Instituto Tecnológico de Apizaco

Departamento de Sistemas Computacionales

Clasificación de formularios de diseño de prototipos usando un algoritmo de máxima expectación

Tesis presentada por Baldemar Zurita Islas
para obtener el grado de Maestro en Sistemas Computacionales

Director de Tesis:
Dr. José Crispín Hernández Hernández

Co-Director de Tesis:
Dr. Edmundo Bonilla Huerta

Apizaco, Tlaxcala. 2019

Agradecimientos

Me gustaría agradecer en estas lineas la ayuda que muchas personas y colegas me han prestado durante el proceso de investigación y redacción de este trabajo. En primer lugar, a Dios por darme fuerza para continuar en este proceso de obtener uno de los anhelos más deseados, a mi madre que me ha ayudado y apoyado en todo momento, a mi tío, Luís Francisco, por haberme orientado en todos los momentos que necesité sus consejos.

Así mismo, deseo expresar mi reconocimiento al Dr. Crispín Hernández por aceptarme como su tutorado y brindarme la confianza y libertad durante todo el proyecto, a la comisión revisora: Dr. Edmundo Bonilla, Dr. Federico Ramírez y M.C. Eduardo Sánchez por todas las atenciones e información brindada a lo largo de esta indagación.

A todos mis amigos, entre ellos el Cucho, vecinos y futuros colegas que me ayudaron de una manera desinteresada y una mención especial a los compañeros que conocí en este trayecto: Chino, Espina, Moi, Chado y China. gracias infinitas por toda su ayuda y buena voluntad.

Al Instituto Tecnológico de Apizaco por ser la sede de todo el conocimiento adquirido en estos años.

GRACIAS... TOTALES.

Abstract

This proyect presents the visual classification of objects and the development of a system that, given several images, can be classified into several categories based on the shared visual characteristics, This is a very important area in computer vision and there is a wide variety of research on this subject, since it has too many important applications at the forefront of computing, specifically in robotics, automated systems and mobile devices.

The key aspects of the research will be highlighted, which cover the antecedents of artificial vision, specifically the classification of images, up to the current research. A method known as Bag of Visual Words will be implemented and compared with the current state of the art. This model will be tested with images other than photographs.

The objective of this project is the classification of images from a database based on the content obtained by users making hand-drawn lines of geometric figures or symbols. These images will contain objects that the classifier must categorize correctly, in order to improve the classification success rate, new methods will be tested and existing ones will be adjusted.

Resumen

En este proyecto se presenta la clasificación visual de objetos y el desarrollo de un sistema que, dadas varias imágenes, pueda clasificar en diferentes categorías en función de las características visuales compartidas. Esta es un área muy importante de la visión por computadora y hay una gran variedad de investigaciones sobre este tema, ya que tiene demasiadas aplicaciones importantes en la vanguardia de la informática, específicamente en robótica, sistemas automatizados y también en dispositivos móviles.

Se resaltarán los aspectos clave de la investigación, que cubren los antecedentes de la visión artificial, específicamente la clasificación de imágenes, hasta la investigación actual. Se Implementará un método conocido como Bag of Visual Words y se comparará con el estado del arte actual. Este modelo se probará con imágenes que no sean fotografías.

El objetivo de este proyecto es la clasificación de imágenes de una base de datos basada en el contenido obtenida por usuarios haciendo trazos a mano de figuras geométricas o símbolos. Estas imágenes contendrán objetos que el clasificador deberá de categorizar de manera correcta, con el propósito de mejorar la tasa de éxito de clasificación se probrarán diferentes métodos y se ajustarán los existentes.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

La visión por computadora es un campo de la informática que está a la vanguardia y ha seguido creciendo enormemente desde la década de 1970, cuando se hizo posible por primera vez, a medida que las computadoras se volvían más potentes y capaces de procesar los datos. Su objetivo general es permitir que las computadoras procesen y entiendan el mundo a su alrededor a través de entradas visuales Estas entradas generalmente provienen de una o varias cámaras, pero también pueden incluir escáneres y otros equipos especializados. La visión por computadora cubre una amplia gama de temas y técnicas que ahora se resumirán algunos de los más importantes.

Un área de visión artificial que se usa frecuentemente en investigación es el análisis de movimiento, donde una computadora recibe una secuencia de imágenes (generalmente un video) y puede rastrear elementos y calcular información tal como su posición en una escena 3D o su velocidad en la imagen actual. A menudo se utiliza para analizar objetivos múltiples, como las personas que se mueven en una multitud (Zhao and Nevatia, 2004). La visión por computadora también aborda problemas tales como la creación de un modelo 3D de una entrada del mundo real, por ejemplo, la reconstrucción de escenas. El uso con robótica es una de las principales aplicaciones de esto.

La restauración de imágenes ahora se usa ampliamente para limpiar imágenes, generalmente de fuentes de baja calidad. Esto abarca desde simples filtros de eliminación de ruido hasta enfoques mucho más complejos. Probablemente el área más

compleja y más investigada de la visión artificial es el reconocimiento. Aquí es donde una computadora identifica algo de una fuente visual y puede traducir la señal visual en datos comprensibles de la máquina. La recuperación de imágenes basada en contenido es una forma de buscar un conjunto de imágenes usando palabras clave o incluso una imagen similar. Por ejemplo esto está implementado a gran escala con las nuevas capacidades del motor de búsqueda de imágenes Google.

Una aplicación común para el reconocimiento es el reconocimiento óptico de caracteres (OCR Optical Charater Recognition), que toma una imagen de texto y lee el texto, y lo emite en un formato de texto digital. Esto se usa comúnmente para digitalizar documentos como viejos diarios, periódicos y manuscritos. El trabajo de Wong, Casey y Wahl (Wong et al., 1982) sobre el Sistema de análisis de documentos formó una gran base para esto y continúa mejorando incluso hoy en día. El área en la que nos centraremos principalmente en este proyecto es la clasificación de objetos que se encuentra en el campo de reconocimiento. La clasificación de objetos tiene como objetivo identificar objetos de manera efectiva desde imágenes.

En general, la clasificación de objetos es donde se toma y se procesa una imagen o serie de imágenes. El clasificador debería mostrar etiquetas para los diferentes segmentos de las imágenes. Estas etiquetas contienen información sobre el objeto de la imagen. Por ejemplo, varias imágenes diferentes de motocicletas deberían tener la misma etiqueta, incluso si se toman desde diferentes ángulos, en diferentes condiciones de iluminación, o incluso si las motocicletas son de diferentes colores y modelos. La clasificación de objetos es un área muy importante de la visión artificial, como lo ilustra la cantidad de investigaciones sobre el tema. Tiene diversas aplicaciones, incluida la robótica, búsqueda de imágenes, seguridad (específicamente reconocimiento facial), asistencia a personas con discapacidades visuales, censura de imágenes, por citar algunas. El campo de la clasificación de objetos ha estado creciendo durante años con técnicas diferentes que van desde completamente manuales (Von Ahn and Dabbish, 2004) hasta totalmente automatizadas (Sivic et al., 2005).

En este trabajo de tesis, con el objetivo de clasificar imágenes de trazos dibujados a mano por usuarios, se propone el uso del modelo Bag of Visual Words en combinación de un algoritmo de máxima expectación (EM) y el uso de la biblioteca de OpenCV en conjunto con el lenguaje de programación Python 2.7.

1.1. Planteamiento del Problema

El reconocimiento de escritura es difícil debido a la gran variabilidad que se encuentra en la escritura humana. Además, la apariencia de los símbolos varía y depende de su contexto local, por ejemplo, un grupo de individuos puede dibujar una serie de símbolos o figuras geométricas, estos dibujos variarán en tamaño, apariencia y simetría por cada individuo que lo dibujó, pero al comparar todos los dibujos entre si, se encontrarán características únicas de cada figura geométrica que la distinga de las demás, así podrán ser catalogadas dentro de la clase que pertenece cada una. Para hacer frente a estos desafíos, un sistema típico de reconocimiento de escritura aplica una serie de métodos. Después de normalizar las imágenes con respecto a las variabilidades mencionadas, se extrae una representación que se puede utilizar en un reconocedor de escritura a mano. El resultado final es una transcripción de la imagen. Se requiere diseñar un reconocedor de escritura a mano que sea preciso y muestre resultados en el menor tiempo posible.

1.2. Pregunta de investigación

¿Qué impacto puede presentar el algoritmo EM en el modelo Bag of Visual Words, se podrá obtener una mejor clasificación de imágenes en un tiempo menor a 1 segundo?

1.3. Justificación

En el modelo bag of visual words, el algoritmo de agrupamiento que utiliza por defecto es K-means, EM y K-means son similares en el sentido de que permiten la refinación de un modelo en un proceso iterativo para encontrar la mejor congestión o agrupamiento de datos. Sin embargo, el algoritmo K-means difiere en el método utilizado para calcular la distancia euclidiana al calcular la distancia entre cada uno de los dos elementos de datos, por otro lado, EM usa métodos estadísticos. El algoritmo EM se usa a menudo para proporcionar las funciones de manera más efectiva.

1.4. Objetivos

El objetivo de este proyecto consiste en la aplicación del modelo bag of words (BoW, bolsa de palabras) al reconocimiento de escritura a mano. Este modelo, procede del análisis y clasificación de contenidos textuales, requiere de una adaptación al procesamiento visual y la clasificación de imágenes. El modelo BoW define una metodología de trabajo para clasificar imágenes, si bien numerosos aspectos concretos de su aplicación quedan pendientes del diseño del desarrollador, dichos parámetros sobre el modelo, tales como el tamaño y estructura de los vocabularios, o el empleo de clasificadores de distinta índole, serán estudiados a lo largo del proyecto. Se implementarán diversas alternativas para la clasificación, lo que requerirá métodos de evaluación y la comparativa de todos los resultados.

1.4.1. Objetivo General

Teniendo en cuenta las ventajas y versatilidad del modelo de bag of words, el objetivo de este trabajo es utilizar una integración eficiente para el procesamiento del algoritmo EM, que permita reducir la magnitud los tiempos de procesamiento respecto a la implementación del algoritmo original.

1.4.2. Objetivos Específicos

Como objetivos específicos de este trabajo se encuentran los siguientes:

- Identificar el modelo más adecuado para desarrollar un sistema de Bag of visual words.
- Usar la biblioteca OpenCV para el manejo eficiente del procesamiento de los píxeles.
- Integrar el clasificador EM (Máxima Expectación)
- Utilizar al menos dos modelos de evaluación para obtener el rendimiento del clasificador de imágenes (bag of visual words).
- Obtener un uso eficiente del clasificador EM en el modelo Bag of visual words para reconocer trazos a mano alzada.

1.5. Organización de Tesis

El proyecto de investigación se divide en 6 capítulos para su lectura, a continuación se resume cada capítulo que lo conforma.

- Capítulo 1. Se introduce al lector en la parte inicial del trabajo, describiendo el tema de investigación, la problemática del tema de investigación y los objetivos a los que se pretende llegar al concluir el trabajo de investigación.

- Capítulo 2. Se desarrolla el estado del arte para conocer propuestas de diferentes trabajos de investigación que se relacionan al abordar la problemática de interés.

- Capítulo 3. En este capítulo se describe el algoritmo EM sus características y aplicaciones. Posteriormente se desarrolla el anáisis de las etapas que comprenden a el modelo de bag of visual words, para identificar aquellas etapas viables de modificar, también el análisis de los métodos de evaluación del modelo bag of visual words. Se aborda el uso de las librerías de el lenguaje Python y OpenCV.

- Capítulo 4.Se muestra la integración del algoritmo EM en el modelo de Bag of visual words, describe la integración de las operaciones para reducir el costo computacional y las distintas tareas que procesa el clasificador para alcanzar la salida deseada.

- Capítulo 5.Se presenta los resultados obtenidos, la comparativa de los tiempos de ejecución y clasificación. Se presenta un análisis de evaluación por etapas (las de mayor interés) y de igual forma el procesamiento global. Finalmente se plantea la discusión de los alcances del trabajo desarrollado en este proyecto de investigación.

- Capítulo 6.Plantea las conclusiones a las que se llega finiquitado el trabajo de investigación, se plantean y visualizan mejoras que se pueden realizar en el trabajo realizado.

Capítulo 2

Estado del Arte

El objetivo de esta revisión de la literatura es evaluar el arte actual en el área de la visión por computadora y más específicamente en torno a la clasificación de objetos y las técnicas involucradas. Esto debería proporcionar un punto de partida para este proyecto con la información necesaria para comparar las diferentes técnicas disponibles, para determinar cuál sería el mejor método a seguir. En las siguientes secciones, nuestro objetivo es definir claramente qué es Computer Vision y por qué es importante, y profundizar en algunas de las investigaciones en el área, centrándose específicamente en las técnicas de clasificación de objetos. Compararemos y contrastaremos las estrategias actuales para determinar qué técnica será más adecuada para este proyecto, especialmente teniendo en cuenta el interés en las imágenes de entrada no fotográficas.

