

Desarrollo de un sistema de reconocimiento de emociones faciales en tiempo real

Héctor Ribes Gil

Resumen— Una organización del trastorno nutricional ha puesto en práctica un nuevo tratamiento para sus pacientes. Éste está directamente relacionado con la emoción del paciente, cuya emoción se clasifica oralmente. Llegados a ese punto aparece el reto de automatizar esta tarea, desarrollando un sistema de reconocimiento de emociones faciales en tiempo real caracterizado por un detector de rostros, un detector de Key Points y una red neuronal convolucional utilizando “DeepLearning” para extraer las características del rostro detectado.

Palabras clave— red neuronal, red neuronal convolucional, detector rostro, detector keypoints, landmarks, deep learning, cnn

Abstract— An nutritional disorder organization has implemented a new treatment for its patients. This is directly related to the mood of the patient, whose emotion is classified orally. At this point, the challenge is to automate this task by developing a real-time facial recognition system, characterized by a face detector, a Keypoint detector, and a convoluted neural network using “ DeepLearning ” to extract characteristics of the face detected.

Keywords— neural network, convoluted neural network, face detector, keypoints detector, landmarks, deep learning, cnn

1 INTRODUCCIÓN

CON la evolución de la tecnología han surgido muchos caminos de desarrollo e investigación. La importancia de este campo de desarrollo puede ser interesante para sus posibles usos, dando infinidad de caminos más a abrir en el campo de la biología, bioinformática y inteligencia artificial [1].

Este documento trata de detallar el desarrollo de un sistema para reconocer las emociones faciales a partir de una fuente visual, lo que implica a priori un reconocimiento facial, detectar los puntos de interés de éste reconocimiento y un análisis sobre un conjunto de imágenes catalogadas por expertos o instituciones con anotaciones.

El sistema tiene que clasificar entre un conjunto de estados de ánimo: felicidad, tristeza, vergüenza, angustia, relajado, miedo, culpable y frustración. Después éste tiene que pintar un avatar relacionado con la emoción detectada en la cara del usuario. El proyecto será destinado a una Orga-

nización del trastorno nutricional, donde han desarrollado un método que se basa en la emoción del paciente. Se trata de automatizar un procedimiento que actualmente se realiza manualmente.

La importancia de que funcione en tiempo real y su correcto funcionamiento se convierte en uno de los objetivos principales, como el correcto funcionamiento de los módulos, ya que sin ellos no sería de utilidad.

El trabajo está estructurado en los siguientes apartados:

- La sección 1.1 presenta cómo ha evolucionado el estado del arte, desde la parte comercial hasta la parte de investigación, así como algunos ejemplos de sus posibles usos.
- La sección 2 define los objetivos principales y objetivos específicos de este sistema.
- La sección 3 presenta la metodología utilizada en este proyecto.
- La sección 4 presenta el análisis de los resultados de la metodología.
- La sección 5 presenta los experimentos realizados y resultados obtenidos.

• E-mail de contacto: hector.ribes@e-campus.uab.cat
 • Mención realizada: Computación
 • Trabajo tutorizado por: Katerine Diaz (CVC)
 • Curso 2016/17

TABLA 1: INFORMACIÓN DE LOS DATASETS DEL PROYECTO Y SU CATALOGADO

Dataset	#Et.	#Im.	Dimensión	Feliz (1)	Enfado (2)	Triste (3)	Angustia (4)	Relaj. (5)	Miedo (6)	Descartadas
jaffe [4]	7	154	[256,256]	Feliz (31)	Enfado (30)	Triste (31)		Neutro (30)	Miedo (32)	Asco Sorprendido
CK [5]	7	281	[490,640]	Feliz (25)	Enfado (45)	Triste (69)	Desprecio (83)		Miedo (59)	Asco Sorprendido
10K [5]	7	2219	[256,185]	Feliz (1751)	Enfado (53)	Triste (186)		Neutro (216)	Miedo (13)	Asco Sorprendido
yale [6]	10	45	[243,320]	Feliz (15)		Triste (15)		Normal (15)		Con/Sin gafas Iluminación izq/der/centro Dormido Guiño

- La sección 6 presenta las conclusiones del proyecto así como tareas futuras.

1.1. Evolución del arte

Actualmente hay varias empresas especializadas en reconocimiento de emociones faciales como Microsoft[2], Eyuris... [3] pero presentan costes elevados y un tiempo de ejecución alto.

Estos sistemas utilizan un API Web Rest para consultar la emoción ejecutando en su servidor una red neuronal que devuelve los porcentajes de cada emoción. Estas compañías se lucran de la información que proporciona las emociones, donde en un posible presente el usuario pueda estar viendo un spoiler o publicidad y se analice su emoción para el mejor manejo de las cookies proporcionando mucho valor de aquí a unos años.

