TRABAJO FIN DE GRADO

INGENIERÍA EN INFORMÁTICA

Reconocimiento de Expresiones Faciales

por Computador

Autor
Francisco José Fajardo Toril

Directores
Miguel García Silvente
Eugenio Aguirre Molina

Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación

—

Granada, 11 de agosto de 2017

Reconocimiento de Expresiones Faciales

por Computador.

Autor
Francisco José Fajardo Toril

Directores
Miguel García Silvente
Eugenio Aguirre Molina

Reconocimiento de Expresiones Faciales: por Computador

Francisco José Fajardo Toril

Palabras clave: Reconocimiento, expresión, facial, visión, computador, aprendizaje, automático, redes, neuronales, convolucionales.

Resumen

El objetivo de este documento es mostrar al lector el proceso seguido para el desarrollo de una aplicación software capaz de resolver el problema del reconocimiento de expresiones faciales por computador, que puede ser descrito como:

Dada una imagen de entrada, obtener la etiqueta de salida que mejor identifique la expresión facial que aparece en dicha imagen.

De entre todas las técnicas posibles para la resolución del problema, he basado mi solución en el entrenamiento una red neuronal convolucional. El modelo entrenado clasifica la imagen de entrada mediante la asignación de una (o varias) etiquetas, incluyendo un porcentaje por cada una que indica cuál es la más probable de estar presente en dicha imagen. El conjunto de posibles etiquetas es:

- Afraid
- Angry
- Disgusted
- Happy
- Neutral
- Sad
- Surprised

Facial Expression Recognition: by Computer

Francisco José Fajardo Toril

Keywords: Facial, expression, recognition, computer, vision, machine, learning, convolutional, neural, network.

Abstract

This document main objective is to show the reader the process I followed to develop a software application which is able to solve the face expression recognition problem through computer. It can be described as:

To obtein the tag that better describe the facial expression that appears in the given input image.

Between all the possible techniques that allow us to solve this problem, I chose to train a convolutional neural network. Trained model classify the input image by the assignment of one or few tags, including a percentage that suggest which is the most probable tag to appear in that image. Here is the whole set of possible tags:

- Afraid
- Angry
- Disgusted
- Happy
- Neutral
- Sad
- Surprised

Yo, **Francisco José Fajardo Toril**, alumno de la titulación Grado en Ingeniería Informática de la **Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación de la Universidad de Granada**, con DNI 76424577Q, autorizo la ubicación de la siguiente copia de mi Trabajo Fin de Grado en la biblioteca del centro para que pueda ser consultada por las personas que lo deseen.

Fdo: Francisco José Fajardo Toril

Granada a 11 de agosto de 2017.

D. **Miguel García Silvente**, Profesor del Área de Ciencias de la Computación e Inteligencia Artificial del Departamento YYYY de la Universidad de Granada.

D. **Eugenio Aguirre Molina**, Profesor del Área de Ciencias de la Computación e Inteligencia Artificial del Departamento YYYY de la Universidad de Granada..

Informan:

Que el presente trabajo, titulado ***Reconocimiento de Expresiones Faciales, por Computador***, ha sido realizado bajo su supervisión por **Francisco José Fajardo Toril**, y autorizamos la defensa de dicho trabajo ante el tribunal que corresponda.

Y para que conste, expiden y firman el presente informe en Granada a 11 de agosto de 2017.

Los directores:

Miguel García Silvente **Eugenio Aguirre Molina**

Agradecimientos

Poner aquí agradecimientos...

Capítulo 1

Introducción

Tal y como mencioné en el resumen, el problema del reconocimiento de expresiones faciales mediante computador puede ser descrito de la siguiente manera:

Dada una imagen de entrada, obtener la etiqueta de salida que mejor identifique la expresión facial que aparece en dicha imagen.

A priori la solución para un ser humano es trivial. Tenemos muchas maneras de adivinar el estado de ánimo de una persona, pero si tenemos que basarnos en nuestra visión, una posible solución sería algo como: *"Simplemente observa los rasgos faciales del sujeto e intenta adivinar su estado de ánimo basándote en tu experiencia pasada con otros seres humanos."*
Imagine que deseamos desarrollar un agente basado en inteligencia artificial para que interactue con otros seres humanos. Sería de gran utilidad que este agente basara sus acciones en función del estado de ánimo de la persona/as con las que está comunicándose para lograr de esta forma un mayor grado de empatía. De la misma manera podríamos pensar en una aplicación que escoge la música idónea para nosotros en función de nuestro estado de ánimo en ese momento a través de una captura mediante la cámara del computador.