2.1. Introducción a la Visión por Computadora

Computer Vision es un campo que está a la vanguardia de la informática moderna y ha continuado Creciendo enormemente desde la década de 1970, cuando se hizo posible por primera vez a medida que las computadoras se hacían más potentes y capaces de procesar los datos. Su objetivo general es permitir que las computadoras procesen y entiendan el mundo que les rodea a través de entradas visuales. Estas entradas son generalmente de cámaras individuales o múltiples, pero también pueden incluir escáneres multidimensionales y otros equipos especializados. Esto tiene

muchas aplicaciones importantes en el mundo real que van desde organizar datos hasta procesamiento de imágenes médicas. La Visión por Computadora cubre una amplia gama de temas y técnicas que veremos ahora.

Se esbozarán algunos de los más importantes. Un área de la visión por computadora que se usa mucho en la investigación es análisis de movimiento, donde a una computadora se le da una secuencia de imágenes (generalmente como un video) para hacer un seguimiento de elementos y calcular información como su posición en una escena 3D o su velocidad en la imagen actual. A menudo se usa para analizar múltiples objetivos, como personas que se mueven en una multitud (Zhao and Nevatia, 2004).

La visión por computadora también aborda problemas como la creación de un modelo 3D de una entrada del mundo real, por ejemplo. reconstrucción de la escena. Esta es la idea de tomar imágenes de una escena específica y/o construir una representación 3D de la misma en la computadora. Esto puede ser útil para identificar o clasificar entornos. (Yang and Ngo, 2007). La robótica es una de sus principales aplicaciones.

La restauración de imágenes ahora se usa ampliamente para limpiar imágenes, generalmente de fuentes de baja calidad. Esta abarca desde simples filtros de eliminación de ruido hasta enfoques mucho más complejos que comprende la estructura lógica de la imagen. Esto permite mejorar las imágenes, a menudo con suciedad, ruido o incluso objetos no deseados son eliminados. También puede restaurar los detalles perdidos. Es una técnica útil cuando se trata de imágenes antiguas, o película de baja calidad como imágenes de CCTV.

Probablemente el área más compleja y más investigada de La visión por computadora es el reconocimiento. Aquí es donde una computadora identifica algo de una fuente visual y puede traducir la señal visual en datos comprensibles de la máquina. El reconocimiento puede ser utilizado para Detectar una afección específica, que es especialmente útil en imágenes médicas para detectar huesos rotos o daño tisular, ya que las computadoras pueden ver detalles mucho más finos que el ojo humano (Koh et al., 2018). Basado en el contenido de la Imagen, la recuperación es una forma de buscar un conjunto de imágenes usando palabras clave o incluso una imagen similar. Esto se implementó a gran escala con las nuevas capacidades de búsqueda de

imágenes similares de Google (Sivic and Zisserman, 2003) .

Una aplicación común para el reconocimiento es el reconocimiento óptico de caracteres (OCR), que toma una imagen de texto y lee el texto, generándolo en un formato de texto digital. Esto es comúnmente usado para digitalizar documentos como periódicos antiguos, libros y manuscritos. (Wong et al., 1982). El trabajo en el análisis de documentos formó una gran base para esto y continúa siendo mejorado incluso hoy.

El área en la que nos centraremos principalmente en este proyecto es la clasificación de objetos, que se encuentra en el campo de reconocimiento. La clasificación de objetos tiene como objetivo identificar efectivamente los objetos de dentro de las imágenes.

2.2. Introducción al Aprendizaje Automático

El Aprendizaje Automático (AA, o Machine Learning, por su nombre en inglés) es la rama de la Inteligencia Artificial que tiene como objetivo desarrollar técnicas que permitan a las computadoras aprender. De forma más concreta, se trata de crear algoritmos capaces de generalizar comportamientos y reconocer patrones a partir de una información suministrada en forma de ejemplos. Es, por lo tanto, un proceso de inducción del conocimiento, es decir, un método que permite obtener por generalización un enunciado general a partir de enunciados que describen casos particulares.

Cuando se han observado todos los casos particulares la inducción se considera completa, por lo que la generalización a la que da lugar se considera válida. No obstante, en la mayoría de los casos es imposible obtener una inducción completa, por lo que el enunciado a que da lugar queda sometido a un cierto grado de incertidumbre, y en consecuencia no se puede considerar como un esquema de inferencia formalmente válido ni se puede justificar empíricamente. En muchas ocasiones el campo de actuación del aprendizaje automático se solapa con el de Data Mining, ya que las dos disciplinas están enfocadas en el análisis de datos, sin embargo el aprendizaje automático se centra más en el estudio de la complejidad computacional de los problemas con la intención de hacerlos factibles desde el punto de vista práctico, no

únicamente teórico.

A un nivel muy básico, se puede decir decir que una de las tareas del AA es intentar extraer conocimiento sobre algunas propiedades no observadas de un objeto basándose en las propiedades que sí han sido observadas de ese mismo objeto (o incluso de propiedades observadas en otros objetos similares), predecir comportamiento futuro a partir de lo que ha ocurrido en el pasado. Un ejemplo de actualidad sería, el de predecir si un determinado producto le va a gustar a un cliente basándose en las valoraciones que ese mismo cliente ha hecho de otros productos que sí ha probado.

En cualquier caso, como el tema del que se está abordando está relacionado con el aprendizaje,¿Qué se entiende por aprender? y, ya que se quiere dar metodologías generales para producir un aprendizaje de forma automática, una vez que se fije este concepto se habrá de dar métodos para medir el grado de éxito o fracaso de un aprendizaje. En cualquier caso, se está trasladando un concepto intuitivo y que se usa normalmente en la vida diaria a un contexto computacional, ha de tenerse en cuenta que todas las definiciones de aprendizaje desde un punto de vista computacional, así como las diversas formas de medirlo, estarán íntimamente relacionadas con contextos muy concretos y posiblemente lejos de lo que intuitivamente, y de forma general, se entiende por aprendizaje.

Una definición relativamente general de aprendizaje dentro del contexto humano podría ser la siguiente: proceso a través del cual se adquieren o modifican habilidades, destrezas, conocimientos, conductas o valores como resultado del estudio, la experiencia, la instrucción, el razonamiento y la observación. De esta definición es importante hacer notar que el aprendizaje debe producirse a partir de la experiencia con el entorno, no se considera aprendizaje toda aquella habilidad o conocimiento que sean innatos en el individuo o que se adquieran como resultado del crecimiento natural de éste. Siguiendo un esquema similar, en el AA se considera aprendizaje a aquello que la máquina pueda aprender a partir de la experiencia, no a partir del reconocimiento de patrones programados a priori. Por tanto, una tarea central de cómo aplicar esta definición al contexto de la computación va a consistir en alimentar la experiencia de la máquina por medio de objetos con los que entrenarse (ejemplos) para, posteriormente, aplicar los patrones que haya reconocido sobre otros objetos

distintos.

Hay un gran número de problemas que caen dentro de lo que se le llama aprendizaje inductivo. La principal diferencia entre ellos estriba en el tipo de objetos que intentan predecir. Algunas clases habituales son:

Regresión

Intentan predecir un valor real. Por ejemplo, predecir el valor de la bolsa mañana a partir del comportamiento de la bolsa que está almacenado (pasado). O predecir la nota de un alumno en el examen final basándose en las notas obtenidas en las diversas tareas realizadas durante el curso.

Clasificación binaria o multi-clase

Intentan predecir la clasificación de objetos sobre un conjunto de clases prefijadas. Por ejemplo, clasificar si una determinada noticia es de deportes, entretenimiento, política, etc. Si solo se permiten 2 posibles clases, entonces se llama clasificación binaria; si se permiten más de 2 clases, estamos hablando de clasificación multi-clase.

Ranking

Intentar predecir el orden óptimo de un conjunto de objetos según un orden de relevancia predefinido. Por ejemplo, el orden en que un buscador devuelve recursos de internet como respuesta a una búsqueda de un usuario.

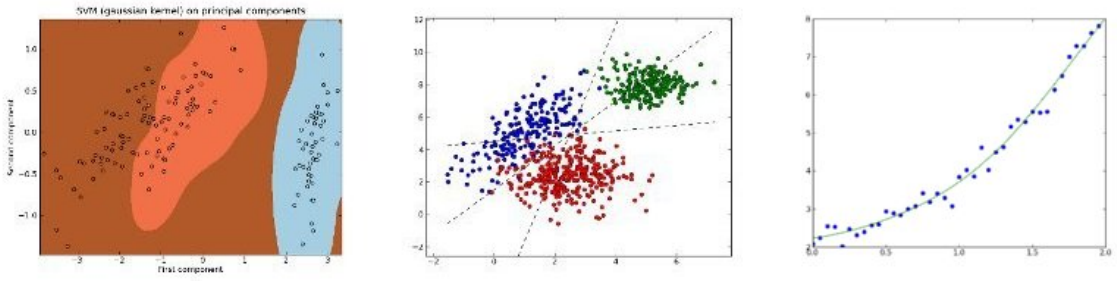


Figura 2.1: Ejemplo de ranking.

Normalmente, cuando se aborda un nuevo problema de AA lo primero que se hace es embarcarlo dentro de alguna de las clases anteriores, ya que dependiendo de cómo se clasifique será la forma en que se puede medir el error cometido entre la predicción y la realidad. En consecuencia, el problema de medir cómo de acertado es el aprendizaje obtenido deberá ser tratado para cada caso particular de metodología aplicada, aunque en general podemos adelantar que se necesitará embeber la representación del problema en un espacio en el que se tenga definida una medida.

Por otra parte, y dependiendo del tipo de salida que se produzca y de cómo se aborde el tratamiento de los ejemplos, los diferentes algoritmos de AA se pueden agrupar en:

Aprendizaje supervisado

Se genera una función que establece una correspondencia entre las entradas y las salidas deseadas del sistema, donde la base de conocimientos del sistema está formada por ejemplos etiquetados a priori (es decir, ejemplos de los que se sabe su clasificación correcta). Un ejemplo de este tipo de algoritmo es el problema de clasificación al que se ha hecho mención anteriormente.

Aprendizaje no supervisado

Donde el proceso de modelado se lleva a cabo sobre un conjunto de ejemplos formados únicamente por entradas al sistema, sin conocer su clasificación correcta. Por lo que se busca que el sistema sea capaz de reconocer patrones para poder etiquetar las nuevas entradas.

Aprendizaje semi-supervisado

Es una combinación de los dos algoritmos anteriores, teniendo en cuenta ejemplos clasificados y no clasificados.

Aprendizaje por refuerzo

En este caso el algoritmo aprende observando el mundo que le rodea y con un continuo flujo de información en las dos direcciones (del mundo a la máquina, y de

la máquina al mundo) realizando un proceso de ensayo-error, y reforzando aquellas acciones que reciben una respuesta positiva en el mundo.

Transducción

Es similar al aprendizaje supervisado, pero su objetivo no es construir de forma explícita una función, sino únicamente tratar de predecir las categorías en las que caen los siguientes ejemplos basándose en los ejemplos de entrada, sus respectivas categorías y los ejemplos nuevos al sistema. Es decir, estaría más cerca del concepto de aprendizaje supervisado dinámico.

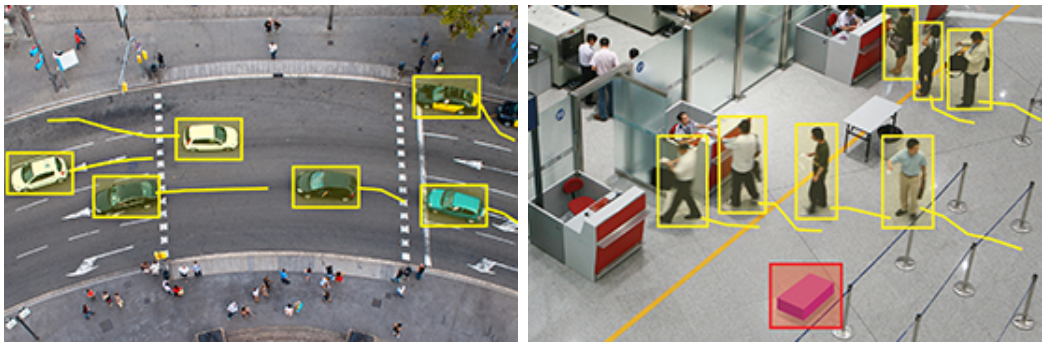
Aprendizaje multi-tarea

Engloba todos aquellos métodos de aprendizaje que usan conocimiento previamente aprendido por el sistema de cara a enfrentarse a problemas parecidos a los ya vistos.

2.3. Introducción a la Clasificación de Objetos

En este proyecto, el objetivo es implementar un clasificador de objetos siguiendo el método en una investigación adecuada en artículos de divulgación científica. Para hacer esto, primero se necesita introducir el concepto de clasificación de objetos y también proporcionar una descripción general del arte actual junto con ciertos trabajos de investigación en torno a áreas clave. Se prestará especial atención a qué métodos pueden ser más adecuados para reconocer imágenes no fotográficas. Se espera que el método Bolsa de palabras sea bastante efectivo para el problema en cuestión. Para probar esta hipótesis, se revisarán los métodos disponibles y se discutirá su idoneidad para esta tarea. Se empezará por definir qué es la clasificación de objetos.