En el campo de la investigación hay muchísimos proyectos en marcha[7] en muchos ámbitos de este arte, siendo de apoyo para coches autónomos (como reconocer a un policía y sus gestos, [20]), en la robótica actuando como un individuo humano en postura y gestos (siendo de ayuda para desarrollos de implantes de miembros [14]), o para la clasificación de personas en género, edad y parentescos [15]. Esta última clasificación puede ser de utilidad para el control de accesos o la búsqueda de imágenes en bases de datos.

Podríamos decir que la investigación de éste arte permite infinidad de posibles usos con las combinaciones de estas técnicas.

2 OBJETIVOS

Los objetivos del proyecto son:

- Localizar datasets adecuados para detectar emociones faciales. Sin ellos no podríamos alcanzar la correcta clasificación de las emociones reales. Para que sean válidos, las emociones de los individuos en principio las tendría que catalogar y validar un experto en esa ciencia.
- Implementar un sistema de detección de rostros que funcione en tiempo real. Si el sistema no es capaz de actuar en tiempo real, no será una aplicación de utilidad para la comunidad porque es una información ne-

cesariamente instantánea para el módulo de catalogado de emociones.

- Desarrollar un sistema que reconozca emociones faciales.

Siendo un requerimiento funcional para la institución a la que se destina.

- Desarrollar un sistema de avatar sobre el rostro detectado. Se trata de pintar en el rostro del usuario un avatar relacionado con la emoción detectada. Éste punto puede ser también uno de los más importantes, ya que los pacientes que van a usar el sistema inicialmente no les gusta ver su reflejo.

3 METODOLOGÍA

Lo primero de todo y para poder empezar a hacer las pruebas y desarrollar el proyecto es encontrar datos con los que aporten información de las emociones a detectar.

La figura 1 presenta de forma esquemática las fases principales del proyecto.

Para empezar a detectar la emoción primero se tiene que detectar el rostro. Una vez detectada la posición del rostro, se procederá a detectar los KeyPoints, seguidamente de clasificar la emoción y pintar un avatar encima del rostro.

3.1. Datasets

Los datasets recopilados para este proyecto así como sus etiquetas y número de muestras, dimensiones y otros datos de interés se presentan en la tabla 1.

Las etiquetas de cada proyecto se han interpretado para poder unir las etiquetas de los datasets y tener las etiquetas del proyecto, catalogadas por números.

3.2. Detector de rostro

Analizando posibles metodologías se han implementado dos algoritmos diferentes para este detector con diferentes Framework (el propio sistema Android, y OpenCV).

Android posee una librería de visión por computador donde incluye un detector de rostros que se puede utilizar con diferentes configuraciones (clase FaceDetector [8]). Ésta se caracteriza por su alta tasa de acierto.

Fig. 1: Fases principales del proyecto

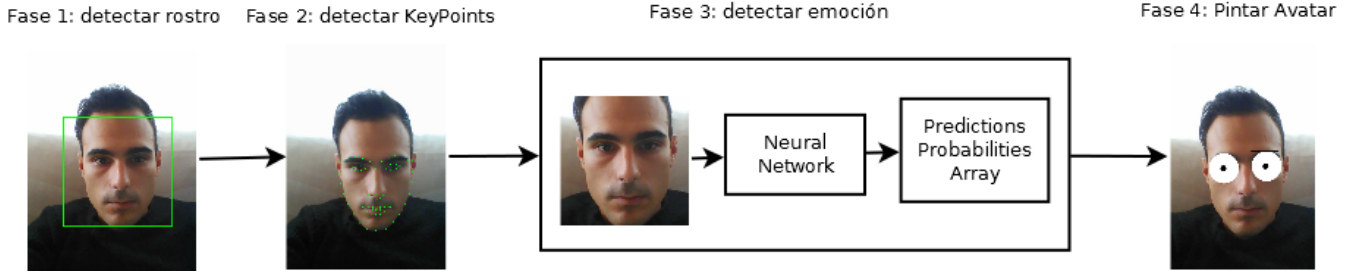
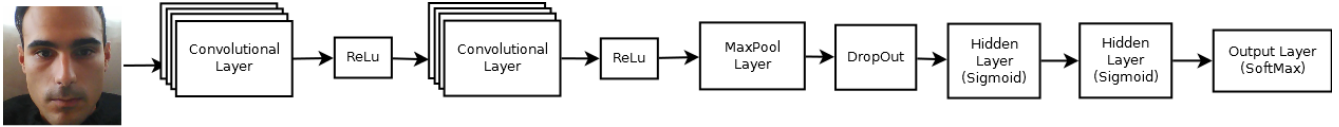


Fig. 2: CNN model 1



Con OpenCV se ha implementado un clasificador haar cascade basado en Viola-Jones que se puede modificar o aprender nuevas muestras [9].

