen que se le entrege, o también, a partir de un conjunto de métodos poder realizar un voto mayoritario para poder decidir a que clase correspondería.

Análisis de texto

Como se indicó previamente, el texto contenido dentro de las imágenes juegan un rol fundamental, pero no es analizado más allá de su estructura visual.

Utilizando técnicas como OCR [2] se podría extraer el texto, y este puede ser procesado con otros tipos de redes diseñadas para trabajar con lenguaje natural. De esta forma se podría obtener una mejor aproximación a lo que el texto quiere comunicar, y podría ser un paso extra de procesamiento que podría facilitar la distinción entre memes, stickers y no-memes.

Bibliografía

[1] Hamed Habibi Aghdam and Elnaz Jahani Heravi. *Guide to Convolutional Neural Networks: A Practical Application to Traffic-Sign Detection and Classification*. Springer Publishing Company, Incorporated, 1st edition, 2017. ISBN 331957549X.

[2] A. Amalia, A. Sharif, F. Haisar, D. Gunawan, and B. B. Nasution. Meme opinion categorization by using optical character recognition (ocr) and naïve bayes algorithm. In *2018 Third International Conference on Informatics and Computing (ICIC)*, pages 1–5, 2018.

[3] P. Bhattacharya. Social degeneration through social media: A study of the adverse impact of 'memes'. In *2019 Sixth HCT Information Technology Trends (ITT)*, pages 44–46, 2019.

[4] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL http://dx.doi.org/10.1023/A%3A1010933404324.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[6] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page I–647–I–655. JMLR.org, 2014.

[7] Theodoros Evgeniou and Massimiliano Pontil. *Support Vector Machines: Theory and Applications*, pages 249–257. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 978-3-540-44673-6. doi: 10.1007/3-540-44673-7_12. URL https://doi.org/10.1007/3-540-44673-7_12.

[8] S. Ferrada. Imgpedia, 2017. URL https://github.com/scferrada/imgpedia. Accessed on: Dec. 10, 2019. [Online].

[9] MW Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)—A review of applications in the atmospheric sciences. *Atmospheric Environment*, 32(14-15): 2627–2636, 1998.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[11] S. He, X. Zheng, J. Wang, Z. Chang, Y. Luo, and D. Zeng. Meme extraction and tracing in crisis events. In *2016 IEEE Conference on Intelligence and Security Informatics (ISI)*, pages 61–66, 2016.

[12] Benjamin Bustos Jesus Perez-Martin and Magdalena Saldana. Semantic search of memes on twitter. In *International Communication Association (ICA)*, 2020. URL https://jssprz.github.io/semantic-memes-docs/.

[13] Klaus Krippendorff. Computing krippendorff 's alpha-reliability, 2011. URL http://repository.upenn.edu/cgi/viewcontent.cgi?article=1043&context=asc_papers. Accessed on: Jun. 11, 2020. [Online].

[14] R. Fergus L. Fei-Fei and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. *IEEE. CVPR 2004, Workshop on Generative-Model Based Vision.*, 2004.

[15] P J (ed.) Lavrakas, 2008. Encyclopedia of survey research methods, SAGE Publications, Inc., Thousand Oaks, CA, viewed 19 July 2020, doi: 10.4135/9781412963947.

[16] Google LLC. Files, Dec 2017. URL https://files.google.com/. Accessed on: Aug. 16, 2019. [Online].

[17] B. S. Manjunath, J. . Ohm, V. V. Vasudevan, and A. Yamada. Color and texture descriptors. *IEEE Transactions on Circuits and Systems for Video Technology*, 11(6): 703–715, 2001.

[18] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[19] Antonio Mucherino, Petraq J. Papajorgji, and Panos M. Pardalos. *k-Nearest Neighbor Classification*, pages 83–106. Springer New York, New York, NY, 2009. ISBN 978-0-387-88615-2. doi: 10.1007/978-0-387-88615-2_4. URL https://doi.org/10.1007/978-0-387-88615-2_4.

[20] Camila Muñoz Villar. El meme como evolución de los medios de expresión social. Master's thesis, Universidad de Chile, Facultad de Economía y Negocios, 2014. URL http://repositorio.uchile.cl/handle/2250/129749.

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[23] G. Rowett. The strategic need to understand online memes and modern information warfare theory. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 4437–4442, 2018.

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma,

Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[25] Claude Sammut and Geoffrey I. Webb, editors. *Logistic Regression*, pages 631–631. Springer US, Boston, MA, 2010. ISBN 978-0-387-30164-8. doi: 10.1007/978-0-387-30164-8_493. URL https://doi.org/10.1007/978-0-387-30164-8_493.

[26] L. Shao, F. Zhu, and X. Li. Transfer learning for visual categorization: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 26(5):1019–1034, 2015.

[27] Sergi Verdaguer and Francisco Tarrés Ruiz. Color based image classification and description. Master's thesis, Universitat Politècnica de Catalunya, 10 2009.

[28] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf.

[29] Harry Zhang. The optimality of naive bayes. In Valerie Barr and Zdravko Markov, editors, *Proceedings of the Seventeenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2004)*. AAAI Press, 2004.

Anexo A

Figuras

A.1. Gráficos

A.1.1. Matrices de confusión para descriptor de Distribución del Color

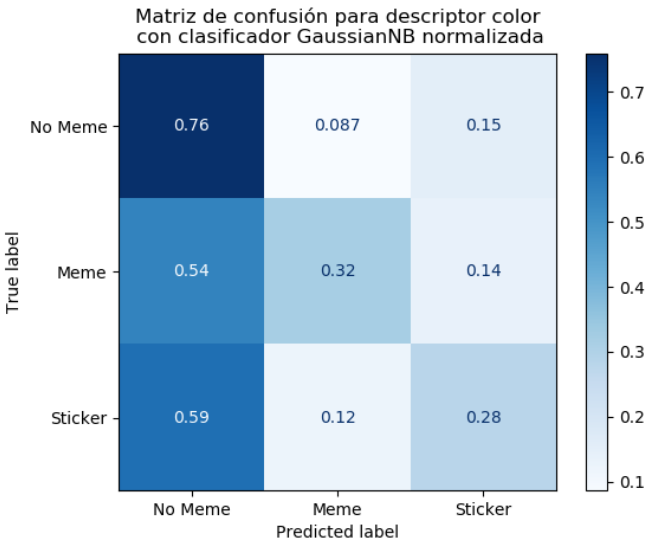


Figura A.1: Matriz de confusión para clasificador Gaussian Naive Bayes con el descriptor de Distribución del Color.