En general, la clasificación de objetos es donde se toma y procesa una imagen o serie de imágenes. El clasificador debe generar etiquetas para los diferentes segmentos de las imágenes. Estas etiquetas contienen información sobre qué objeto es la imagen. Por ejemplo (véase en la Figura 2.2a), varias imágenes diferentes de coches deben todos recibir la misma etiqueta , incluso si se toman desde diferentes ángulos, en



(a) auto (b) Personas

Figura 2.2: Detección de objetos

diferentes condiciones de iluminación, o Incluso si los coches son de diferentes colores y modelos. La clasificación de objetos es un área muy importante de La visión por computadora, como lo ilustra la cantidad de investigación alrededor del tema. Tiene muchas aplicaciones Incluyendo robótica, búsqueda de imágenes, seguridad (Figura 2.2b específicamente reconocimiento facial), ayuda a personas con discapacidades visuales, censura de imágenes, entre otras más.

El campo de clasificación de objetos ha ido creciendo durante años con variadas técnicas que van desde lo completamente manual (Von Ahn and Dabbish, 2004) a totalmente automatizado (Sivic et al., 2005). En la siguiente sección,se analiza algunos de los enfoques totalmente automatizados y se considerarán sus fortalezas y debilidades.

Un punto clave en el reconocimiento de objetos es que los objetos que intentamos identificar son objetos reales, el primero El método que veremos aprovecha este hecho. Clasificación de objetos basada en CAD (Arman, 1993) utiliza no solo datos de imágenes sino también datos de rango para producir una vista tridimensional de un escena. Esto es muy útil ya que permite detectar formas e ignorar texturas. El rango de datos es utilizado para hacer un modelo parcial de la escena utilizando un software de diseño asistido por computadora (CAD), este modelo se puede generalizar con el resultado que coincide con una categoría, o el modelo puede ser emparejado contra una base de datos a un diseño específico. Este método puede ser muy efectivo y es comúnmente usado para el rango de detección en robótica, sin embargo, para la mayoría de las tareas, el rango de detección no es una opción, ya que una gran can-

tidad de clasificación de objetos se realiza en imágenes y fotografías que no siempre son en tiempo real, para el caso especial que se está considerando, este sistema no sería adecuado ya que la escritura a mano alzada es obviamente bidimensional.

Si bien los humanos tienen una percepción profunda, se cuenta con la capacidad de identificar objetos con un ojo cerrado, esto demuestra que existe una forma natural de identificar objetos sin necesidad de datos de rango. En su trabajo de investigación (Pentland, 1987) muestra que nuestra percepción del objeto está formada por estructuras parciales donde cada objeto se divide en partes que identificamos, y que mientras las características visuales ayudan a hacer esto, no son la única parte que se utiliza, también plantea la hipótesis de que parece que las características de la imagen por sí solas generalmente no puede apoyar el reconocimiento. Sin embargo, dada la efectividad de la Bolsa de Palabras, el método y otros enfoques basados en características se ha demostrado que es posible con la tecnología moderna Identificar objetos utilizando solo características. La forma en que están vinculadas y las estructura que forman podría ser útil para clasificar trazos a mano alzada, ya que las características individuales pueden no ser las mismas pero podría ser en la estructura general.

Otra importante área de visión utilizada por los humanos es el color. Si bien el color no es de ninguna manera esencial para identificar la mayoría de los objetos, puede proporcionar información valiosa sobre ellos. Citando a (Swain and Ballard, 1990) su trabajo en histogramas de color muestra cómo se puede usar la información de color para identificar objetos rápida y eficientemente. En este proyecto no se utilizarán las representaciones de color particularmente, en parte porque hay muchos métodos efectivos en escala de grises y porque los trazos a menudo son en negro Y fondo blanco.

Un método que se ha vuelto cada vez más popular para el reconocimiento de objetos es encontrar características visuales clave en las imágenes y compararlas con otras imágenes. La obra de (Turk and Pentland, 1991) en Eigenfaces for Recognition muestra una forma práctica en que esto puede ser aplicado. Dada la imagen de un rostro, resuelve las características visuales clave (no necesariamente los ojos, oídos, nariz) y crea un conjunto de vectores propios de definición para estas características. Los vectores propios pueden ser emparejados contra una base de datos de otros

conjuntos de vectores propios y cuando dos coinciden pueden asumir que las caras son las mismas.

También hay muchos otros métodos que utilizan características visuales, en las primeras etapas de la clasificación de objetos, la mayoría de los esfuerzos se centró en la detección y descripción de características. Esto se puede ver en investigación de (Harris and Stephens, 1988). Recientemente la investigación en clasificación de imágenes parece ser más en torno al área de probabilidad y encontrar formas más eficientes de comparar imágenes (Grauman and Darrell,). También hay bastante investigación para aplicaciones específicas, como la clasificación de escenas, que tiene usos en robótica (Lazebnik et al.,) e incluso trabajar con vídeos y categorizar acciones humanas (Niebles et al., 2006).

Un método ahora común de clasificación que usa características es el enfoque Bolsa de palabras. Sigue Algunos de los principios similares a los otros métodos con una diferencia importante, los datos de posición en relación con las características se ignora. Esta es una idea que fue usada por primera vez por (Leibe and Schiele, 2003) donde intentan resolver el problema del reconocimiento con un enfoque a partir de características visuales y no desde segmentación. Un problema que surge a menudo en la clasificación de objetos es clasificar un área no segmentada, pero también existe el problema de segmentar una imagen no clasificada. Leibe y Schiele intentan abordar la primera parte de este problema utilizando detectores de características y probabilidad con un método basado en el método de bolsa de palabras utilizado para clasificar la literatura. El método para identificar las características varían según la implementación, al igual que el modelo de probabilidad, sin embargo, el principio sigue siendo similar.

2.4. Introducción a el modelo Bolsa de palabras

El concepto de bolsa de palabras se diseñó originalmente para clasificar libros en el género correcto sin supervisión humana. En esencia, toma todas las palabras potencialmente significativas (es decir, excluye palabras comunes como en y la) y crea una bolsa de palabras. La bolsa de palabras no contiene información sobre el orden o la posición de las palabras en el libro, simplemente con qué frecuencia ocurre cada

una. Desde un grupo de datos de entrenamiento es posible calcular la probabilidad de que cada palabra aparezca en un género dado de en ese libro, estas probabilidades se pueden aplicar sobre toda la bolsa de palabras para calcular el género más probable del libro. Se demostró que esta técnica era muy exitosa en la identificación de la literatura.

Este concepto se puede aplicar a la clasificación de imágenes con solo cambios menores. La principal y la más obvia, es que las imágenes no contienen palabras, en lugar de eso necesitamos usar palabras visuales, estas apuntan a ser la sección significativa de la imagen. Para determinar si una sección de una imagen es significativa podemos usar diversas técnicas de filtrado lineal y detección de bordes, que ahora están bien documentadas en varios artículos y libros de texto. Las secciones significativas pueden ser desde simples esquinas hasta algo más específico como una rueda o una nariz. El siguiente problema es cómo comparar una sección con otra o como dos imágenes de una rueda pueden verse muy diferentes. Los descriptores se utilizan para describir la sección en el objetivo, mantener la esencia de la sección sin tener demasiados datos específicos, esto debería permitir cambios de perspectiva e iluminación, las diferentes técnicas para esto se explican en el capitulo 3. Una vez que la bolsa crea las palabras visuales y las palabras se comparan con las de los datos de entrenamiento, las probabilidades son calculadas y la categoría más probable se puede identificar, por ejemplo, si la imagen contiene dos ruedas y esquinas afiladas es más probable que sea una imagen de un automóvil que una cara. La razón por la que se decidió utilizar el método de la bolsa de palabras es porque ha sido demostrado ser extremadamente confiable y eficiente, especialmente con texto.

2.4.1. Descriptores

Los descriptores de características se utilizan para detectar y describir características locales de las imágenes. El objetivo de un descriptor es encontrar una característica de la imagen y describirla de una manera que no se vea afectada por la perspectiva, la escala, la oclusión o la iluminación. Uno de los métodos más comunes es la transformación de características invariantes de escala (SIFT), desarrollado por (Lowe, 1999), se considera una de los descriptores más sólidos. Más tarde se hizo una modificación a ese descriptor, descriptor de características robustas aceleradas

(SURF) desarrollado por (Bay and Van Gool, 2006) es un método inspirado en SIFT y que vive hasta su nombre, ya que se considera igual, si no más robusto que SIFT y es notablemente más eficiente (Mikolajczyk and Schmid,). Sin embargo, como SIFT es un método más probado y comprobado, y el código está disponible de forma gratuita, se utilizará como detector de características. Hay detectores más modernos. como PCA-SIFT (Ke and Sukthankar, 2004) que se basa en SIFT que tiene como objetivo mejorar su rendimiento, sin embargo, se dejarán pruebas de diferentes descriptores para trabajos futuros.

2.4.2. Probabilidad y modelos de entrenamiento

Calcular la probabilidad de que un segmento de imagen sea de un objeto específico es un paso crucial en el objeto. clasificación, requiere un gran conjunto de datos de entrenamiento con los que se pueden comparar las características de la imagen, las características se combinan con las categorías posibles y el tema puede predecirse. Hay varios modelos estadísticos comunes utilizados para esto, como los histogramas, aunque estos se consideran un Método menos eficiente (Grauman and Darrell,). También hay modelos estadísticos como el Análisis semántico latente probabilístico (pLSA) y la Asignación de Dirichlet Latente (LDA) como se explica por (Sivic et al., 2005), sin embargo estos son muy complejos. Comunmente se utiliza un clasificador de Naive Bayes (Lewis, 1998)

2.5. Evolución de la clasificación de objetos

En esta parte, se muestra el proceso de clasificación de objetos y su evolución hasta el método que se utiliza en la actualidad.

2.5.1. Esquema de la clasificación de objetos

El objetivo es que a partir de una imagen de entrada, al final se pueda asignar una determinada categoría, de entre las que se habrá aprendido a partir del conjunto de aprendizaje. El primer paso va a ser siempre lo que se conoce como extracción de características, permite obtener características visuales a partir de la imagen,

resumiendo este primer paso de extracción de características, se compone de dos subprocesos, la detección y la descripción de puntos de interés, y como resultado da un conjunto de puntos de interés en la imagen, que podrá ser variable de una imagen a otra, y una descripción en forma de vector numérico, para cada uno de estos puntos.

el segundo paso en el esquema será obtener una representación única de toda la imagen, en forma de vector numérico. Esta representación se obtendrá agregando o combinando la descripción de todos los puntos de interés. A partir de esta representación única de toda la imagen, el paso final va a ser el paso de clasificación propiamente dicho.

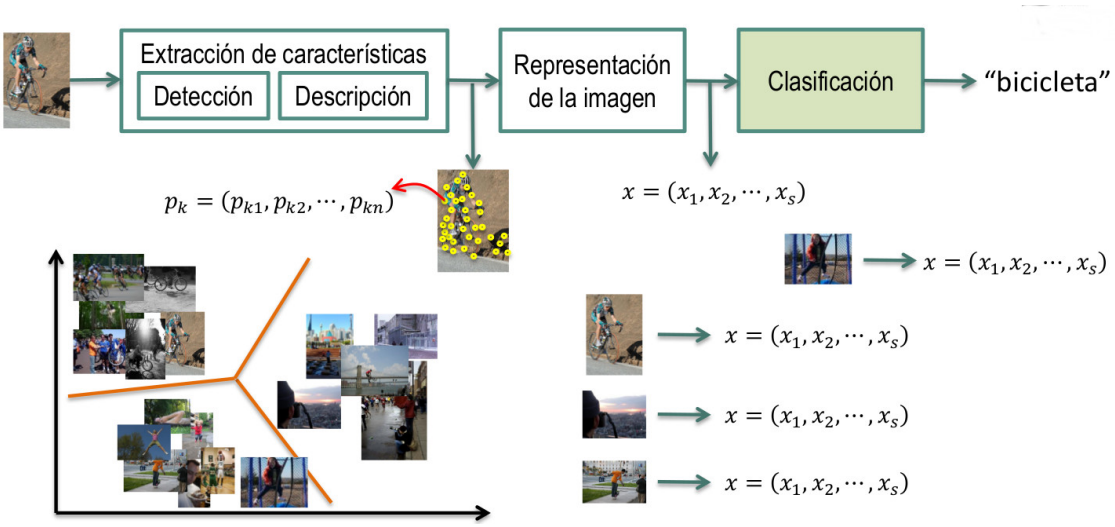


Figura 2.3: Proceso de clasificación de objetos.