3.3. Detector de KeyPoints

Se ha evaluado el detector de KeyPoints de Android (la misma clase FaceDetector[8] del módulo anterior) y el de OpenCV[19] utilizando la librería DLib[10] que presentan 5 y 68 KeyPoints respectivamente.

Se ha incorporado a Android el detector de rostros y de KeyPoints de OpenCV realizados en C++ con Android NDK[16], haciendo llamadas nativas desde Java al archivo C++ de nuestro proyecto.

3.4. Detector de emociones

Para este módulo se han implementado diferentes redes neuronales con arquitecturas y capas diferentes, junto a una máquina de vectores por soporte (SVM) [11].

En las redes neuronales, cada neurona tiene un peso y cada capa dispone de neuronas. Cada capa se dota de una función de activación, en la que los pesos serán multiplicados por esta función. También disponen de un bias con dimensión de la salida de la capa, al que es sumado el resultado (Ecuación 1).

Los diferentes tipos de funciones en las capas que se han utilizado son:

- Convolucionales para dotar de más información a una misma neurona.
- Relu para detectar neuronas que no se han activado y simplificar la convolución.
- MaxPool para simplificar aún más cogiendo el máximo por área definida (*altura, anchura*).
- Función sigmoid (Ecuación 2) para separar o simplificar uno o varios conjuntos.
- Función softmax (Ecuación 3) para calcular probabilidad de clasificación de etiquetas (perfecta para detectar el grado de probabilidad de emoción).

Para actualizar los pesos de cada neurona se tiene que dar una función de coste, en este trabajo se ha implementado para una red neuronal convolucional con función de coste sencilla (media de predicciones erróneas) y otra sencilla con backpropagation en el que descarta las neuronas culpables del error (función de coste en la ecuación 4).

$$\sigma'(x^i) = \sigma(w^i x^i + b^i) \quad (1)$$

$$\text{sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (2)$$

$$\text{softmax}(x)_j = \frac{e^{x_j}}{\sum_{k=1}^K e^{x_k}} \quad (3)$$

$$\delta_j^K = \frac{\delta C}{\delta w_j^K} \sigma'(x^i) \quad (4)$$

x : valor entrante a la neurona

j : posición de la neurona dentro de la capa.

K : número máximo de elementos de la capa.

C : función de coste.

w : peso correspondiente a la neurona.

b : bias de la neurona.

Un problema muy frecuente en las redes neuronales es el Overfitting [22] (sobre entreno). Se produce cuando una red neuronal ha adaptado los pesos demasiado a sus muestras y por lo tanto es muy probable que dadas otras muestras no incluidas en el dataset no nos proporcione la correcta predicción.

Para evitarlo, hemos implementado una parada preventiva, que consiste en dejar de entrenar una vez encontrado el mínimo global. Se ha implementado con un contador inicializado por un valor grande inicializado empíricamente (como 10000). Cada vez que encuentra una mejor validación se incrementa por una constante pequeña definida empíricamente (como por ejemplo 10). Cuando ésta es inferior al número de iteraciones multiplicado por el número de lotes saldremos del bucle evitando sobre-entrenar, intentando parar en el mínimo global.