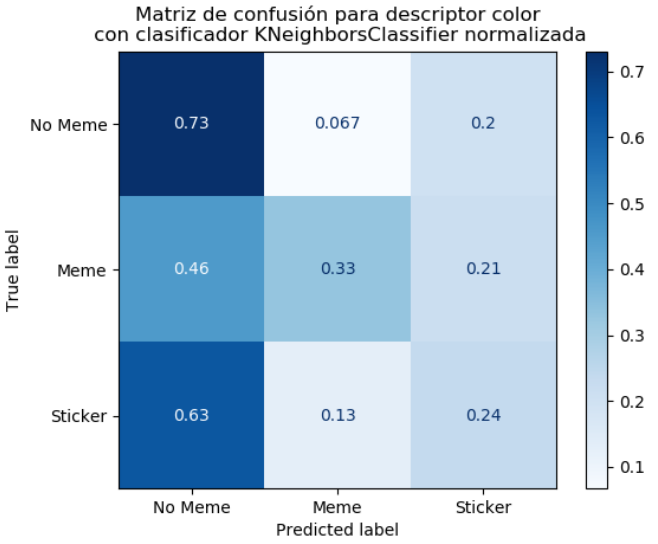


Figura A.2: Matriz de confusión para clasificador K Nearest Neighbors con el descriptor de Distribución del Color.

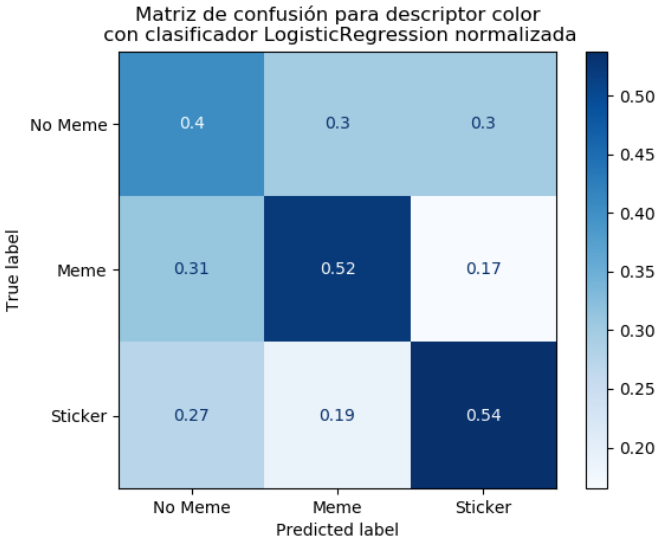


Figura A.3: Matriz de confusión para clasificador Regresión Logística con el descriptor de Distribución del Color.

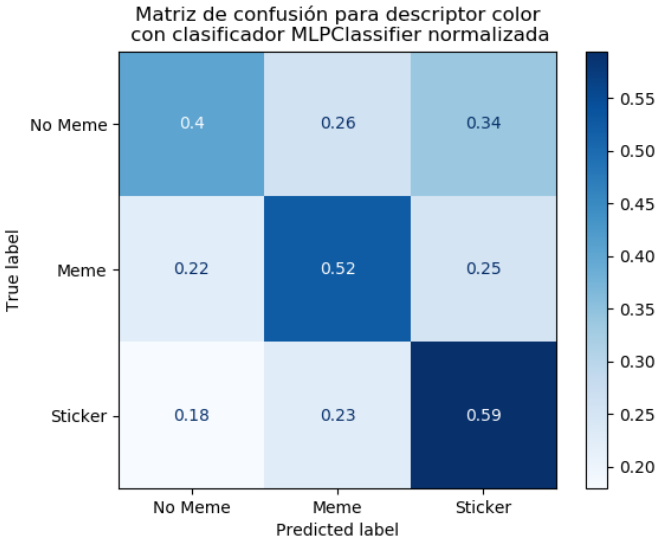


Figura A.4: Matriz de confusión para clasificador Perceptrón Multicapa con el descriptor de Distribución del Color.

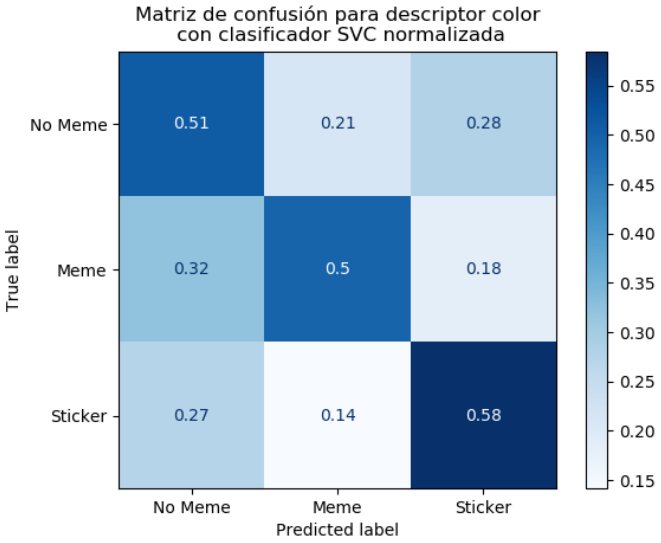


Figura A.5: Matriz de confusión para clasificador Máquina de Vectores de Soporte con el descriptor de Distribución del Color.

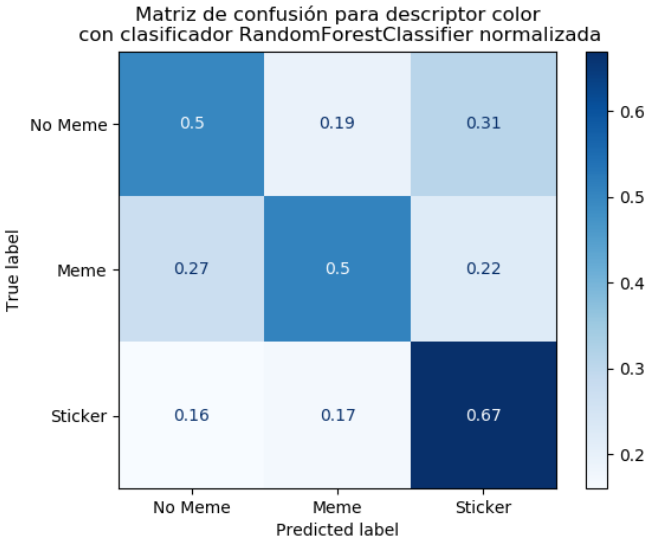


Figura A.6: Matriz de confusión para clasificador Random Forest con el descriptor de Distribución del Color.