La tarea del paso de clasificación va a ser, precisamente, encontrar la frontera óptima de separación entre las diferentes clases en este espacio multidimensional, siempre a partir del conjunto de aprendizaje. De esta forma, cuando se tenga una nueva imagen que se deba clasificar se obtendrá también la representación de la imagen, se coloca en el espacio del descriptor, y se espera que esté en lado de la frontera de la etiqueta que deberíamos darle; en este caso, bicicleta. Véase la Figura B.3

2.5.2. Métodos de clasificación

Se creó una tabla 2.1 donde se encuentra en orden los diferentes métodos del proceso de clasificación mostrado en la figura B.3, en la primera fila se muestra un sistema básico que cubrirá todos los pasos del esquema, de una forma simple. En la segunda fila se introduce una nueva forma de obtener la representación global de la imagen, es lo que se conoce como bag of words en inglés. Este método de representación ha sido la base estándar de representación de imágenes para clasificación a lo largo de los últimos 10 años, y será la representación sobre la que se basará la propuesta sobre el problema en cuestión, además, se introduce uno de los métodos de clasificación más utilizados, que se conoce como support vector machine.

En la tercera fila, se muestra diferentes alternativas para realizar el primer paso de detección y descripción de puntos de interés, incorporando también descriptores que permiten incorporar diferentes tipos de contenido visual, como por ejemplo la forma y el color. En la fila, y sobre la representación gráfica del bag of words, se proponen diferentes formas para describir los puntos de interés que se pueden combinar, a diferentes niveles, para mejorar el rendimiento del clasificador.

En la quinta fila, se extiende la representación básica del bag of words para superar una de sus limitaciones, ya que, bag of words no tiene en cuenta la información espacial de dónde está cada uno de los puntos de interés en la imagen. Únicamente tiene en cuenta su descripción. Es así que se incorpora esta información espacial a la representación con el método de pirámide espacial. Finalmente, en la última fila se hace mención a lo que se conoce como redes neuronales convolucionales, o CNNs. Que están adquiriendo relevancia en los 2, 3 últimos años, ya que permiten obtener unas tasas de clasificación muy elevadas. Aunque también hay que decir que el número de imágenes necesario y el coste del aprendizaje es muy elevado. Seguramente, las CNNs constituyen el futuro de la clasificación de imágenes.

Detector de características	**Descriptor de características**	**Representación de la imagen**	**Clasificación**
SIFT	SIFT	Conjunto de puntos	K-NN
		BoW Básico	SVM
Dense sampling, Harris, SURF	SURF, Descriptores de color, PCA		
	Fusión de características	Fusión de representaciones	Combinación de clasificadores
		Piramides espaciales	
CNN	CNN	GMM, Fisher Vector, VLAD	CNN

Cuadro 2.1: Métodos de clasificación.

Capítulo 3

Metodología

3.1. Algoritmo EM

Los algoritmos de Expectativa-Maximización (EM) son procedimientos para imitando una función LL (log-likelihood) cuando los procedimientos estándar son numéricamente difíciles o inviables. El procedimiento fue introducido por (Dempster et al., 1977) como una forma de manejar los datos faltantes. Sin embargo, es aplicable de manera mucho más general y se ha utilizado con éxito en muchos campos de estadística. McLachlan y Krishnan (Mclachlan and Krishnan, 2008) proporcionan una revisión de aplicaciones en el campo del modelado de elección discreta, algoritmos EM han sido utilizados por Bhat (Bhat, 1997) y Train (Train, 2008).

El procedimiento consiste en definir una expectativa particular y entonces maximizárla (de ahí el nombre). Esta expectativa está relacionada con La función LL (log-likehood) de una manera que se describirá más adelante, pero difiere de una manera que facilita la maximización. El procedimiento es iterativo, comenzando en algún valor inicial para los parámetros y actualizando los valores en cada iteración. Los parámetros actualizados en cada iteración son los valores a maximizar la expectativa en esa iteración particular. Como se mostrará, La maximización repetida de esta función converge al máximo de la función LL.

En este capítulo, se describe el algoritmo EM en general, se puede utilizar para estimar distribuciones muy flexibles, incluyendo especificidades no paramétricas. que pueden aproximarse asintóticamente a cualquier distribución.

3.1.1. Procedimiento general

En esta sección se describe el procedimiento de EM de una manera muy general. con el fin de dilucidar sus características. En las siguientes secciones, se aplica el procedimiento general a modelos específicos. Se deja que la variable dependiente observada se denotará colectivamente como y, representando las elecciones o secuencia de opciones para una muestra completa de tomadores de decisiones. Las elecciones dependen En las variables explicativas observadas que, por conveniencia de notación, no se denota explícitamente. Las opciones también dependen de los datos que están faltantes, denotados colectivamente como z. Dado que los valores de estos faltantes no se observa, el investigador especifica una distribución que representa los valores que los datos faltantes podrían tomar. Por ejemplo, si el ingreso o remuneración de algunos individuos de la muestra está faltante, la distribución de ingreso en la población puede ser una especificación útil para la distribución de los valores de ingresos faltantes. La densidad de los datos faltantes se denota $f(x|\theta)$, que depende en general de los parámetros θ a ser estimado.

El modelo de comportamiento relaciona los datos observados y faltantes con el elecciones u opciones. Este modelo predice las elecciones que surgirían si los los datos desaparecidos fueran observados en lugar de faltar. Este modelo de comportamiento se denota como $P(y|z,\theta)$ donde θ son parámetros que pueden superponerse o extender los de f. (Para la compacidad notacional, se usa θ para denotar todos los parámetros a estimar, incluidos los que entran f y los ingresandos P.) Como, sin embargo, faltan los datos faltantes, la probabilidad de las elecciones observadas, en base a la información que el investigador observa, es la integral de la probabilidad condicional sobre la densidad de los datos faltantes:

$$P(y|\theta) = \int P(y|z,\theta) f(z|\theta) dz$$

La densidad de los datos faltantes, $f(z|)$, se utiliza para predecir las opciones observadas y por lo tanto no depende de y . Sin embargo, se puede obtener información sobre los datos faltantes mediante la observación de las opciones que fueron hechas. Por ejemplo, en la elección de un vehículo, si falta el ingreso de una persona pero se observa que la persona compró un Mercedes, se infiere que es probable que los ingresos de esta persona estén por encima del promedio. se define $g(z|y,\theta)$ como la

densidad de los datos faltantes condicionales a las opciones observadas en la muestra. Esta densidad condicional está relacionada con la densidad incondicional a través de la identidad de Bayes:

$$h(z|y,\theta) = \frac{P(y|z,\theta)f(z|\theta)}{P(y|\theta)}$$

Dicho de manera sucinta: la densidad de z condicional en las elecciones observadas es proporcional a la densidad incondicional de z. Suponemos en esta expresión que z es continuo, de modo que lo incondicional a la probabilidad es una integral. Si z es discreto, o una mezcla de variables continuas y discretas, entonces la integración se reemplaza con una suma sobre los valores discretos, o una combinación de integrales y sumas.

De las elecciones observadas dada esta z. El denominador es simplemente la constante de normalización, igual a la integral del numerador. Esta el concepto de una distribución condicional. Ahora consideremos la estimación, la función log-verosimilitud se basa en La información que tiene el investigador, que no incluye los datos que faltan. La función LL es:

$$LL(\theta) = logP(y|\theta) = logP(\int(y|z,\theta)f(z|\theta))dz$$

A menudo es mucho más fácil maximizar LL de una manera diferente. El procedimiento es iterativo, comenzando con un valor inicial de parámetros y actualizándolos. Deja el valor de los parámetros en una iteración dada se denota θ^t. Definamos una nueva función en θ^t que se relaciona con LL pero utiliza el condicional densidad h. Esta nueva función es:

$$\epsilon(\theta|\theta^t) = \int h(z|y,\theta^t)logP(y|z,\theta)f(z|\theta)dz$$

donde la densidad condicional h se calcula utilizando la prueba actual del valor de los parámetros θ^t. Esta función tiene un significado específico tenga en cuenta que la parte del extremo derecho, $P(y|z,\theta)f(z|\theta)$, es la articulación probabilistica de las elecciones observadas y los datos faltantes. El registro de esta probabilidad conjunta es la probabilidad logarítmica de las elecciones observadas y los datos faltantes combinados. Esta probabilidad de registro conjunta se integra sobre un densidad, es decir, $h(z|y,t)$. Nuestra función ϵ es por lo tanto una expectativa. de la probabilidad conjunta de los datos faltantes y las opciones observadas. Eso es una expectativa

específica, es decir, la expectativa sobre la densidad de los datos faltantes condicionales a las elecciones observadas. Ya que la densidad condicional de z depende de los parámetros, esta densidad se calcula utilizando los valores θ^t. Dicho de manera equivalente, ϵ es el promedio ponderado de la articulación probabilidad logarítmica usando $h(z|y, \theta^t)$ como pesos.

El procedimiento EM consiste en maximizar repetidamente ϵ. Comenzando Con algún valor inicial de los parámetros, los parámetros se actualizan en cada iteración por la fórmula:

$$\theta^{t+1} = argmax\theta\epsilon(\theta|\theta^t) \quad (14.1)$$

En cada iteración, los valores actuales de los parámetros θ^t, se utilizan para calcular los pesos h , y luego la probabilidad de registro conjunta ponderada es maximizada. El nombre EM deriva del hecho de que el procedimiento Utiliza una expectativa que se maximiza. Es importante reconocer el doble papel de los parámetros en E. Primero, los parámetros ingresan a la probabilidad conjunta de las elecciones observadas. y los datos faltantes, $P(y|z,)f(z|\theta)$. En segundo lugar, los parámetros entran, la densidad condicional de los datos faltantes, $h(z|y,)$. La función E se maximiza con respecto a lo anterior manteniendo la constante posterior. Es decir, E se maximiza sobre θ ingresando $P(y|z, \theta)f(z|\theta)$, manteniendo presionado θ que ingresa los pesos $h(z|y, \theta)$ en sus valores actuales θ^t. Para denotar este doble papel, $\epsilon(\theta|\theta^t)$ se expresa en función de θ, su argumento sobre el que se realiza la maximización, dado θ, el valor se utiliza en los pesos que se mantienen fijos durante la maximización. En condiciones muy generales, las iteraciones definidas por la ecuación convergen al máximo de LL . Bolyes (Boyles, 1983) y Wu (Wu, 1983) proporcionan mayores pruebas.

3.1.2. Desarrollo del algoritmo EM

La relación del algoritmo EM con la función log-verosimilitud se puede explicar en tres pasos. Cada paso es un poco ambiguo, pero los tres combinados proporcionan una comprensión sorprendentemente intuitiva.

Primer paso

Ajustar ϵ igual a LL en θ^t

$\epsilon(\theta|\theta^t)$ No es lo mismo que $LL(\theta)$. Para facilitar la comparación entre ellos, se añade una constante a $\epsilon(\theta|\theta^t)$ que es igual a la diferencia entre las dos funciones en θ^t:

$$\epsilon * (\theta|\theta^t) = \epsilon(\theta|\theta^t) + [LL(\theta^t) - \epsilon(\theta^t|\theta^t)]$$

El término entre paréntesis es constante con respecto a θ y, por lo tanto, maximización de ϵ es lo mismo que la maximización de ϵ en sí. Sin embargo, por construcción es, $\epsilon * (\theta|\theta^t) = LL(\theta)$ en $\theta = \theta^t$

Segundo paso

Se tiene en cuenta que el θ derivada es el mismo para $\epsilon*$ y LL evaluado en $\theta = \theta^t$
La derivada de $\epsilon * (\theta|\theta^t)$ respecto a θ:

$$\begin{aligned}\tfrac{d\epsilon*(\theta|\theta^t)}{d\theta} &= \tfrac{d\epsilon(\theta|\theta^t)}{d\theta} \\ =&\int h(z|y,\theta^t)\tfrac{dlogP(y|z,\theta)f(z|\theta)}{d\theta}dz \\ =&\int h(z|y,\theta^t)\tfrac{1,dP(y|z,\theta)f(z|\theta)}{P(y|z,\theta),f(z|\theta)d\theta}dz.\end{aligned}$$

Ahora se calcula la derivada de $\theta = \theta^t$

$$\begin{aligned}&d\epsilon * (\theta|\theta^t)d\theta|\theta^t \\ =&h(z|y,\theta^t)\tfrac{1}{P(y|z,\theta^t f(z|\theta^t)}\tfrac{dP(y|z,\theta)f(z|\theta)}{d\theta}\theta^t dz \\ =&\int \tfrac{P(y|z,\theta^t)f(z|\theta^t)}{P(y|\theta^t)}\tfrac{1}{P(y|z,\theta^t)f(z|\theta^t)}\tfrac{dP(y|z,\theta)f(z|\theta)}{d\theta}\theta^t dz \\ =&\int \tfrac{1}{P(y|\theta^t)}\tfrac{dP(y|z,\theta)f(z|\theta)}{d\theta}\theta^t dz \\ =&\tfrac{1}{P(y|\theta^t)}\int \tfrac{dP(y|z,\theta)f(z|\theta)}{d\theta}\theta^t dz \\ =&\tfrac{dlogP(y|\theta)}{d\theta}d\theta \\ =&\tfrac{dLL(\theta)}{d\theta}\theta^t\end{aligned}$$

En $\theta = \theta^t$, las dos funciones, $\epsilon*$ y LL , tienen la misma pendiente