Fig. 3: Red neuronal convolucional 2

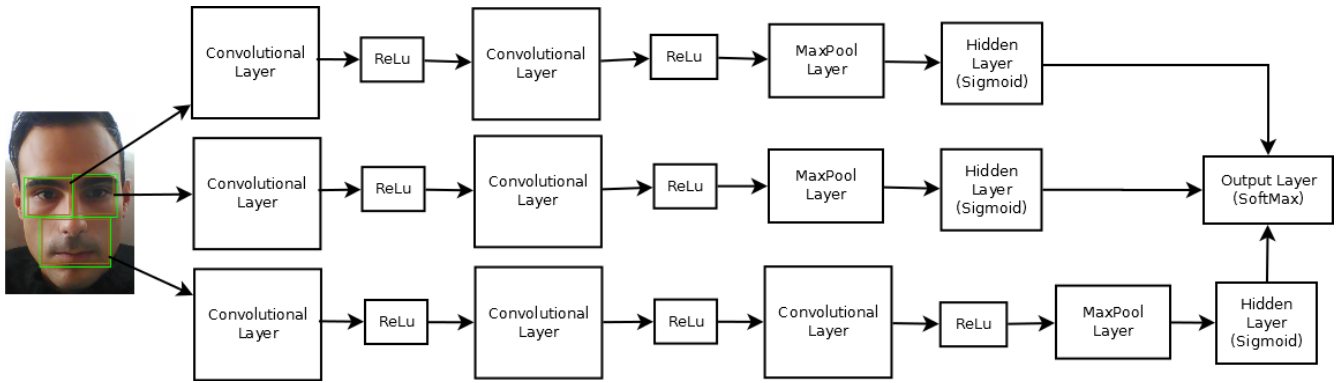


TABLA 2: TIEMPOS DE EJECUCIÓN EN MS. Y TASA DE ACIERTO DE LAS IMPLEMENTACIONES DEL MÓDULO DE DETECCIÓN DE ROSTRO

Dataset	KeyPoint	FaceDetector Android				Viola-Jones (OpenCV)	
		Accurate		Fast		(Without Landmarks)	
		Time Average	% wins	Time Average	% wins	Time Average	% wins
Yalefaces	yes	265,41	100,00 %	139,8	100,00 %	20,49	97,33 %
	no	106,65		76,53			
10K US Faces	yes	236	99,81 %	122,66	99,79 %	13,86	85,56 %
	no	83,46		56,77			
CK	yes	360,35	100,00 %	231,64	100,00 %	23,32	98,38 %
	no	157,09		125,61			
Jaffe	yes	287,38	100,00 %	139,09	100,00 %	19,75	100 %
	no	117,42		70,68			

Para poder tener mejores resultados en las redes neuronales se ha utilizado DeepLearning, un conjunto de algoritmos basados en extraer características, formados por capas convolucionales para agrupar información y MaxPool para simplificar las muestras.

A continuación se presenta el clasificador entrenado en este proyecto:

- Máquina vector soporte (SVM), un clasificador muy sencillo caracterizado por encontrar entre dos conjuntos una función lineal o no lineal que los separa. Esta restricción obliga a tener más de un clasificador para más de dos salidas. Entrena con la salida del detector de rostros (normalizado de 0 a 1).

A continuación se detallan las redes neuronales entrenadas en este proyecto:

- Red neuronal sencilla caracterizada por la manera de actualizar sus pesos (backpropagation). A la hora de entrenar, busca las neuronas que son culpables de la predicción errónea y actualiza ese error para futuros posibles errores. La entrada es la parte de imagen detectada por el detector de rostros normalizado de 0 a 1, en algunos datasets ha obtenido buenos resultados pero también ha sido descartado como clasificador de este proyecto.
- Red neuronal de tipo red neuronal convolucional (Figura 2) desarrollada con Theano[12] y Lassagne[13]. Como entrada tiene la parte de imagen detectada por

el detector de rostros, normalizada de 0 a 1 y tratada con PCA [23].

4 ANÁLISI DE RESULTADOS

Analizando los resultados aportados sobre el detector de rostros 2, podemos concluir que la clase FaceDetector de Android es mucho más eficiente, pero más costoso computacionalmente. La ventaja que ofrece OpenCV frente a Android es que aprovecha la GPU, pudiendo ofrecernos mejores tiempos de ejecución. Una de las características a destacar sobre los tiempos de ejecución es la dimensión del dataset1.

Comparando los detector de KeyPoints se ha decidido trabajar con DLib[10] cantidad de información aportada (68 KeyPoints, respecto a los 5 de Android). Otro aspecto importante a destacar que el tiempo de ejecución no se ve incrementado al de Android (incrementando entre 1 y 10 ms) debido a OpenCV[19].

Analizando los resultados de los detectores de emociones encontramos que SVM no ha llegado a una función lo suficiente compleja para clasificar correctamente (Apéndice A.1) y el hecho de tener varios clasificadores hace tener un coste computacional alto, por lo que se descarta del proyecto.

Respecto a los resultados de la red neuronal simple con backpropagation (Apéndice A.2) no ha llegado tampoco a una función lo suficiente compleja[22], por lo que se descarta del proyecto.

Con los resultados obtenidos sobre la red neuronal convolucional (Apéndice A.3) encontramos que sí ha dado con una estructura correcta con 20 y 40 filtros respectivamente en cada convolución, lo que hace tener un parámetro de entrada computacionalmente más costoso y con capas ocultas (Hidden Layer) con función de activación sigmoid y softmax respectivamente para clasificar. En la tabla 23 no ha producido buenos resultados porque se ha producido overfitting [22].

5 EXPERIMENTOS Y RESULTADOS

TABLA 3: RESUMEN DE LOS CLASIFICADORES DE EMOCIONES

Con los resultados obtenidos (Tabla 3, se ha llegado a la conclusión de que se podía obtener mejores resultados y con menos capas, es decir un clasificador más sencillo.