A.1.2. Matrices de confusión para descriptor de Bordes

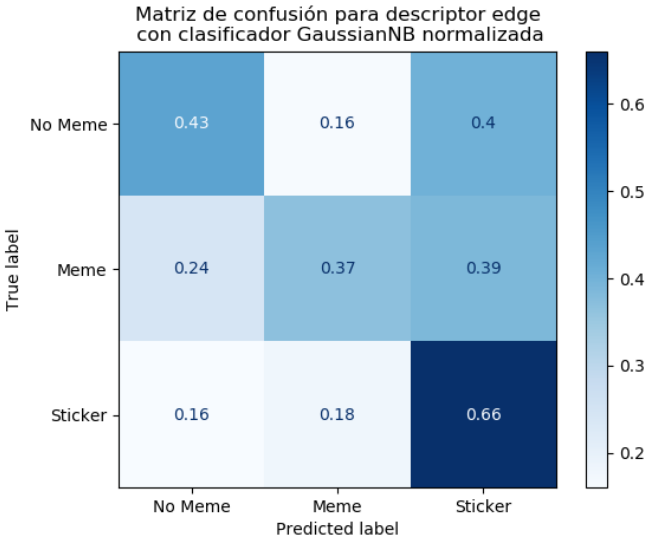


Figura A.7: Matriz de confusión para clasificador Gaussian Naive Bayes con el descriptor de Bordes.

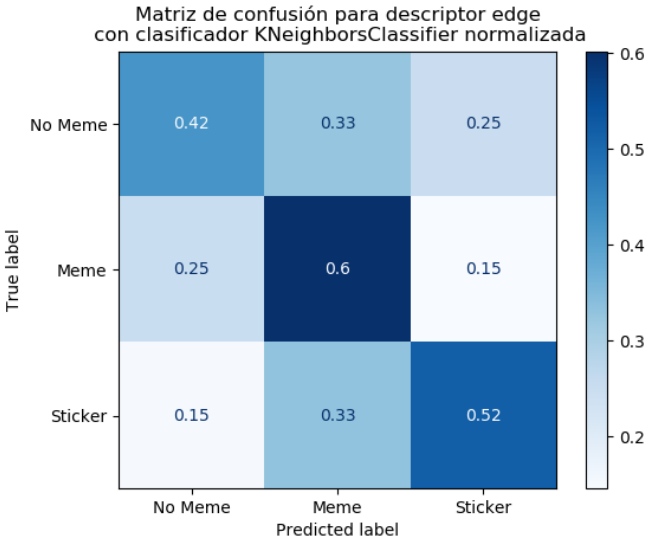


Figura A.8: Matriz de confusión para clasificador K Nearest Neighbors con el descriptor de Bordes.

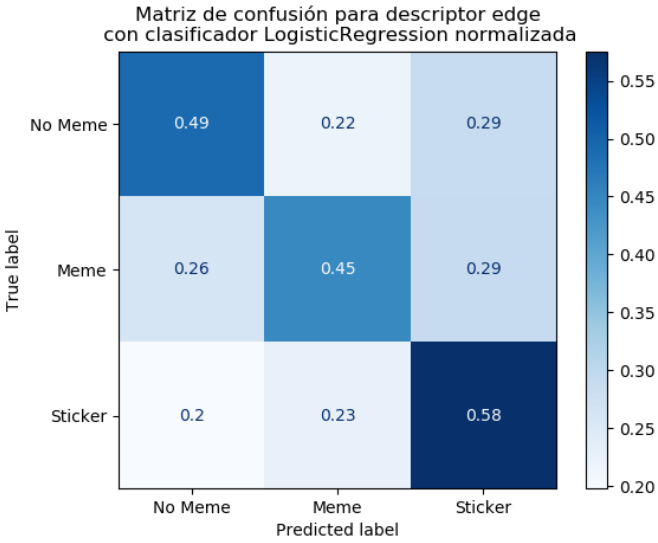


Figura A.9: Matriz de confusión para clasificador Regresión Logística con el descriptor de Bordes.

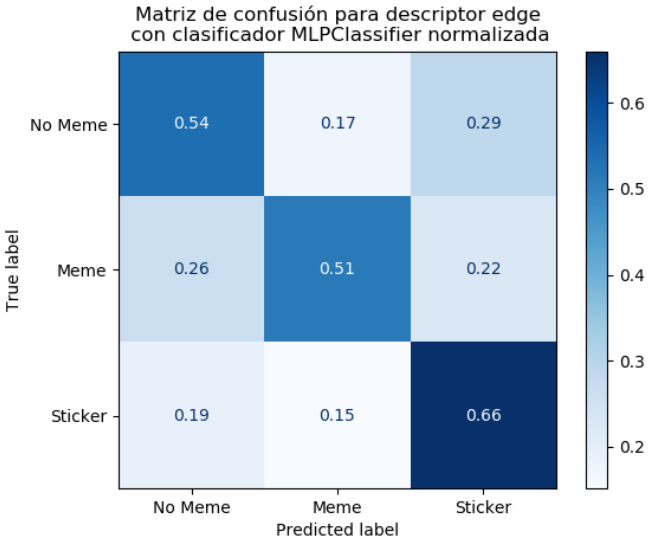


Figura A.10: Matriz de confusión para clasificador Perceptrón Multicapa con el descriptor de Bordes.

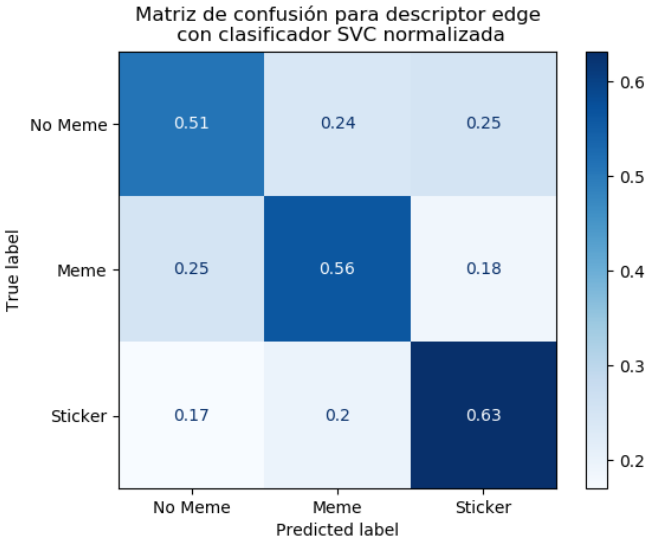


Figura A.11: Matriz de confusión para clasificador Máquina de Vectores de Soporte con el descriptor de Bordes.