Para ser capaces Sin embargo, ¿cómo podría un computador dar solución al problema que aquí se presenta?
Imagine que deseamos El objetivo de este documento es mostrar al lector el proceso seguido para el desarrollo de una aplicación software capaz de resolver este problema. Esto implica discutir sobre las técnicas referentes a la Inteligencia Artificial que nos proporcionan una mejor aproximación a la solución. Para ello se intentará dar respuesta a las preguntas que este problema plantea de manera inmediata, como por ejemplo:

- ¿Cómo saber si aparece una cara en una imagen?

- Y suponiendo que hay una cara, ¿Cómo codificar una expresión facial de manera que el computador pueda distinguirlas? ¿Es realmente necesario codificarlas?

Una vez seleccionadas las técnicas más útiles que responden a las preguntas anteriores, se tratará el problema desde el punto de vista del desarrollo software. Es decir, justificar las decisiones que se han tomado y describir el proceso que se ha seguido para la implementación de la aplicación final.

- Aquí hay que contextualizar el TFG, es decir, explicar la motivación que existe para realizar el TFG describiendo la realidad de la que se parte haciendo un estudio del estado del arte breve con referencias bibliográficas a trabajos que han tratado también el tema que vamos a abordar.
- Una vez descrito el marco general, se explica lo que se quiere hacer con el proyecto con la propuesta concreta que se pretende conseguir.
- Se termina explicando la organización de la memoria diciendo lo que se trata en cada capítulo.

Capítulo 2

Objetivos

Objetivo general a conseguir: se enuncia lo que se quiere hacer sin entrar en detalles.

Objetivos específicos: Es dividir el objetivo general en los pasos a seguir con sub-objetivos más simples. Dicho de otro modo, son cada uno de los pasos a realizar para alcanzar el objetivo general, es decir solucionando todos y cada uno de los objetivos específicos se resuelve el objetivo general. (Poner entre 3 y 5 como mucho).

Se puede decir en qué apartado se tratará cada objetivo específico.

Poner también los aspectos formativos previos utilizados, por ejemplo si se han usado técnicas de visión concretas como Transformada de Hough, Método de detección de rostros Viola-Jones, Filtro de Partículas, o técnicas de aprendizaje por SVM, etc. Se explica un poco cada método.

Capítulo 3

Resolución del trabajo

Como método de ingeniería del software decir que vamos a seguir la técnicas de modelo de prototipos rápido o también llamado modelado de prototipado rápido.

Ver `http://www.ecured.cu/index.php/Modelo_de_Prototipos`

3.1. Recursos

Decir los recursos humanos (autor y directores), hardware y software que se van a utilizar.

3.2. Especificación de requisitos

Decir que se partió de una especificación inicial de requisitos que a medida que se fueron implementando los prototipos se fue refinando posteriormente. Se puede poner la inicial y la final o solo la final indicando que se están poniendo los requisitos que finalmente tiene que tener el sistema.

Los requisitos se pueden referir a las necesidades del usuario del sistema (requisitos del usuario), a lo que tiene que hacer la aplicación (requisito funcional) o a cómo tiene que hacerlo (requisito no funcional). Ejemplo:

En este sistema un robot tiene que coger con sus pinzas un envase de medicamento y llevárselo a una persona anciana que por sí misma no puede recordar su medicación.

Requisitos del usuario:

RU1. La persona puede moverse libremente por una habitación donde está el robot.

RU2. La persona es capaz de coger el medicamento cuando se lo ofrece el robot.

RU3. La persona es capaz de tomarse el medicamento por sí misma.

Requisitos funcionales.

RF1. El robot mediante la cámara kinect debe poder localizar a la persona.

RF2. El robot conoce la posición de la mesa pues tiene un mapa de la habitación.

RF3. El robot debe identificar el medicamento correcto según un plan de medicación previamente establecido.

RF4. El robot debe poder coger el medicamento con sus pinzas.

etc...

Requisitos no funcionales

RNF1. El robot no puede atropellar ni dañar a la persona en ningún momento.