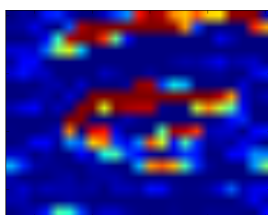
Dataset	Jaffe	Cohn-Kanade	10K US	Yale
SVM	33 %	16 %	79 %	50 %
NN	20 %	47 %	79 %	33 %
CNN	63 %	71 %	10 %	66 %
CNN custom	73 %	71 %	75 %	78 %

Simplificando los datos a entrenar separando el rostro los ojos y la boca obtenidos a partir de los KeyPoints para entrenar por separado (Figura 3), y normalizados de 0 a 1) se han obtenido resultados similares a la misma red convolucional con muchos filtros.

Otro de los experimentos realizados es entrenar con capas convolucionales preentrenadas y encontrar la arquitectura con menos elementos para conseguir un clasificador mucho más rápido (con menos operaciones).

Haciendo una comparativa de los datasets y dado el más grande (10K US Faces, tabla 1), se ha entrenado con más de 10 mil epochs, exportando las convoluciones a los otros datasets, pero con los mismos resultados que inicializando los pesos aleatoriamente. Seguidamente realizar el mismo procedimiento con el dataset más balanceado (Jaffe, tabla 1), que muestra 30-31 muestras por emoción. Se entrenó expresamente con todas las muestras para conseguir en las primeras convoluciones un extractor de características, teniendo como resultado de las convoluciones pre-entrenadas en la figura 4, 5 y 6, donde el color rojo es donde presenta más información.

Fig. 4: Resultado de la convolución entrenada en el ojo derecho



Con la conclusión de que una red neuronal convolucional necesita muchas iteraciones para llegar a un buen resultado, se consigue con un ojo buenos resultados en la iteración 300 en el dataset 10K US Faces (Tabla 4).

Fig. 5: Resultado de la convolución entrenada en el ojo izquierdo

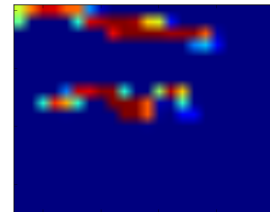


Fig. 6: Resultado de la convolución entrenada en la boca

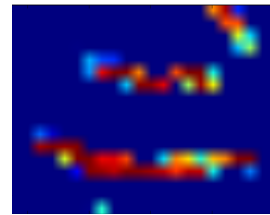


TABLA 4: TABLA DE CONFUSIÓN DEL DATASET 10K ENTRENANDO EL OJO IZQUIERDO A LA ITERACIÓN 300 CON CNN 2

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	254	4	25	30	1	.81
	2	3	2	1	0	1	.29
	3	30	2	3	5	0	.07
	5	60	2	8	8	1	0.1
	6	0	0	0	0	0	0.0
Accuracy		.73	.2	.09	0.18	0.0	61 %

Los resultados sobre las tres redes neuronales juntas eran peores que cada una por separada, donde entrenando cada red neuronal independientemente como extractor de características y entrenar la salida independientemente encontramos resultados como el de la tabla 5, 7, 6 y 8. Hasta aquí tenemos el mismo resultado que la red neuronal convolucional con 20 y 40 filtros, obteniendo mejores resultados en el dataset 10K (Tabla7) y muchísimo más sencilla y por tanto mucho más eficiente computacionalmente.

TABLA 5: TABLA DE CONFUSIÓN DEL DATASET JAFFE CNN2 CON 50 NEURONAS

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	4	0	0	1	0	.8
	2	0	5	1	0	0	.83
	3	0	0	4	0	0	1.0
	5	0	0	1	3	0	.75
	6	2	1	0	2	6	.55
Accuracy		.67	.83	.67	.5	1.0	73 %

5.1. Avatar

Se han evaluado y implementado diferentes técnicas como triangulación Delanuy[18] (consiste en triangular un conjunto de puntos donde cada triángulo no contiene ningún otro punto) para después poder pintar triángulo a triángulo

TABLA 6: TABLA DE CONFUSIÓN DEL DATASET CK CNN 2 CON 100 NEURONAS

		Esperado						Acc.
		1	2	3	4	6		
Predecido	1	14	0	0	0	0	1.0	
	2	0	6	2	2	1	.55	
	3	0	3	3	2	0	.37	
	5	0	0	1	0	0	0.0	
	6	0	0	1	0	4	.8	
Accuracy		1.0	.67	.42	0.0	.8	71 %	

TABLA 7: TABLA DE CONFUSIÓN DEL DATASET 10K CNN 2 CON 100 NEURONAS

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	307	8	27	28	2	.83
	2	0	0	0	0	0	0.0
	3	10	1	2	5	1	.11
	5	20	1	8	10	0	.25
	6	0	0	0	0	0	0.0
Accuracy		.91	0	.05	.23	0.0	73 %