Tercer paso

Se tiene en cuenta que LL $\epsilon*$ es menor o igual a cero para todos.
Esta relación se puede mostrar de la siguiente manera:

$$
\begin{aligned}
& LL(\theta) \\
=& logP(y|\theta) \ (14.2) \\
=& \ log \int P(y|z,\theta) f(z|\theta) dz \\
=& \ log \int \tfrac{P(y|z,\theta) f(z|\theta)}{h(y|z,\theta^t)} h(y|z,\theta^t) dz \\
\geq& \int h(y|z,\theta^t) log \tfrac{P(y|z,\theta) f(z|\theta)}{h(y|z,\theta^t)} dz \ (14.3) \\
=& \int h(y|z,\theta^t) logP(y|z,\theta) f(z|\theta) dz - \int h(y|z,\theta^t) logh(y|z,\theta) dz \\
=& \epsilon(\theta|\theta^t) - \int h(y|z,\theta^t) logh(y|z,\theta^t) dz \\
=& \epsilon(\theta|\theta^t) - \int h(y|z,\theta^t) log(h(y|z,\theta^t \tfrac{P(y|\theta^t)}{P(y|\theta^t)}) dz \\
=& \epsilon(\theta|\theta^t) + \int h(y|z,\theta^t) logP(y|\theta^t) dz - \int h(y|z,\theta^t) logh((y|z,\theta^t) P(y|\theta^t)) dz \\
=& \epsilon(\theta|\theta^t) + logP(y|\theta^t) \int h(y|z,\theta^t) dz - \int h(y|z,\theta^t) logh((y|z,\theta^t) P(y|\theta^t)) dz \\
=& \epsilon(\theta|\theta^t) + logP(y|\theta^t) - \int h(y|z,\theta^t) log(h(y|z,\theta^t) P(y|\theta^t)) dz \ (14.4) \\
=& \epsilon(\theta|\theta^t) + LL(\theta^t) - \int h(y|z,\theta^t) logP((y|z,\theta^t) f(z|\theta^t)) dz \ (14.5) \\
=& \epsilon(\theta|\theta^t) + LL(\theta^t) - \epsilon(\theta^t|\theta^t) \\
=& \epsilon * (\theta|\theta^t)
\end{aligned}
$$

La desigualdad en la línea 14.3 se debe a la desigualdad de Jensen, que afirma ese registro $(\epsilon(x)) > \epsilon(registro(x))$. En este caso, x es la estadística. $P(y|z,)f(z|\theta)h(y|z,\theta^t)$ y la expectativa es sobre densidad $h(y|z,t)$. Un ejemplo de esto la desigualdad se muestra en la Figura 3.1, donde los promedios son más de dos valores etiquetados ayb . El promedio de $log(a)ylog(b)$ es el punto medio de la línea de puntos que conecta estos dos puntos en la curva de registro. El registro evaluado en el promedio de ayb es $log((a+b)/2)$, que está arriba del punto medio de la línea de puntos. La desigualdad de Jensen es simplemente un resultado de la forma cóncava de la función log. La línea 14.4 se obtiene porque la densidad h se integra a 1. Línea 14.5 se obtiene sustituyendo $h(y|z,\theta^t) = P(y|z,\theta^t)f(z|\theta^t)/P(y|*^t)$ dentro del registro y luego cancelando la $P(y|\theta^t)$.

3.1.3. Combinar resultados para comparar E* y LL

En la figura 3.2 se muestra $\epsilon * (\theta|\theta^t) y LL(\theta)$ en relación apropiada una con otra. Como se ha demostrado, estas dos funciones son iguales y tienen la misma pendiente en $\theta = \theta^t$. Estos resultados implican que las dos funciones son tangentes entre si en $\theta = \theta^t$. También se demuostra que $\epsilon * (\theta|\theta^t) \leq LL(\theta)$ para todos. De acuerdo con esta

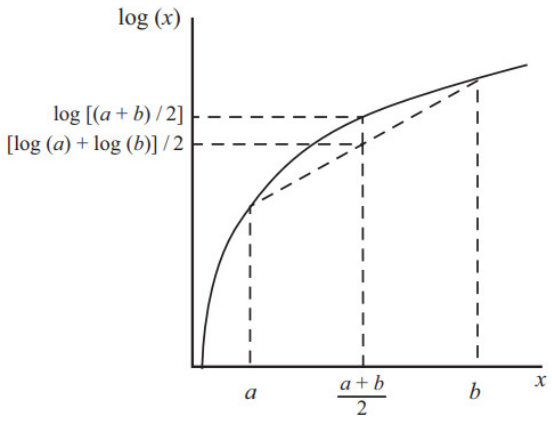


Figura 3.1: Ejemplo desigualdad de Jensen.

relación, ϵ se dibuja debajo de LL () en la figura 3.2 en todos los puntos excepto en donde son iguales.

El algoritmo EM maximiza $\epsilon * (\theta|\theta^t)$ para encontrar el siguiente valor de prueba de θ. El valor de maximización se muestra como $\theta^t + 1$. Como indica el gráfico, la función de probabilidad logarítmica es necesariamente más alta en el valor del parámetro nuevo, $\theta^t + 1$, que en el valor original, θ^t. Siempre que la derivada de la función log-verosimilitud no sea cero en θ^t, la maximización de $\epsilon * (\theta|\theta^t)$ aumenta LL (θ). Cada iteración del algoritmo EM eleva la función de probabilidad de registro hasta que el algoritmo converge al máximo de la función de probabilidad de registro.

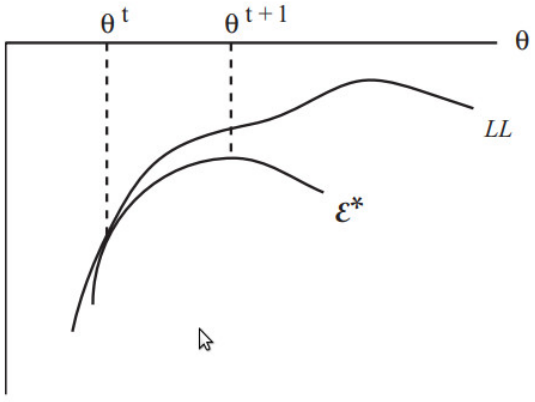


Figura 3.2: Relación entre ϵ y LL

3.1.4. Convergencia

La convergencia del algoritmo EM se define generalmente como un cambio suficientemente pequeño en los parámetros por ejemplo, (wil,) o en la función de probabilidad de registro por ejemplo (Weeks and Lange, 1989), y (Aitkin and Aitkin, 1996). Estos criterios deben usarse con cuidado, ya que el algoritmo EM puede moverse lentamente cerca de la convergencia. (Ruud, 1991) muestra que la estadística de convergencia se puede usar con el gradiente y la arpillera de E en lugar de LL. Sin embargo, el cálculo de esta estadística puede ser más intensivo en computación que la iteración del algoritmo EM en sí mismo, y en algunos casos puede ser inviable.

3.1.5. Errores estándar

Hay tres formas en que se pueden calcular los errores estándar. Primero, una vez que se ha encontrado el máximo de LL (θ) con el algoritmo EM, los errores estándar se pueden calcular a partir de LL de la misma manera que si la función log-likelihood se hubiera maximizado directamente. Los procedimientos son aplicables: Los errores estándar asintóticos pueden calcularse a partir de la arpillera o de la varianza de los gradientes específicos de observación (es decir, las puntuaciones), calculados a partir de LL (θ) evaluados a θ.

Una segunda opción surge del resultado que se obtuvo en el paso 2 anterior. Se demuestra que ϵ y LL tienen los mismos gradientes en $\theta = \theta^t$. En la convergencia, el valor de θ no cambia de una interacción a la siguiente, de manera que $\theta = \theta^t + 1 = \theta^t$. Por lo tanto, en, las derivadas de estas dos funciones son las mismas. Este hecho implica que las puntuaciones se pueden calcular a partir de E en lugar de LL. Si ϵ toma una forma más conveniente que LL, como suele ser el caso al aplicar un algoritmo EM, este El cálculo alternativo puede ser atractivo.

Una tercera opción es bootstrap, Bajo esta opción, el algoritmo EM se aplica varias veces, utilizando una muestra diferente de las observaciones cada vez. En muchos contextos en los que se aplican algoritmos EM, los errores estándar de arranque son más factibles y útiles que las fórmulas asintóticas.

3.2. Modelo bolsa de palabras visuales

El modelo bolsa de palabras (del inglés, Bag of Words) es un método que se utiliza en el procesado del lenguaje para representar documentos ignorando el orden de las palabras. En este modelo, cada documento parece una bolsa que contiene algunas palabras. Por lo tanto, este método permite un modelado de las palabras basado en diccionarios, donde cada bolsa contiene unas cuantas palabras del diccionario. En el campo de reconocimiento de objetos, se utiliza una idea similar para las representaciones de imágenes, es decir, una imagen puede ser tratada como un documento y las características extraídas de ciertos puntos de la imagen son consideradas palabras visuales. Las principales ventajas de utilizar este modelo es su facilidad de uso y su eficiencia computacional.

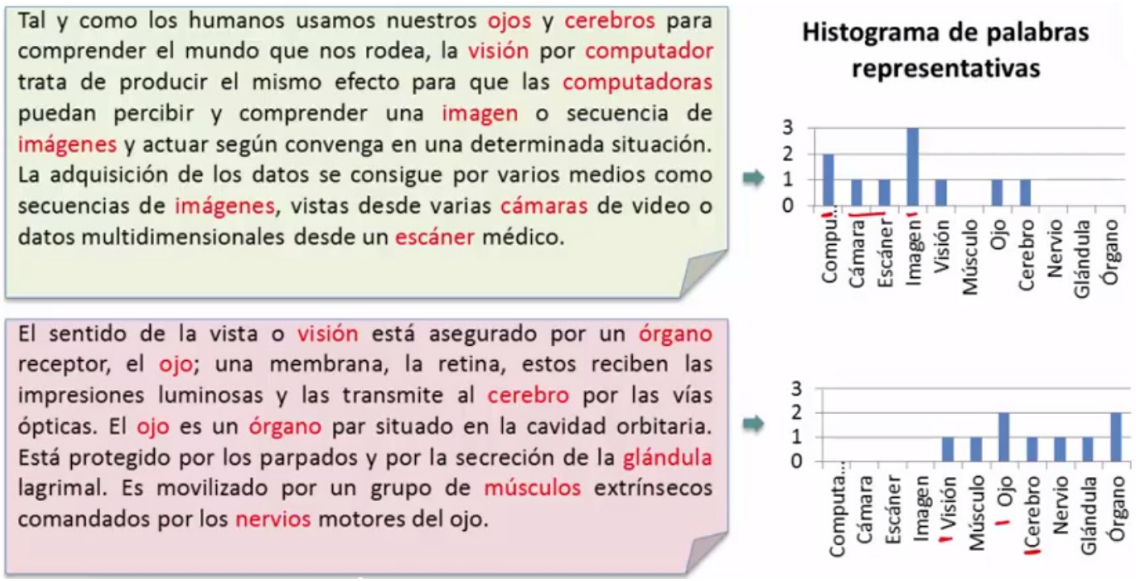


Figura 3.3: Origen de bag of words: clasificación de documentos

En el modelo de bag of visual words se requiere de un conjunto de datos para poder entrenar el algoritmo, pasará por tres principales etapas: la detección y descripción de las imágenes, la agrupación de datos y generación de un diccionario de palabras y por ultimo la clasificación por medio de un algoritmo especializado.

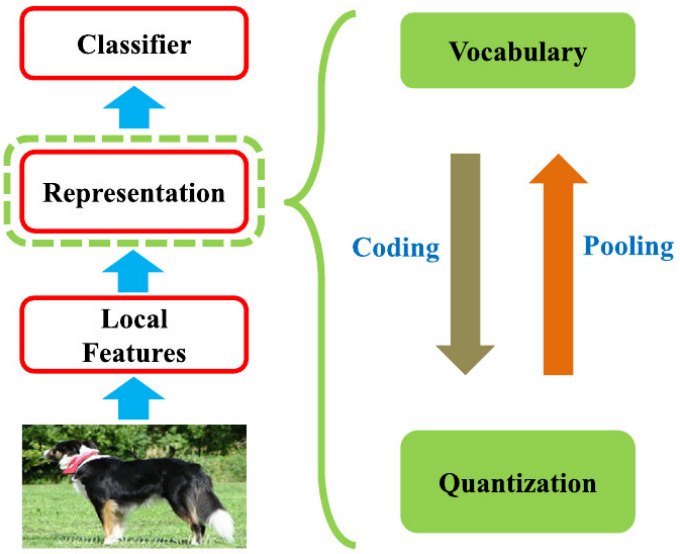


Figura 3.4: Diagrama de flujo de bag of visual words

3.2.1. Conjunto de datos

Es necesario tener un conjunto de datos para entrenar el modelo, los más utilizados en visión computacional y procesamiento de imágenes son: 15 Escenas de (Lazebnik et al.,), Caltech 101 utilizado en un proyecto de (Niebles et al., 2006), Caltech 256 por (lah,), PASCAL VOC utilizado en diferentes aspectos de la visión por computadora como lo muestra (Everingham et al., 2009). Estos conjuntos de datos se dividen en dos partes, generalmente 80 % para entrenamiento y 20 % para prueba de clasificación.