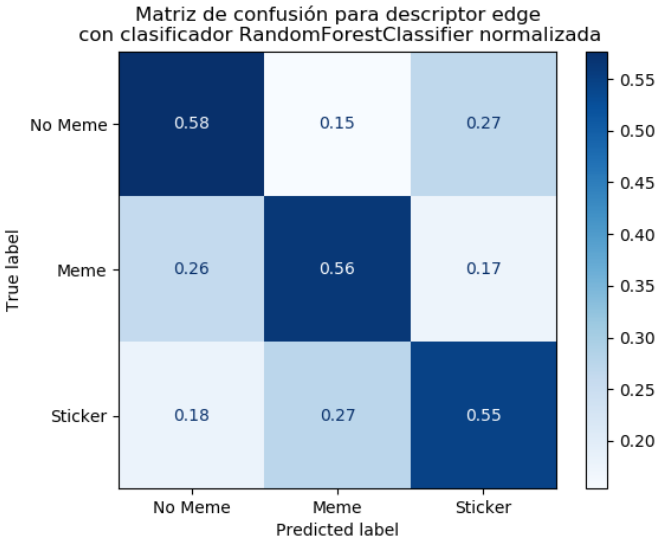


Figura A.12: Matriz de confusión para clasificador Random Forest con el descriptor de Bordes.

A.1.3. Matrices de confusión para descriptor Histogramas de Gradientes Orientados

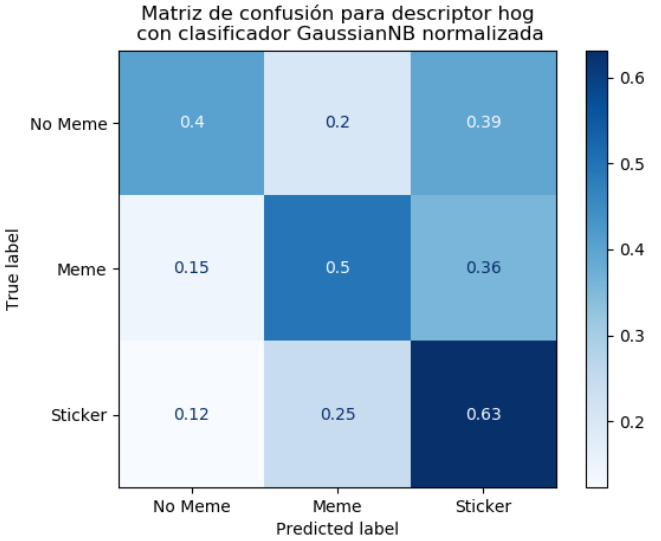


Figura A.13: Matriz de confusión para clasificador Gaussian Naive Bayes con el descriptor Histograma de Gradientes Orientados.

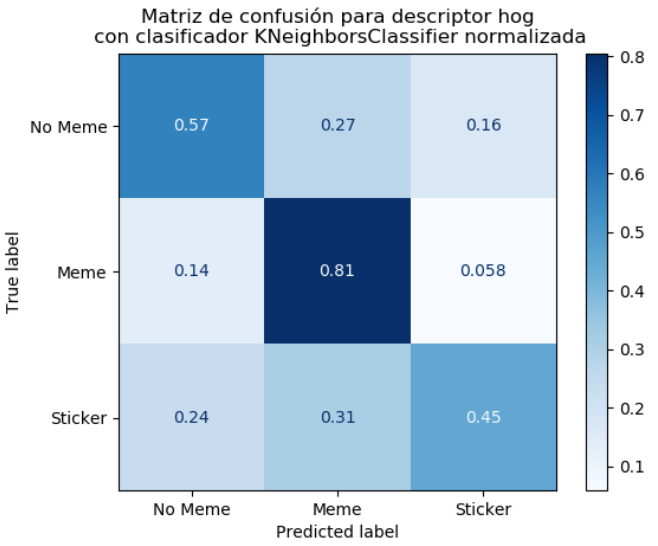


Figura A.14: Matriz de confusión para clasificador K Nearest Neighbors con el descriptor Histograma de Gradientes Orientados.

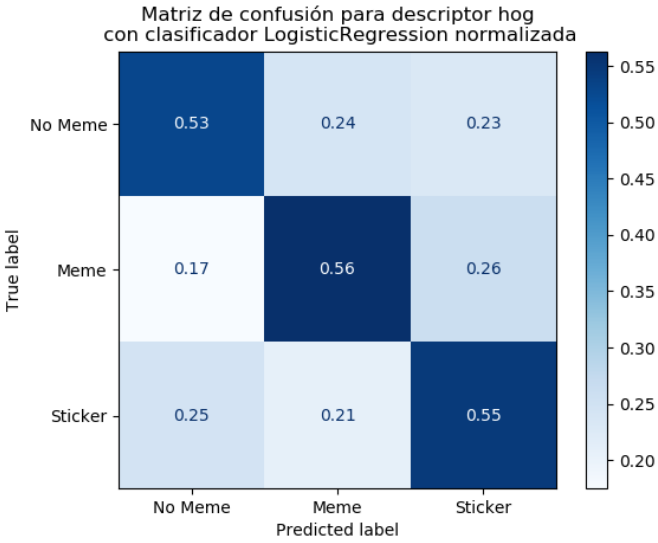


Figura A.15: Matriz de confusión para clasificador Regresión Logística con el descriptor Histograma de Gradientes Orientados.

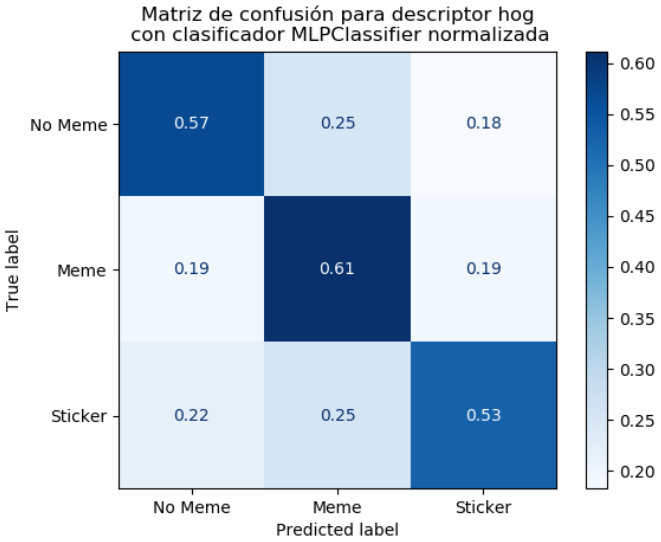


Figura A.16: Matriz de confusión para clasificador Perceptrón Multicapa con el descriptor Histograma de Gradientes Orientados.