TABLA 8: DATASET YALEFACES CNN 2 CON 50 NEURONAS

		Esperado			Acc.
		1	3	5	
Predecido	1	3	2	0	.6
	3	0	1	0	1.0
	5	0	0	3	1.0
Accuracy		1.0	.33	10	78 %

TABLA 9: TABLA DE CONFUSIÓN DE TODOS LOS DATASETS CNN 2

		Esperado						Acc.
		1	2	3	4	5	6	
Pred.	1	354	10	30	1	43	2	.81
	2	1	2	1	0	1	0	.4
	3	6	7	11	1	1	5	.35
	4	0	0	0	0	0	0	0.0
	5	9	5	2	1	8	3	.29
	6	0	1	0	0	0	4	.8
Accuracy		.95	.08	.25	0.0	.15	.28	75 %

partes de otra imagen (la figura 7 muestra la implementación eliminando los triángulos interiores de los ojos y boca), pero dada la complejidad computacional y que el tiempo es un requisito se ha decidido montar gráficamente el avatar a partir de los KeyPoints encima del rostro, haciendo éste módulo instantáneo, aportando información de avatar en las cejas (Figura 8).

6 CONCLUSIONES Y TRABAJO FUTURO

Según la tabla 2 podemos confirmar que la clase FaceDetector de Android es prácticamente 100 % fiable con cualquier configuración, pero dado que el tiempo de ejecución es uno de los objetivos principales de este proyecto es más

Fig. 7: Implementación de la triangulación delaunay

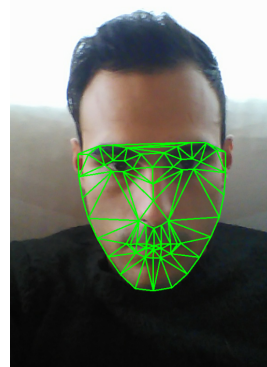


Fig. 8: Pintado encima del rostro según KeyPoints



apropiado OpenCV.

Otra de las ventajas que aporta OpenCV es la libertad de código, pudiendo optimizar el código a nuestras necesidades.

TABLA 10: COMPARATIVA DE RESULTADOS CON OTROS PROYECTOS DE LOS DATASETS MÁS FAMOSOS DEL PROYECTO.

Autor	Jaffe	Cohn-Kanade
Xianlin Peng, Zhaoqiang Xia, Lei Li, Xiaoyi Feng CNN (2016) [26]	76 %	67 %
Ahmad P., Hossein A. N., Marina G., Stevlana N.Y. based on AU (2012) [25]	96 %	89 %
Surti S. Sachin P. based on AU (Gaber & SVM) 2012 [24]	93 %	89 %
Este proyecto CNN (2017)	73 %	71 %

Normalmente, las redes convolucionales para realizar

“Deep Learning” siguen una estructura de tipo capa convolucional, ReLU, maxpooling, pero visto en un artículo sobre detección de emociones faciales su configuración [26] y con la conclusión de que el maxpooling reduce dimensionalidad pero también calidad, se ha usado esta estructura en nuestra red neuronal convolucional obteniendo mejores resultados.

Es interesante comparar nuestros resultados con los mejores resultados obtenidos de otros proyectos (Tabla 10), donde comprobamos que realmente no hemos llegado a un tan buen detector de emociones, pero sí muchísimo más simple.

Analizando los resultados de los otros proyectos y donde los mejores resultados obtenidos han sido entrenando con AU (Action Units, detallado en el artículo Cohn-Kanade [5] y donde se da en cada sujeto que Action Unit tiene) otra de las tareas pendientes y con posible futuro es entrenar de esta forma.

Otra de las tareas es entrenar con una red neuronal convolucional preentrenada por otro dataset relacionado con características faciales donde podríamos obtener mejores resultados, encontrando una arquitectura más adecuada como sustituir las capas ocultas de la red convolucional elegida del proyecto (figura 3) por otra red neuronal convolucional para tener más información en cada neurona.

También es interesante de la misma red convolucional del proyecto (figura 3) incorporar los pesos de las convoluciones en una única capa de convolución, teniendo tres filtros en cada convolución y una sola imagen (el rostro detectado).

AGRADECIMIENTOS

Mis mas sinceras gracias a la tutora de este trabajo Katerine Díaz, ya que ha apoyado hasta el último momento de este trabajo.

Gracias también a Estela Sierra por el apoyo y los consejos otorgados, ya que sin ellos no podría haber llegado a tan buena conclusión. También a mi amigo Jacobo Delgado por tan buenos consejos me dió en su día.