3.2.2. Detección y descripción de imagen

Se utilizará el algoritmo de Scale-invariant feature transform (o SIFT) que es un algoritmo usado en visión artificial para extraer características relevantes de las imágenes que posteriormente pueden usarse en reconocimiento de objetos, detección de movimiento, registro de la imagen y otras tareas. El algoritmo fue publicado por primera vez por (Lowe, 1999) pero lo describió completamente y patentó en Estados Unidos en 2004.

Detección de extremos en el espacio de escala

En el primer paso el algoritmo busca regiones de interés sobre todas las localizaciones de las imágenes a diferentes escalas. Es implementado eficientemente usando una función de diferencia de Gaussianas para identificar puntos de interés potenciales que son invariantes a la escala y a la orientación. En la Figura 3.5, se muestra cómo para cada octava del espacio escala, la imagen inicial es repetidamente convolucionada con Gaussianas para producir el conjunto de imágenes espacio escala mostrado en la izquierda. Las imágenes Gaussianas adyacentes son substraídas para producir las imágenes diferencia de Gaussiana (DoG) de la derecha. Después de cada octava, la imagen Gaussiana es submuestreada por un factor de 2, y el proceso es repetido. El objetivo de la detección es encontrar extremos en el espacio DoG, que se corresponderán con puntos de interés (keypoints).

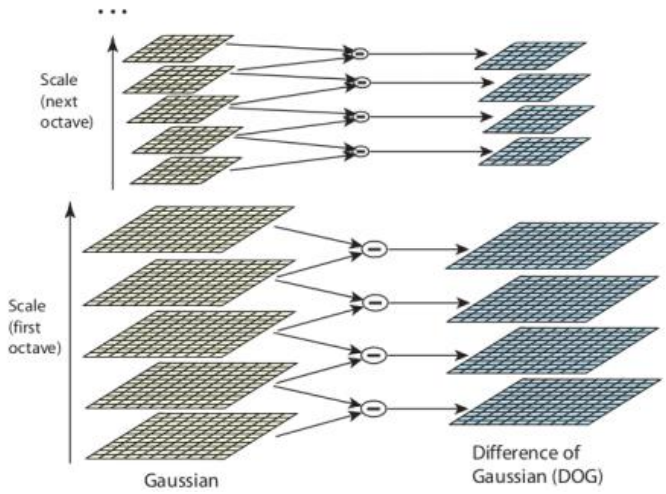


Figura 3.5: Representación del proceso que sigue cada octava del espacio escala

Localización de puntos clave

A cada localización candidata, un modelo detallado se ajusta para determinar la localización y la escala. Los puntos clave son seleccionados basados en medidas de su estabilidad. En la Figura 3.6 se muestran, a) la imagen original, b) los 832 puntos clave originales al máximo y mínimo de la función Diferencia de Gaussianas, c) 729 puntos clave restantes tras aplicar un umbral con un mínimo contraste, y d) los 536

puntos clave finales que quedan siguiendo un umbral adicional en proporción a las principales curvaturas.

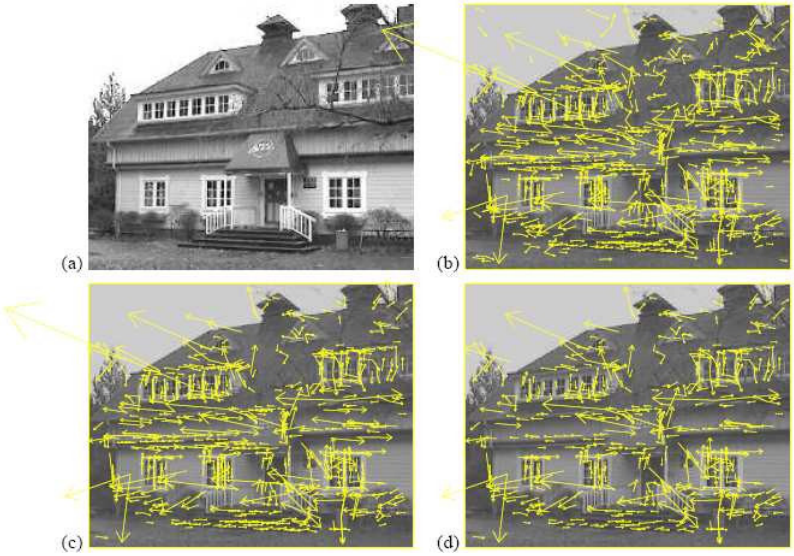


Figura 3.6: Fases de selección de puntos clave

Asignación de la orientación

A cada localización de puntos clave se le asigna una o más orientaciones basadas en direcciones de gradientes de imagen local. Todas las operaciones futuras son realizadas en los datos de imagen que han sido transformados en relación a la orientación asignada, escala y localización para cada característica, proporcionando así invarianza a dichas transformaciones. En la Ilustración 3.7 se muestran tres líneas: la primera representa el porcentaje de localizaciones de puntos clave y escalas que son detectados repetidamente como una función de ruido de píxel, la segunda línea muestra la repetitividad después de también requerir un acuerdo en orientación, y la última línea indica el porcentaje final de descriptores correspondidos correctamente en una gran base de datos.

Descriptor de puntos clave

Los gradientes de imagen locales son medidos a la escala seleccionada en la región alrededor de cada punto clave. Estos son transformados en una representación que

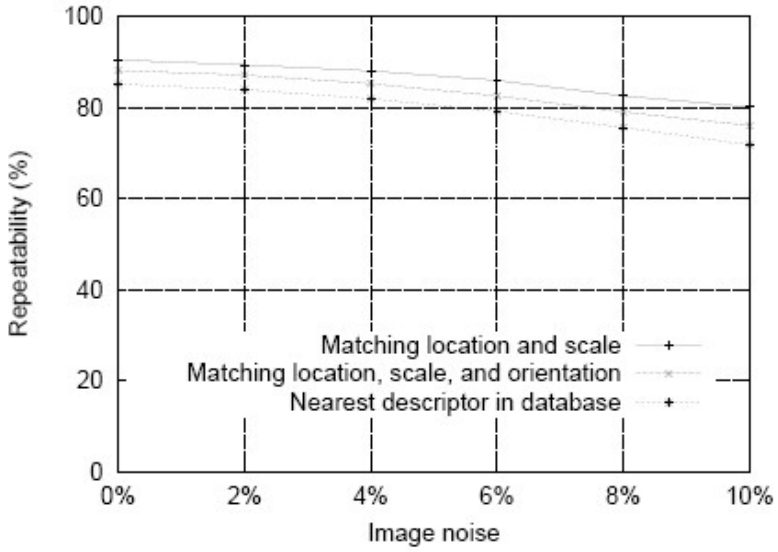


Figura 3.7: Gráfico Repetitividad-Ruido de imagen

tiene en cuenta significativos niveles de distorsión de forma local así como cambios en la iluminación. En la Figura 3.8 se puede ver cómo un descriptor de puntos clave es creado calculando la transformada SIFT. Inicialmente, como se ve en la imagen de la izquierda, se calcula la magnitud del gradiente y la orientación de cada punto en una región alrededor de la localización del punto clave. A estos se les asigna un peso con una ventana Gaussiana, indicado por un círculo superpuesto (según el peso asignado, el círculo tendrá un radio mayor o menor). Estas muestras son después acumuladas en histogramas de orientaciones, resumiendo los contenidos sobre subregiones resultantes de una división de la región en un grid 4x4, como se muestra en la imagen derecha. La longitud de cada flecha corresponde a la suma de las magnitudes de gradiente en dicha dirección dentro de la región. Esta figura muestra un array de descriptores 2x2 calculado en la subregión correspondiente.

Esta aproximación ha sido llamada la transformación de características invariante a la escala (SIFT), y transforma los datos de la imagen en coordenadas invariantes a la escala relativas a las características locales.

Un aspecto importante de esta aproximación es que genera un gran número de características que cubre densamente la imagen sobre el rango completo de escalas y localizaciones. Una imagen típica de 500x500 píxeles de tamaño ocasionará unas

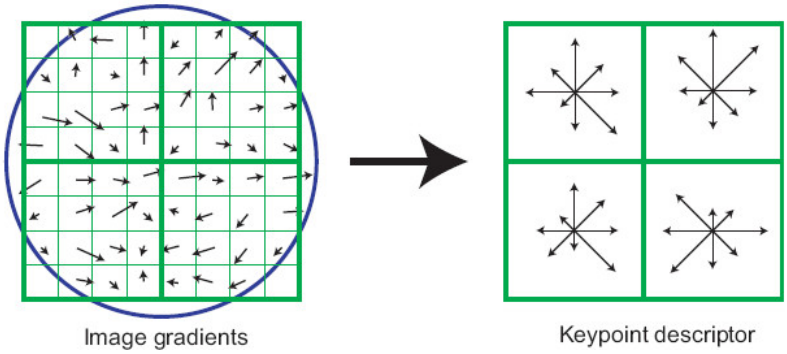


Figura 3.8: Gradientes de la imagen y descriptor de puntos clave.

2000 características estables (aunque ese número depende tanto del contenido de la imagen como de las elecciones de varios parámetros). La cantidad de parámetros es particularmente importante para el reconocimiento de objetos, donde la habilidad para detectar objetos pequeños en fondos abarrotados requiere que al menos 3 características sean correctamente correspondidas para cada objeto para una identificación fiable.

Para correspondencia y reconocimiento de imágenes, las características SIFT son primero extraídas de un conjunto de imágenes de referencia y almacenadas en una base de datos. Una nueva imagen es correspondida individualmente comparando cada característica suya con esa base de datos previa y encontrando un candidato igualando las características basadas en la distancia euclídea de sus vectores de características.

Descripción

Después de la detección de características, cada imagen es representada a través de sus parches locales. Los métodos de representación tratan de describir los parches como vectores numéricos, llamados descriptores de características. Un buen descriptor debe tener la habilidad de manejar la intensidad, rotación, escala y variaciones afines de la misma dimensión (128 para SIFT por ejemplo), cuando el orden de los diferentes vectores no importa.

3.2.3. Construcción de vocabulario

Una vez todas las regiones de interés de las imágenes han sido descritas, se genera un vocabulario (codebook) representativo de las características que aparecen en los datos. Para ello, a través de métodos de agrupamiento (no supervisados) se organizan los datos en ciertos grupos que se corresponden con palabras visuales (codewords) de dicho vocabulario.

Cada palabra visual, por lo tanto, constituirá la caracterización (ya sea a través del color, textura u otra descripción) de un patrón visual. Posteriormente, cada descripción asociada a un punto de interés (keypoint) en una imagen se proyectará sobre el vocabulario asignándosele la palabra más parecida. En la práctica, para generar codebooks se emplean algoritmos de agrupamiento bien conocidos y simples, como el algoritmo k-means, dada la gran cantidad de datos y la elevada dimensión de los mismos.

La generación de codebooks tiene un doble objetivo: por un lado permite reducir la dimensionalidad de los datos de entrada (128 en el caso de emplear descriptores SIFT, por ejemplo) al asignar cada descriptor a un único codeword y, en conjunción con el empleo de otras técnicas como la creación de histogramas normalizados o los modelos generativos de bag-of-words, permite clasificar imágenes indexadas a través de un número variable de descriptores locales (al ser variable este número, la mera concatenación de descriptores no es posible). Por otro lado, generar clasificadores a nivel de keypoint no es factible, pues el etiquetado de imágenes se hace a nivel global, no formando parte todos los keypoints del objeto de interés (el hecho de que una imagen se catalogue como coche no implica que todos los keypoints pertenezcan al coche).

Algoritmo de agrupamiento K-means

K-medias es un método de agrupamiento, que tiene como objetivo la partición de un conjunto de n observaciones en k grupos en el que cada observación pertenece al grupo cuyo valor medio es más cercano. Es un método utilizado en minería de datos.

El problema es computacionalmente difícil (NP-hard). Sin embargo, hay eficientes heurísticas que se emplean comúnmente y convergen rápidamente a un óptimo local. Estos suelen ser similares a los algoritmos expectation-maximization de mezclas de

distribuciones gausianas por medio de un enfoque de refinamiento iterativo empleado por ambos algoritmos. Además, los dos algoritmos usan los centros que los grupos utilizan para modelar los datos, sin embargo k-medias tiende a encontrar grupos de extensión espacial comparable, mientras que el mecanismo expectation-maximization permite que los grupos tengan formas diferentes.

El algoritmo más común utiliza una técnica de refinamiento iterativo. Debido a su ubicuidad a menudo se llama el algoritmo k-medias, también se le conoce como algoritmo de Lloyd, sobre todo en la comunidad informática.