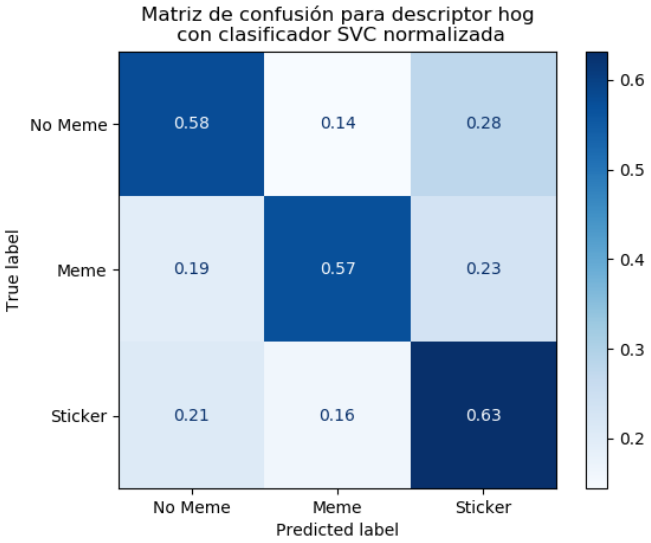


Figura A.17: Matriz de confusión para clasificador Máquina de Vectores de Soporte con el descriptor Histograma de Gradientes Orientados.

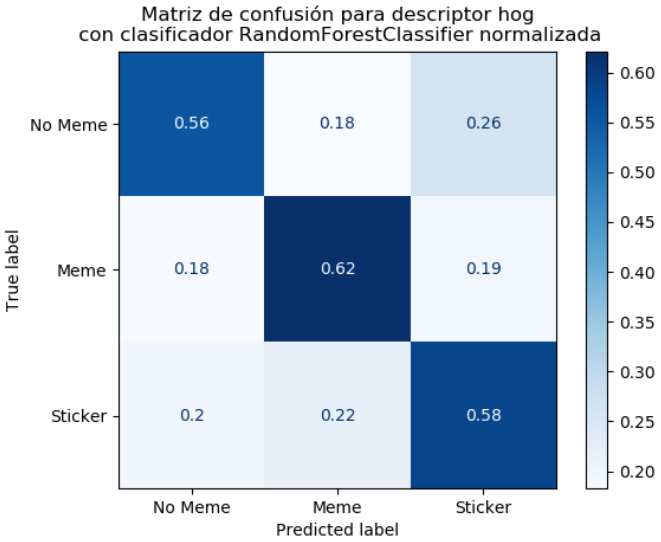


Figura A.18: Matriz de confusión para clasificador Random Forest con el descriptor Histograma de Gradientes Orientados.

A.1.4. Matrices de confusión para descriptor Histograma de Grises

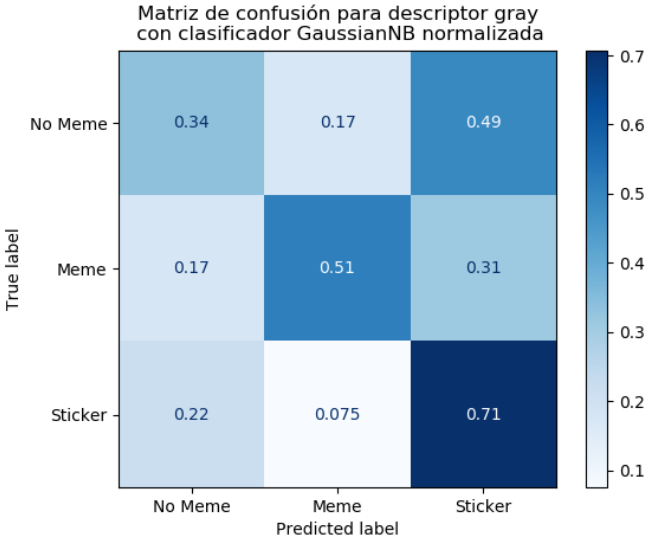


Figura A.19: Matriz de confusión para clasificador Gaussian Naive Bayes con el descriptor Histograma de Grises.

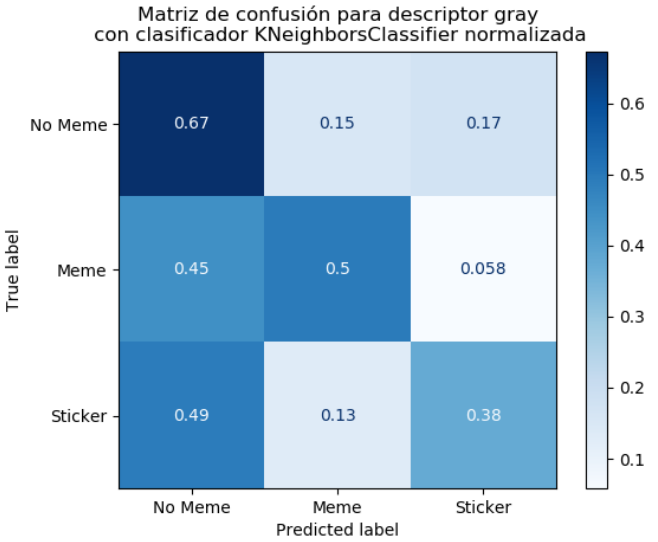


Figura A.20: Matriz de confusión para clasificador K Nearest Neighbors con el descriptor Histograma de Grises.

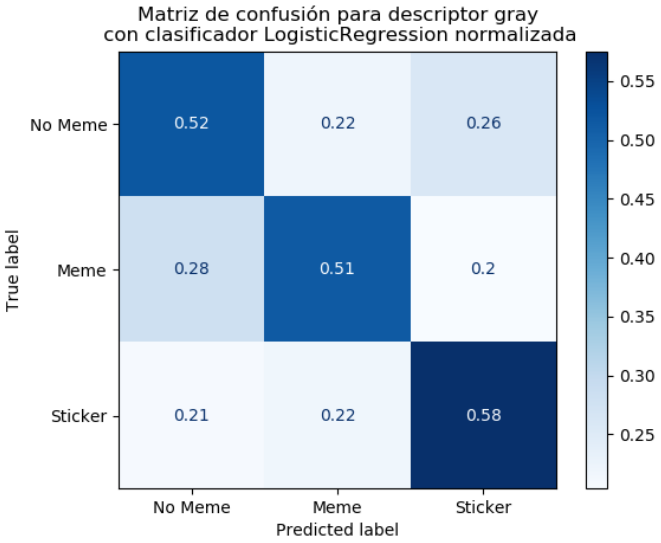


Figura A.21: Matriz de confusión para clasificador Regresión Logística con el descriptor Histograma de Grises.