REFERENCIAS

- [1] <http://iwann.uma.es>
- [2] API de Microsoft relacionados con reconocimiento de emociones y de rostros. Link: <https://github.com/Microsoft/>
- [3] Artículo sobre las APIs de empresas actuales en reconocimiento de emociones. Link: <http://nordicapis.com>
- [4] Dataset de rostros faciales con diferentes sujetos. Link: <http://www.karsl.org>
- [5] Datasets 10k US Faces y Cohn Kanade con diferentes sujetos y anotaciones de las emociones proporcionados por la profesora Katherine Diaz.
- [6] Dataset yale-face de rostros faciales con 15 sujetos, donde constan 11 imágenes de cada sujeto con diferentes expresiones. Link: <http://vision.ucsd.edu>
- [7] Proyectos de investigación sobre la detección de rostros, reconocimiento, seguimiento, reconocimiento de gestos y biometría. Link: <http://www.visionbib.com>
- [8] Clase FaceDetector de Android para detectar rostros y KeyPoints. Link: <https://developer.android.com>
- [9] Tutorial y definición del clasificador Haarcascade con OpenCV. Link: <http://docs.opencv.org>
- [10] Librería DLib usando la detección de KeyPoints. Link: <http://dlib.net>
- [11] Artículo sobre máquina vector por soporte. Link: <http://nlp.stanford.edu>
- [12] Framework en Python donde trabaja con funciones precompiladas. Link: <http://deeplearning.net>
- [13] Framework que trabaja sobre Theano. Link: <http://lassagne.readthedocs.io>
- [14] Artículo sobre robotica afectando positivamente a la salud humana. Link: <https://news.vanderbilt.edu>
- [15] Artículo de investigación sobre clasificación de personas. Link: <http://cs231n.stanford.edu>
- [16] Introducción a incorporar NDK a Android. Link: <https://developer.android.com>
- [17] Información, tutoriales y estructuras para implementar DeepLearning en una red neuronal. Link: <http://deeplearning.net>
- [18] Triangulación delaunay explicado paso a paso. Link: <https://es.mathworks.com>
- [19] Framework en C++ para visión por computador. Link: <http://opencv.org>
- [20] Artículo sobre el diseño de detección de señales de tráfico y sistemas de transmisión para vehículos inteligentes. Link: <http://ieeexplore.ieee.org>
- [21] Librerías de C++ en reconocimiento de rostros en tiempo real usando OpenCV. Link: <http://facetracker.net>
- [22] Artículo sobre Overfitting y Underfitting. Link: <http://visionbib.com>
- [23] Análisis de componentes principales de los datos en bruto (PCA). Descripción y implementación en Python. Link: <http://scikit-learn.org>
- [24] Artículo sobre reconocimiento de emociones utilizando el dataset Jaffe y CK. Link: <http://research.ijcaonline.org>
- [25] Artículo sobre reconocimiento de emociones basado en los AU. Link: <https://www.researchgate.net>
- [26] Artículo proporcionado por la profesora Katherine Diaz sobre el reconocimiento de emociones faciales con una red neuronal convolucional. Link: <http://dl.acm.org>

APÉNDICE

A.1. Mejores resultados con SVM

A continuación presentamos mejores resultados con SVM.

TABLA 11: TABLA DE CONFUSIÓN DEL DATASET CK

		Esperado						Acc.
		1	2	3	4	6		
Predecido	1	0	0	3	0	0	0	
	2	0	1	0	0	0	1	
	3	14	3	1	3	1	.05	
	4	0	5	0	0	0	0	
	6	0	0	2	1	4	.57	
Accuracy		0	.11	.16	0	.8	16 %	

TABLA 12: TABLA DE CONFUSIÓN DEL DATASET JAFFE

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	0	1	1	2	1	0
	2	2	5	3	3	2	.33
	3	1	0	1	0	0	.5
	5	3	0	0	1	0	.25
	6	0	0	1	0	3	.75
Accuracy		0.0	.83	.17	.17	.5	33 %

TABLA 13: TABLA DE CONFUSIÓN DEL DATASET 10K

		Esperado						Acc.
		1	2	3	5	6		
Predecido	1	350	10	37	43	10	.79	
	2	0	0	0	0	0	0.0	
	3	0	0	0	0	0	0.0	
	5	0	0	0	0	0	0.0	
	6	0	0	0	0	0	0.0	
Accuracy		1.0	0.0	0.0	0.0	0.0	79 %	

TABLA 14: TABLA DE CONFUSIÓN DEL DATASET YALE-FACES

		Esperado				Acc.
		1	3	5		
Predecido	1	3	0	0		1.0
	3	2	6	6		.42
	5	1	0	0		0.0
Accuracy		0.6	1.0	0.0		50 %

Estudiando los resultados de las tablas de confusión aportadas, se ha llegado a la conclusión que el clasificador no ha llegado a un buen resultado porque necesita una función más compleja, siendo imposible clasificar correctamente.