RNF2. La aplicación debe ejecutarse en entornos linux

RNF3. La aplicación debe utilizar pocos recursos para reaccionar con rapidez.

algo de la interfaz, como tratar posibles fallos, etc.

3.3. Planificación

Poner una tabla de tiempos con las planificación del proyecto diciendo cuando se tiene previsto alcanzar cada subobjetivo planteado. Con su correspondiente división en fases y tareas, y la posterior comparación con los datos reales obtenidos tras realizar el proyecto. Entre las fases está la realización de los diferentes prototipos I, II y III por ejemplo.

Poner presupuesto según horas de trabajo estimadas.

3.4. Análisis funcional

A partir de aquí nos referimos solamente al prototipo final que da lugar a la aplicación final.

Hay que describir la funcionalidad que debe poseer el sistema para poder cumplir con los objetivos y requisitos que se han dicho previamente.La descripción de esta funcionalidad puede hacerse analizando las tareas (que aparecerán en la planificación) y estudiando la inter-relación entre ellas y sus conexiones.

Para la realización de este análisis se pueden utilizar Diagramas de Flujo para poder conocer generalmente un único punto de inicio y un único punto de término o en varios.

`https://es.wikipedia.org/wiki/Diagrama_de_flujo`

Se pueden plantear también casos de uso. Los diagramas de casos de uso sirven para describir la inter-relación entre el sistema y el usuario del mismo. Se pueden utilizar para plantear diferentes casos de interacción entre el robot y la persona y cómo tiene que reaccionar el sistema en cada caso.

`https://es.wikipedia.org/wiki/Diagrama_de_casos_de_uso`

3.5. Implementación y pruebas

Decir qué lenguaje de programación se ha utilizado y las tecnologías implicadas aunque se hayan comentado en el apartado de recursos. Justificar su uso (rendimiento, disponibilidad, etc.). Si se ha usado open source decirlo y explicar las ventajas.

Las pruebas se realizan para comprobar la verificación y validación del producto software. La verificación consiste es comprobar que el producto realiza lo que está programado, es decir la programación no tiene errores y funciona en todos los casos cumpliendo los requisitos. La validación tiene que ver con que cumpla con lo que espera el usuario.

Verificación y Validación: Conjunto de procesos de comprobación y análisis que aseguran que el software que se desarrolla está acorde a su especificación y cumple las necesidades de los clientes.

Verificación: ¿Estamos construyendo el producto correctamente? Se comprueba que el software cumple los requisitos funcionales y no funcionales de su especificación.

Validación: ¿Estamos construyendo el producto correcto? Comprueba que el software cumple las expectativas que el cliente espera Importante: Nunca se va a poder demostrar que el software está completamente libre de defectos

las pruebas que pueden utilizarse son muy diversas. Aconsejo centrarnos en pruebas de caja blanca y de caja negra.

Las pruebas de la caja blanca se centran en la estructura interna del programa para elegir los casos de prueba. El objetivo de estas pruebas consiste en probar todos los posibles casos de ejecución de la aplicación para comprobar que los datos se comportan de manera correcta internamente.

Decir que se han hecho las pruebas de caja blanca.

Las pruebas de caja negra son aquellas que se centran en las salidas y entradas de los módulos, sin atender a su comportamiento interno (comprobando mediante las pruebas de caja blanca). Las pruebas de caja negra garantizan la interconectividad entre los diferentes módulos de la aplicación, así como su correcto funcionamiento final.

Poner algunos casos de prueba de caja negra.

Capítulo 4

Conclusiones y trabajo futuro

Decir lo que se ha conseguido realizar comentando sus puntos fuertes y débiles. Decir si se han alcanzado los objetivos específicos y el general propuesto y en qué grado.

Indicar las asignaturas del grado más relacionadas con la ejecución del TFG y cómo el TFG ha ayudado a afianzar los conocimientos adquiridos en el Grado.

Valoración personal si se quiere.

Avanzar algunas líneas de trabajo futuro para solucionar las debilidades detectadas o para conseguir nuevas funcionalidades interesantes.

Capítulo 5

Bibliografía

Poner las citas bibliográficas, direcciones de internet, etc.

Capítulo 6

Anexo

Al final de la memoria hay que añadir un anexo de una página o dos, explicando como se usa el software a modo de manual de usuario. Es decir, como se llaman los comandos, qué parámetros hay que darle, como se llaman los ficheros de datos de entrada, etc.