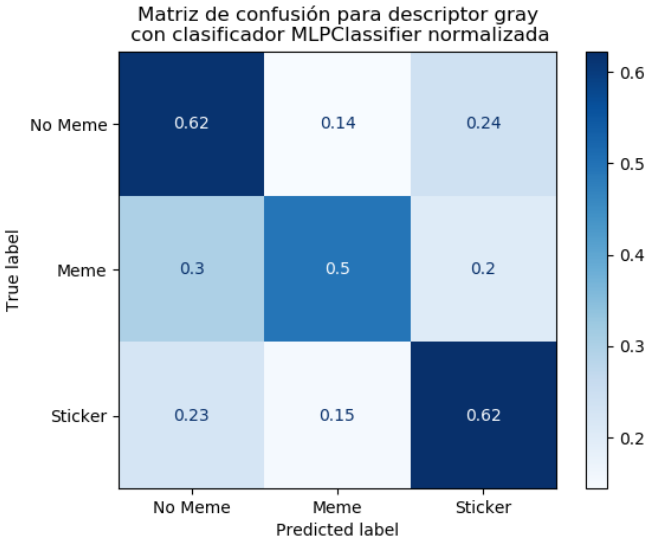


Figura A.22: Matriz de confusión para clasificador Perceptrón Multicapa con el descriptor Histograma de Grises.

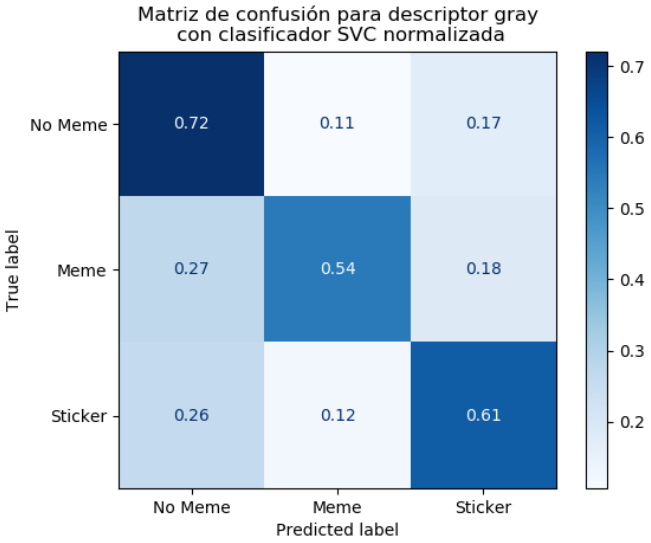


Figura A.23: Matriz de confusión para clasificador Máquina de Vectores de Soporte con el descriptor Histograma de Grises.

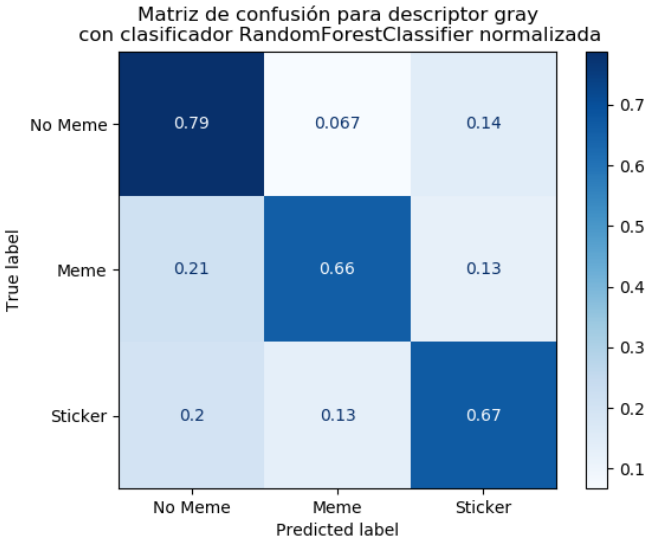


Figura A.24: Matriz de confusión para clasificador Random Forest con el descriptor Histograma de Grises.

A.1.5. Matrices de confusión para descriptor DeCAF7

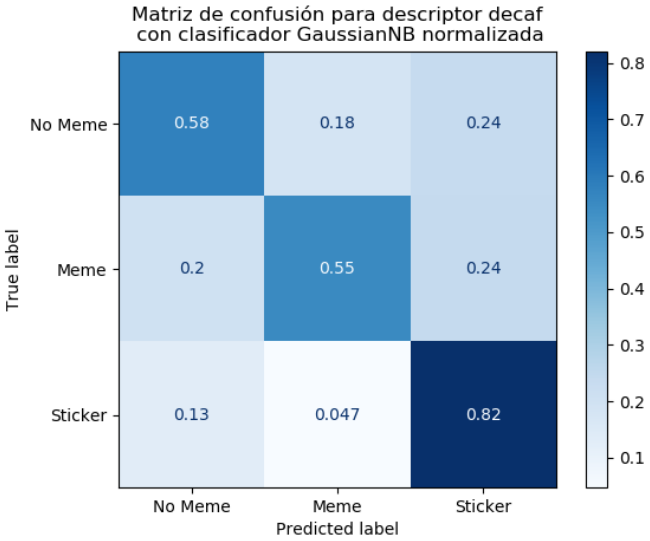


Figura A.25: Matriz de confusión para clasificador Gaussian Naive Bayes con el descriptor DeCAF7.

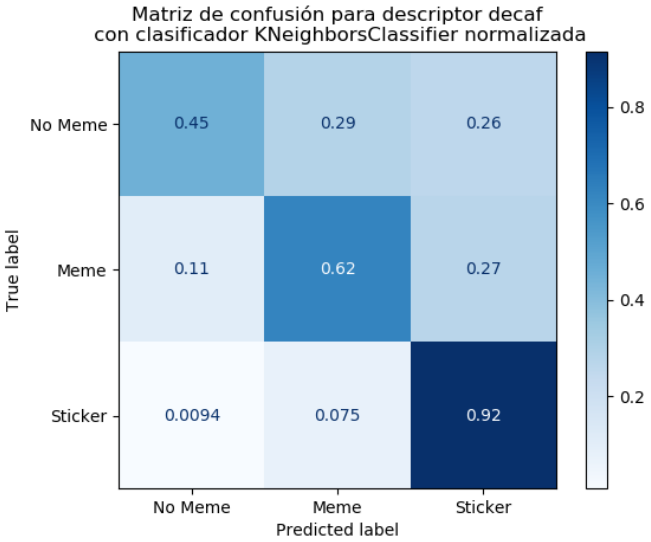


Figura A.26: Matriz de confusión para clasificador K Nearest Neighbors con el descriptor DeCAF7.