Dado un conjunto inicial de k centroides m1(1),. . . ,mk(1), el algoritmo continúa alternando entre dos pasos:

Paso de asignación: Asigna cada observación al grupo con la media más cercana (es decir, la partición de las observaciones de acuerdo con el diagrama de Voronoi generado por los centroides).

$$S_1^t = X_p||X_p - m_i^t|| \leq ||X^p - m_j^t|| \bigtriangledown 1 \leq j \leq k$$

Paso de actualización: Calcular los nuevos centroides como el centroide de las observaciones en el grupo.

$$m_i^{t+1} = \frac{1}{|S_i^t|} \sum_{Xj \epsilon S_i^t} Xj$$

El algoritmo se considera que ha convergido cuando las asignaciones ya no cambian.

Los métodos de inicialización de Forgy (Hamerly and Elkan, 2002) y Partición Aleatoria son comúnmente utilizados. El método Forgy elige aleatoriamente k observaciones del conjunto de datos y las utiliza como centroides iniciales. El método de partición aleatoria primero asigna aleatoriamente un clúster para cada observación y después procede a la etapa de actualización, por lo tanto calcular el clúster inicial para ser el centro de gravedad de los puntos de la agrupación asignados al azar. El método Forgy tiende a dispersar los centroides iniciales, mientras que la partición aleatoria ubica los centroides cerca del centro del conjunto de datos. Según Hamerly y compañía, el método de partición aleatoria general, es preferible para los algoritmos tales como los k-medias armonizadas y fuzzy k-medias. Para expectation maximization y el algoritmo estándar el método de Forgy es preferible.

Demostración del algoritmos estándar K means

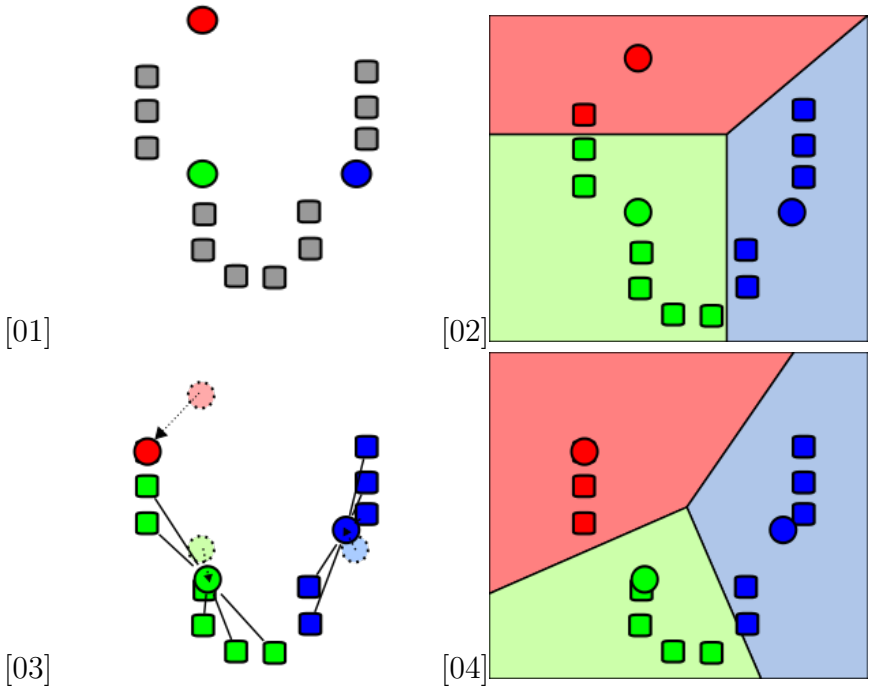


Figura 3.9: Procedimiento Kmeans

01: k centroides iniciales (en este caso k=3) son generados aleatoriamente dentro de un conjunto de datos (mostrados en color). 02: k grupos son generados asociándole el punto con la media más cercana. La partición aquí representa el diagrama de Voronoi generado por los centroides. 03: EL centroide de cada uno de los k grupos se recalcula. 04: Pasos 2 y 3 se repiten hasta que se logre la convergencia.

Como se trata de un algoritmo heurístico, no hay ninguna garantía de que convergen al óptimo global, y el resultado puede depender de los grupos iniciales. Como el algoritmo suele ser muy rápido, es común para ejecutar varias veces con diferentes condiciones de partida. Sin embargo, en el peor de los casos, k-medias puede ser muy lento para converger: en particular, se ha demostrado que existen conjuntos de determinados puntos, incluso en 2 dimensiones, en la que k-medias toma tiempo exponencial, es decir 2O(n), para converger. Estos conjuntos de puntos no parecen surgir en la práctica: esto se ve corroborado por el hecho de que en la mayoría de los casos el tiempo de ejecución de k-medias es polinomial.

3.2.4. Representación de la imagen

Si bien existen multitud de técnicas para generar la entrada a nivel de imagen, la más común es el histograma normalizado de codewords. Para generarlo, se contabiliza la aparición de las distintas palabras del vocabulario a lo largo de la imagen y se normaliza finalmente entre el número total de palabras encontradas. Así, con independencia del número de palabras presentes en cada imagen, todos los histogramas tendrán la misma longitud (el tamaño del codebook) y cumplirán que la suma de los valores de todas sus barras será igual a 1. Se puede destacar los pasos:

- Asignar cada característica local a la palabra visual más cercana
- Acumular el número de características asignadas a cada palabra visual
- Necesidad de comparar cada característica con todas las palabras

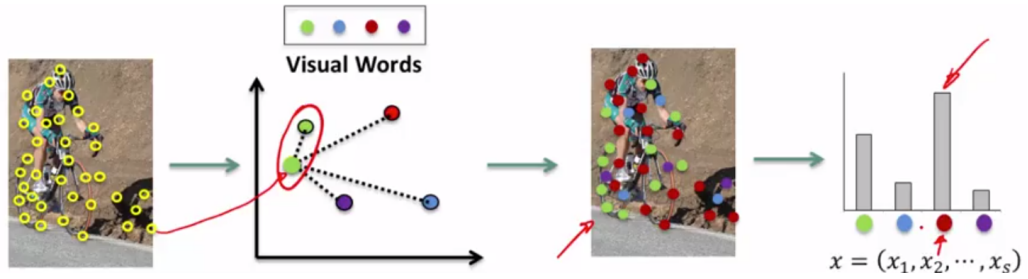


Figura 3.10: Construcción del histograma

3.2.5. Clasificación

El aprendizaje máquina trata el diseño y desarrollo de algoritmos que permite a los ordenadores mejorar su rendimiento a la hora de analizar datos procedentes de diversas fuentes, como los de un sensor o los de una base de datos. Un mayor enfoque en la investigación del aprendizaje máquina produce modelos, reglas y patrones de los datos. Como los conjuntos de entrenamiento son finitos, la teoría de aprendizaje normalmente no da garantías absolutas en el rendimiento de los algoritmos En este proyecto se van a utilizar 7 tipos de algoritmos de aprendizaje máquina: SVM, Gaussian NB, KNN, Decision Tree, Random Forest, Neural Network y AdaBoost

SVM

Las máquinas de vectores de soporte (SVM) son un conjunto de métodos de aprendizaje supervisado que se utilizan para la clasificación, regresión y detección de valores atípicos.

Las ventajas de las máquinas de vectores de soporte son:

- Eficaz en espacios de alta dimensión.
- Aún efectivo en casos donde el número de dimensiones es mayor que el número de muestras.
- Utiliza un subconjunto de puntos de entrenamiento en la función de decisión (llamados vectores de soporte), por lo que también es eficiente en memoria.
- Versátil: se pueden especificar diferentes funciones del Kernel para la función de decisión. Se proporcionan núcleos comunes, pero también es posible especificar núcleos personalizados.

Las desventajas de las máquinas de vectores de soporte incluyen:

- Si la cantidad de funciones es mucho mayor que la cantidad de muestras, evitar la adaptación excesiva en la elección de las funciones del Kernel y el término de regularización es crucial.
- Los SVM no proporcionan directamente estimaciones de probabilidad, se calculan utilizando una costosa validación cruzada de cinco veces.

Estos métodos están propiamente relacionados con problemas de clasificación y regresión. Dado un conjunto de ejemplos de entrenamiento (de muestras) podemos etiquetar las clases y entrenar una SVM para construir un modelo que prediga la clase de una nueva muestra. Intuitivamente, una SVM es un modelo que representa a los puntos de muestra en el espacio, separando las clases a 2 espacios lo más amplios posibles mediante un hiperplano de separación definido como el vector entre los 2 puntos, de las 2 clases, más cercanos al que se llama vector soporte. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, en función de los espacios a los que pertenezcan, pueden ser clasificadas a una o la otra clase.

Más formalmente, una SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta (o incluso infinita) que puede ser utilizado en problemas de clasificación o regresión. Una buena separación entre las clases permitirá una clasificación correcta.

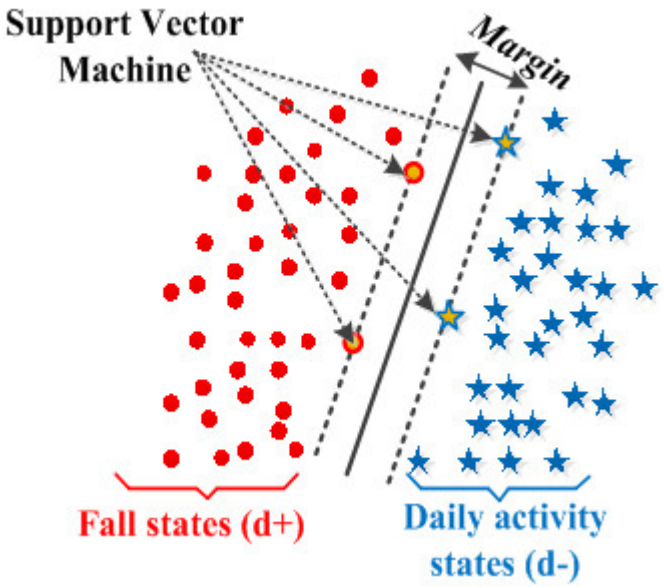


Figura 3.11: Clasificación SVM

Bayes ingenuo

En términos simples, un clasificador de Bayes ingenuo asume que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, dada la clase variable. Por ejemplo, una fruta puede ser considerada como una manzana si es roja, redonda y de alrededor de 7 cm de diámetro. Un clasificador de Bayes ingenuo considera que cada una de estas características contribuye de manera independiente a la probabilidad de que esta fruta sea una manzana, independientemente de la presencia o ausencia de las otras características.

Para otros modelos de probabilidad, los clasificadores de Bayes ingenuo se pueden entrenar de manera muy eficiente en un entorno de aprendizaje supervisado. En muchas aplicaciones prácticas, la estimación de parámetros para los modelos Bayes

ingenuo utiliza el método de máxima verosimilitud, en otras palabras, se puede trabajar con el modelo ingenuo de Bayes sin aceptar probabilidad bayesiana o cualquiera de los métodos bayesianos.

Una ventaja del clasificador de Bayes ingenuo es que solo se requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros (las medias y las varianzas de las variables) necesarias para la clasificación. Como las variables independientes se asumen, solo es necesario determinar las varianzas de las variables de cada clase y no toda la matriz de covarianza.

KNN

El método de los k vecinos más cercanos (en inglés, k-nearest neighbors, abreviado k-nn es un método de clasificación supervisada (Aprendizaje, estimación basada en un conjunto de entrenamiento y prototipos) que sirve para estimar la función de densidad $F(x/Cj)$ de las predictoras x por cada clase C_j.

Este es un método de clasificación no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad a posteriori de que un elemento x pertenezca a la clase C_j a partir de la información proporcionada por el conjunto de prototipos. En el proceso de aprendizaje no se hace ninguna suposición acerca de la distribución de las variables predictoras.

En el reconocimiento de patrones, el algoritmo k-nn es usado como método de clasificación de objetos (elementos) basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos. k-nn es un tipo de aprendizaje vago (lazy learning), donde la función se aproxima solo localmente y todo el cómputo es diferido a la clasificación.

El ejemplo que se desea clasificar es el círculo verde. Para k = 3 este es clasificado con la clase triángulo, ya que hay solo un cuadrado y 2 triángulos, dentro del círculo que los contiene. Si k = 5 este es clasificado con la clase cuadrado, ya que hay 2 triángulos y 3 cuadrados, dentro del círculo externo.

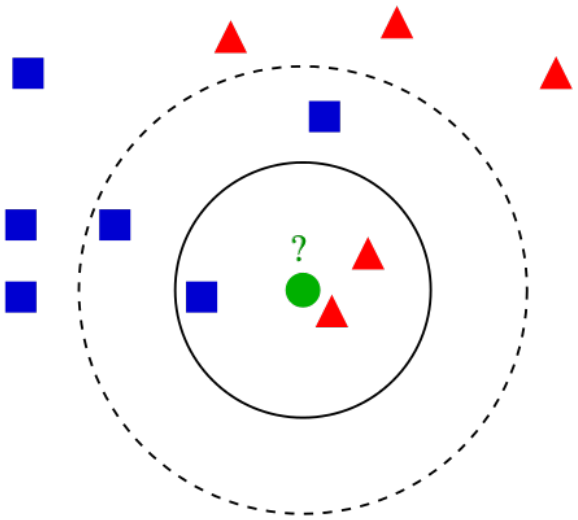


Figura 3.12: Ejemplo del algoritmo Knn.