TABLA 15: TABLA DE CONFUSIÓN DE TODOS LOS DATASETS

		Esperado						Acc.
		1	2	3	4	5	6	
Pred.	1	370	25	37	4	45	14	.75
	2	0	0	3	0	0	0	0.0
	3	0	0	3	0	2	0	.6
	4	0	0	0	0	0	0	0.0
	5	0	0	0	0	1	0	1.0
	6	0	0	15	0	4	0	0.0
Accuracy		1.0	0.0	.05	0.0	.01	0.0	73 %

A.2. Mejores resultados NN con backpropagation

A continuación presentamos los mejores resultados con una red neuronal simple con backpropagation.

TABLA 16: TABLA DE CONFUSIÓN DEL DATASET CK

		Esperado						Acc.
		1	2	3	4	6		
Predecido	1	14	2	2	0	2	.70	
	2	0	1	0	0	0	1.00	
	3	0	2	0	2	0	0	
	4	0	0	0	0	0	0	
	6	0	4	4	2	3	.24	
Accuracy		1.0	.11	0	0	.60	47 %	

TABLA 17: TABLA DE CONFUSIÓN DEL DATASET JAFFE

		Esperado						Acc.
		1	2	3	5	6		
Predecido	1	0	0	0	0	0	0	
	2	6	6	6	6	6	.2	
	3	0	0	0	0	0	0	
	5	0	0	0	0	0	0	
	6	0	0	0	0	0	0	
Accuracy		0.0	1	0.0	0.0	0.0	20 %	

TABLA 18: TABLA DE CONFUSIÓN DEL DATASET 10K

		Esperado						Acc.
		1	2	3	5	6		
Predecido	1	350	10	37	43	10	.79	
	2	0	0	0	0	0	0.0	
	3	0	0	0	0	0	0.0	
	5	0	0	0	0	0	0.0	
	6	0	0	0	0	0	0.0	
Accuracy		1.0	0.0	0.0	0.0	0.0	79 %	

Aportando un poco más de información a los resultados que el clasificador anterior, sigue necesitando una función más compleja (no es capaz de clasificar correctamente).

TABLA 19: TABLA DE CONFUSIÓN DEL DATASET YALE-FACES

		Esperado			Acc.
		1	3	5	
Predecido	1	0	0	0	0.0
	3	6	6	6	.33
	5	0	0	0	0.0
Accuracy		0.0	1.0	0.0	33 %

TABLA 20: TABLA DE CONFUSIÓN DE TODOS LOS DATASETS

		Esperado						Acc.
		1	2	3	4	5	6	
Pred.	1	370	25	58	4	52	14	.71
	2	0	0	0	0	0	0	0.0
	3	0	0	0	0	0	0	0.0
	4	0	0	0	0	0	0	0.0
	5	0	0	0	0	0	0	0.0
	6	0	0	0	0	0	0	0.0
Accuracy		1.0	0.0	0.0	0.0	0.0	0.0	71 %

A.3. Mejores resultados CNN

A continuación se presenta los mejores resultados de la red neuronal convolucional con 8 filtros en la primera convolución, 40 en la segunda y 200 unidades en la capa oculta.

TABLA 21: TABLA DE CONFUSIÓN DEL DATASET CK

		Esperado					Acc.
		1	2	3	4	6	
Predecido	1	13	1	1	0	0	.86
	2	1	5	2	1	1	.5
	3	0	2	3	0	2	.42
	4	0	0	0	3	0	1.00
	6	0	1	0	0	2	.67
Accuracy		.93	.63	.50	.75	.50	71 %

TABLA 23: TABLA DE CONFUSIÓN DEL DATASET 10K

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	0	0	0	0	0	0.0
	2	0	0	0	0	0	0.0
	3	0	0	0	0	0	0.0
	5	350	10	37	43	10	0.1
	6	0	0	0	0	0	0.0
Accuracy		0.0	0.0	0.0	1.0	0.0	10 %

TABLA 22: TABLA DE CONFUSIÓN DEL DATASET JAFFE

		Esperado					Acc.
		1	2	3	5	6	
Predecido	1	2	0	0	0	0	1
	2	0	6	2	0	1	.67
	3	1	0	2	0	1	.5
	5	2	0	1	5	0	.71
	6	1	0	1	1	4	.57
Accuracy		.33	1	.33	.83	.67	63 %

Analizando los resultados, esta arquitectura sí ha llegado a una función compleja debido a las capas convolucionales, ya que agrupan más información, produciendo mejores resultados.

En la tabla 23 ha clasificado incorrectamente debido a que se ha producido overfitting.