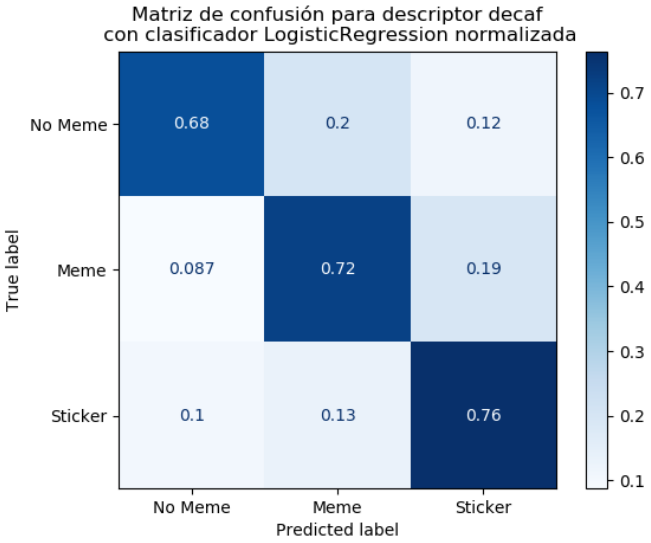


Figura A.27: Matriz de confusión para clasificador Regresión Logística con el descriptor DeCAF7.

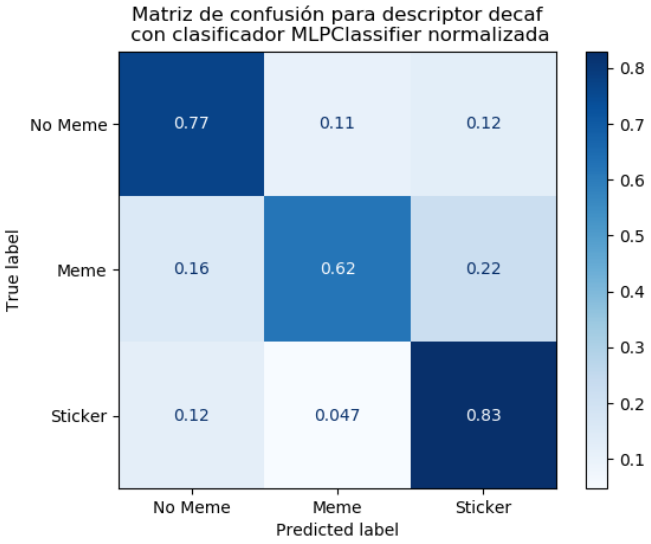


Figura A.28: Matriz de confusión para clasificador Perceptrón Multicapa con el descriptor DeCAF7.

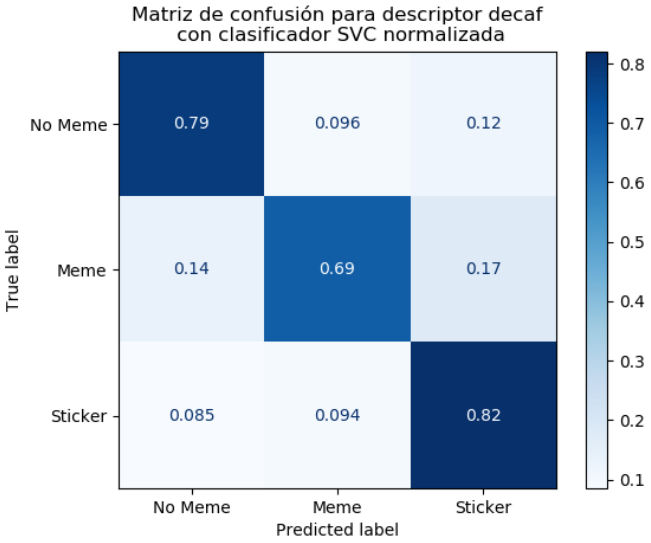


Figura A.29: Matriz de confusión para clasificador Máquina de Vectores de Soporte con el descriptor DeCAF7.

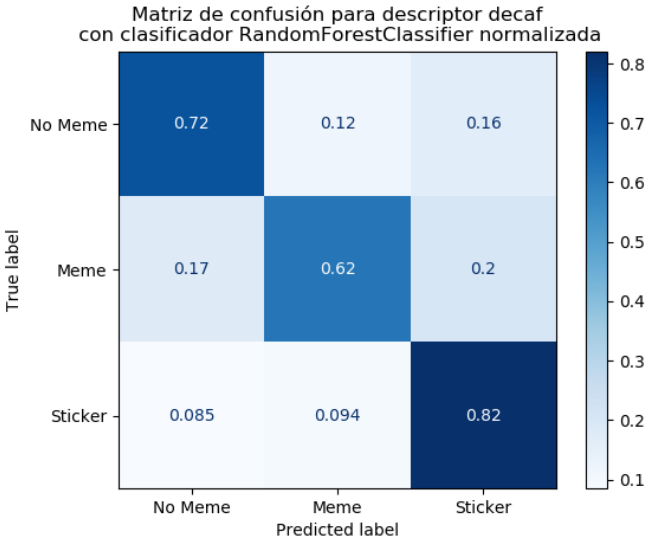


Figura A.30: Matriz de confusión para clasificador Random Forest con el descriptor DeCAF7.

A.1.6. Matrices de confusión para descriptor ResNet152

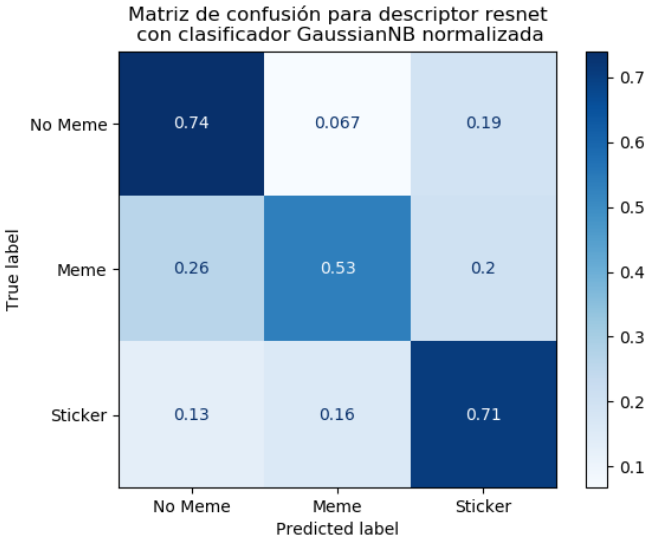


Figura A.31: Matriz de confusión para clasificador Gaussian Naive Bayes con los descriptores profundos de ResNet152.

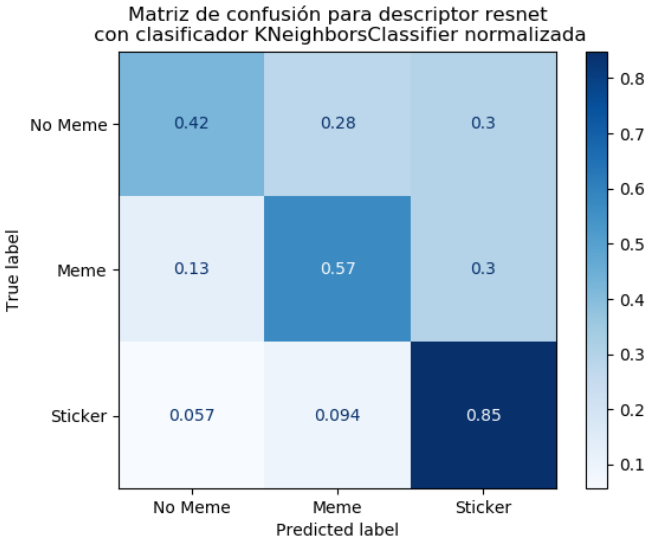


Figura A.32: Matriz de confusión para clasificador K Nearest Neighbors con los descriptores profundos de ResNet152.

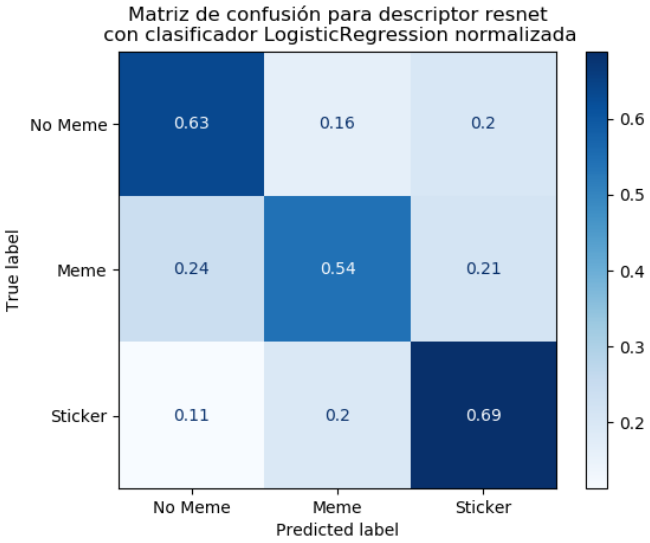


Figura A.33: Matriz de confusión para clasificador Regresión Logística con los descriptores profundos de ResNet152.

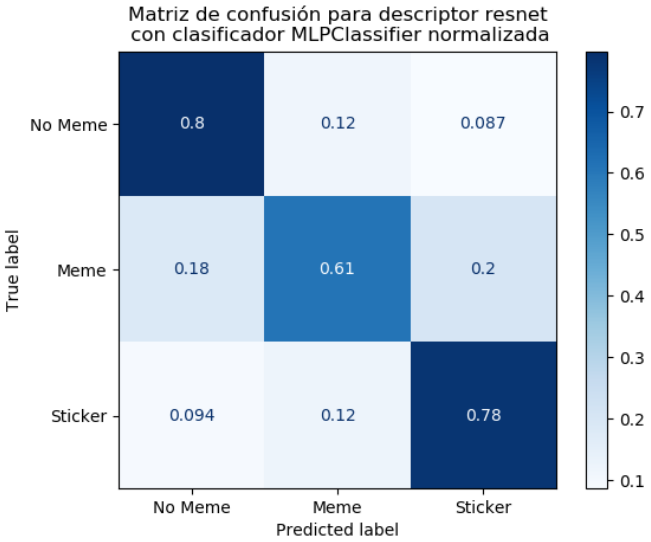


Figura A.34: Matriz de confusión para clasificador Perceptrón Multicapa con los descriptores profundos de ResNet152.

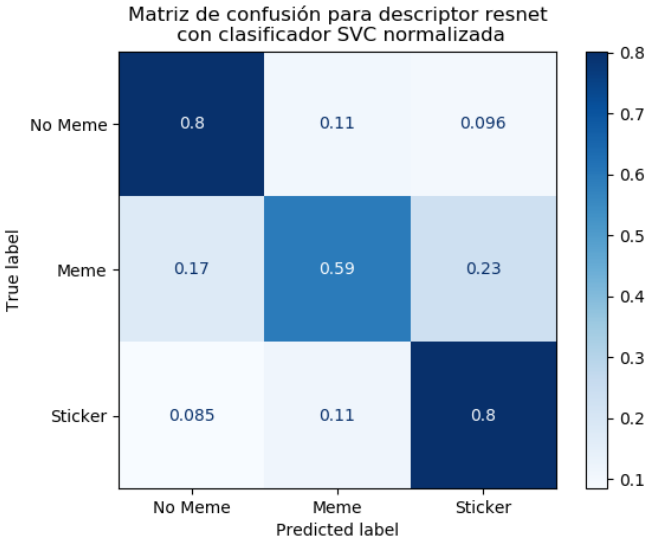


Figura A.35: Matriz de confusión para clasificador Máquina de Vectores de Soporte con los descriptores profundos de ResNet152.

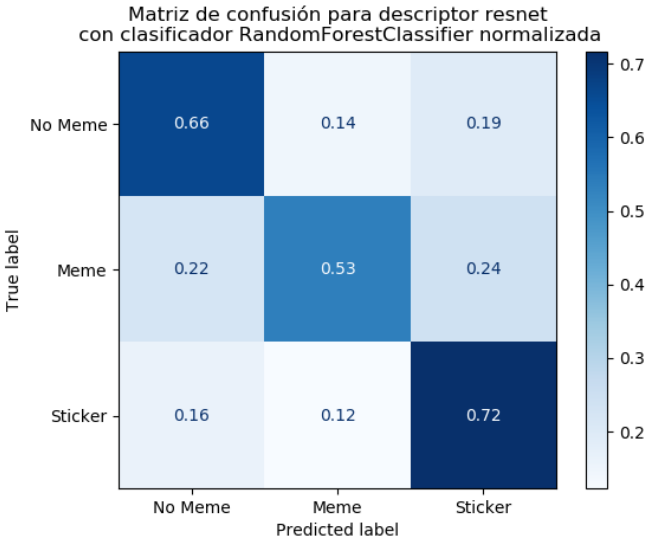


Figura A.36: Matriz de confusión para clasificador Random Forest con los descriptores profundos de ResNet152.