Árbol de decisión

Un árbol de decisión1 es un modelo de predicción utilizado en diversos ámbitos que van desde la inteligencia artificial hasta la Economía. Dado un conjunto de datos se fabrican diagramas de construcciones lógicas, muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que ocurren de forma sucesiva, para la resolución de un problema.

Los árboles de decisión están formados por nodos, vectores de números, flechas y etiquetas.

- Cada nodo se puede definir como el momento en el que se ha de tomar una decisión de entre varias posibles, lo que va haciendo que a medida que aumenta el número de nodos aumente el número de posibles finales a los que puede llegar el individuo. Esto hace que un árbol con muchos nodos sea complicado de dibujar a mano y de analizar debido a la existencia de numerosos caminos que se pueden seguir.

- Los vectores de números serían la solución final a la que se llega en función de las diversas posibilidades que se tienen, dan las utilidades en esa solución.

- Las flechas son las uniones entre un nodo y otro y representan cada acción distinta.

- Las etiquetas se encuentran en cada nodo y cada flecha y dan nombre a cada acción.

En los árboles de decisión se tiene que cumplir una serie de reglas.

- Al comienzo del juego se da un nodo inicial que no es apuntado por ninguna flecha, es el único del juego con esta característica.
- El resto de nodos del juego son apuntados por una única flecha.
- De esto se deduce que hay un único camino para llegar del nodo inicial a cada uno de los nodos del juego. No hay varias formas de llegar a la misma solución final, las decisiones son excluyentes.

En los árboles de decisiones las decisiones que se eligen son lineales, a medida que vas seleccionando entre varias opciones se van cerrando otras, lo que implica normalmente que no hay marcha atrás. En general se podría decir que las normas siguen una forma condicional: Opción 1-opción 2-opción 3-Resultado Final X Estas reglas suelen ir implícitas en el conjunto de datos a raíz del cual se construye el árbol de decisión.

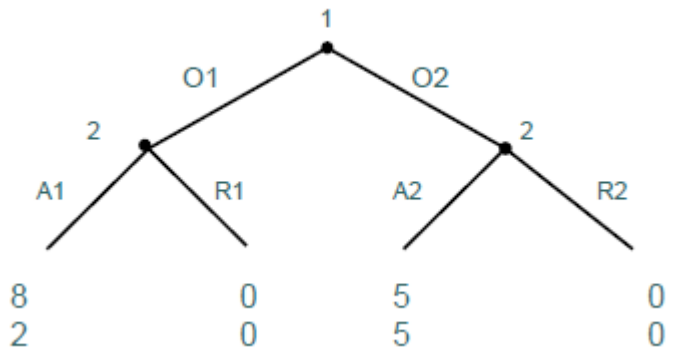


Figura 3.13: Ejemplo de un árbol de decisión

Random forest

Random forest (o random forests) también conocidos en castellano como 'Bosques Aleatoriosés una combinación de árboles predictores tal que cada árbol depende

de los valores de un vector aleatorio probado independientemente y con la misma distribución para cada uno de estos. Es una modificación sustancial de bagging que construye una larga colección de árboles no correlacionados y luego los promedia.

El algoritmo para inducir un random forest fue desarrollado por Leo Breiman1 y Adele Cutler y Random forests es su marca de fábrica. El término aparece de la primera propuesta de Random decision forests, hecha por Tin Kam Ho de Bell Labs en 1995. El método combina la idea de bagging de Breiman y la selección aleatoria de atributos, introducida independientemente por Ho, Amit y Geman, para construir una colección de árboles de decisión con variación controlada.

La selección de un subconjunto aleatorio de atributos es un ejemplo del método random subspace, el que, según la formulación de Ho, es una manera de llevar a cabo la discriminación estocástica propuesta por Eugenio Kleinberg.

En muchos problemas el rendimiento del algoritmo random forest es muy similar a la del boosting, y es más simple de entrenar y ajustar. Como consecuencia, el Random forest es popular y ampliamente utilizado.

Cada árbol es construido usando el siguiente algoritmo:

- Sea N el número de casos de prueba, M es el número de variables en el clasificador.
- Sea m el número de variables de entrada a ser usado para determinar la decisión en un nodo dado; m debe ser mucho menor que M
- Elegir un conjunto de entrenamiento para este árbol y usar el resto de los casos de prueba para estimar el error.
- Para cada nodo del árbol, elegir aleatoriamente m variables en las cuales basar la decisión. Calcular la mejor partición del conjunto de entrenamiento a partir de las m variables.

Para la predicción un nuevo caso es empujado hacia abajo por el árbol. Luego se le asigna la etiqueta del nodo terminal donde termina. Este proceso es iterado por todos los árboles en el ensamblado, y la etiqueta que obtenga la mayor cantidad de incidencias es reportada como la predicción.

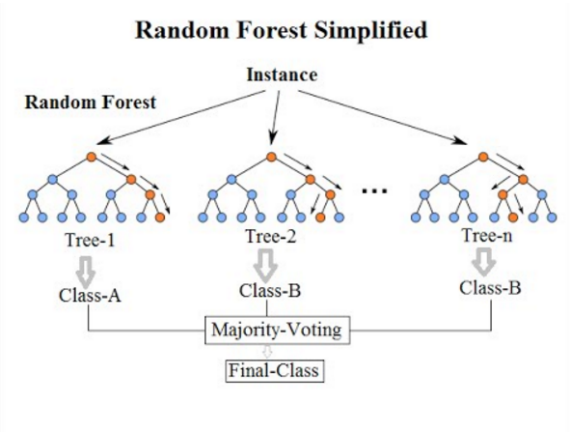


Figura 3.14: Ejemplo de random forest

Red neuronal

Las redes neuronales (también conocidas como sistemas conexionistas) son un modelo computacional basado en un gran conjunto de unidades neuronales simples (neuronas artificiales) de forma aproximadamente análoga al comportamiento observado en los axones de las neuronas en los cerebros biológicos. La información de entrada atraviesa la red neuronal (donde se somete a diversas operaciones) produciendo unos valores de salida.

Cada neurona está conectada con otras a través de unos enlaces. En estos enlaces el valor de salida de la neurona anterior es multiplicado por un valor de peso. Estos pesos en los enlaces pueden incrementar o inhibir el estado de activación de las neuronas adyacentes. Del mismo modo, a la salida de la neurona, puede existir una función limitadora o umbral, que modifica el valor resultado o impone un límite que se debe sobrepasar antes de propagarse a otra neurona. Esta función se conoce como función de activación.

Estos sistemas aprenden y se forman a sí mismos, en lugar de ser programados de forma explícita, y sobresalen en áreas donde la detección de soluciones o características es difícil de expresar con la programación convencional. Para realizar este aprendizaje automático, normalmente, se intenta minimizar una función de pérdida que evalúa la red en su total. Los valores de los pesos de las neuronas se van actualizando buscando reducir el valor de la función de pérdida. Este proceso se realiza mediante la propagación hacia atrás.

El objetivo de la red neuronal es resolver los problemas de la misma manera que el cerebro humano, aunque las redes neuronales son más abstractas. Los proyectos de redes neuronales modernos suelen trabajar desde unos miles a unos pocos millones de unidades neuronales y millones de conexiones que, si bien son muchas órdenes, siguen siendo de una magnitud menos compleja que la del cerebro humano, más bien cercana a la potencia de cálculo de un gusano.

Nuevas investigaciones sobre el cerebro a menudo estimulan la creación de nuevos patrones en las redes neuronales. Un nuevo enfoque está utilizando conexiones que se extienden mucho más allá y capas de procesamiento de enlace en lugar de estar siempre localizado en las neuronas adyacentes. Otra investigación está estudiando los diferentes tipos de señal en el tiempo que los axones se propagan, como el aprendizaje profundo, interpola una mayor complejidad que un conjunto de variables booleanas que son simplemente encendido o apagado.

Las redes neuronales se han utilizado para resolver una amplia variedad de tareas, como la visión por computador y el reconocimiento de voz, que son difíciles de resolver usando la ordinaria programación basado en reglas. Históricamente, el uso de modelos de redes neuronales marcó un cambio de dirección a finales de los años ochenta de alto nivel, que se caracteriza por sistemas expertos con conocimiento incorporado en si entonces las reglas, a bajo nivel de aprendizaje automático, caracterizado por el conocimiento incorporado en los parámetros de un modelo cognitivo con algún sistema dinámico.

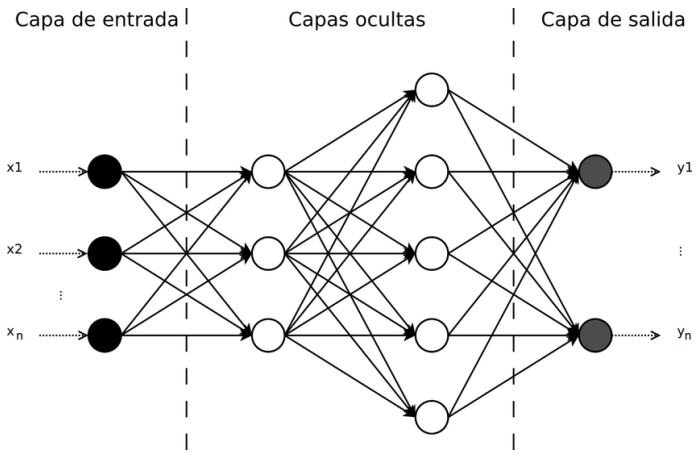


Figura 3.15: Red neuronal

AdaBoost

AdaBoost, abreviatura de Adaptive Boosting, es un meta-algoritmo de aprendizaje automático formulado por Yoav Freund y Robert Schapire, quienes ganaron el Premio Gödel 2003 por su trabajo. Puede usarse junto con muchos otros tipos de algoritmos de aprendizaje para mejorar el rendimiento. La salida de los otros algoritmos de aprendizaje (aprendices débiles) se combina en una suma ponderada que representa la salida final del clasificador potenciado. AdaBoost es adaptable en el sentido de que los estudiantes débiles subsiguientes se ajustan a favor de aquellos casos clasificados erróneamente por clasificadores anteriores. AdaBoost es sensible a datos ruidosos y valores atípicos. En algunos problemas, puede ser menos susceptible al problema de sobrealimentación que otros algoritmos de aprendizaje. Los aprendices individuales pueden ser débiles, pero mientras el desempeño de cada uno sea ligeramente mejor que las conjeturas aleatorias, se puede probar que el modelo final converge en un aprendiz fuerte.

Cada algoritmo de aprendizaje tiende a adaptarse a algunos tipos de problemas mejor que a otros, y generalmente tiene muchos parámetros y configuraciones diferentes para ajustar antes de lograr un rendimiento óptimo en un conjunto de datos, AdaBoost (con árboles de decisión como los aprendices débiles) a menudo se conoce como la mejor salida clasificador de la caja. Cuando se utiliza con el aprendizaje del árbol de decisión, la información recopilada en cada etapa del algoritmo de AdaBoost sobre la "durezarelativa de cada muestra de entrenamiento se introduce en el algoritmo de crecimiento de árboles, de modo que los árboles posteriores tienden a centrarse en ejemplos más difíciles de clasificar.

Los problemas en el aprendizaje automático a menudo sufren la maldición de la dimensionalidad: cada muestra puede consistir en un gran número de características potenciales (por ejemplo, puede haber 162,336 características de Haar, como las utiliza el marco de detección de objetos Viola-Jones, en un formato 24x24 de imagen de píxeles), y evaluar cada característica puede reducir no solo la velocidad de entrenamiento y ejecución del clasificador, sino también reducir el poder predictivo, según el efecto Hughes. A diferencia de las redes neuronales y los SVM, el proceso de capacitación de AdaBoost selecciona solo aquellas características conocidas para mejorar el poder predictivo del modelo, reduciendo la dimensionalidad y potencialmente

mejorando el tiempo de ejecución, ya que no es necesario calcular las características irrelevantes.

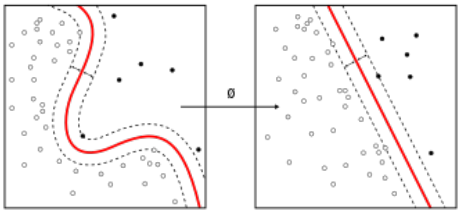


Figura 3.16: Separación de datos

3.2.6. Evaluación

Es indispensable evaluar el rendimiento de los diferentes algoritmos estudiados en el proyecto, para así poder compararlos y utilizar el óptimo en cada caso. Para este proyecto es necesario introducir los conceptos de precision-recall y la medida F así como la matriz de confusión.

Precisión

La precisión se mide calculando la suma de la diagonal de la matriz, que representa las imágenes correctamente clasificadas entre el número total de imágenes en la matriz. Esta tabla muestra el promedio de la precisión de todas las clases o categorías encontradas en los conjuntos de datos que representa la calidad de la respuesta del clasificador.

$$Precision = \frac{TP}{TP+FP}$$

Recall

La sensibilidad mide la eficiencia en la clasificación de todos los elementos de la clase mediante el cálculo de los positivos reales entre la suma de los positivos reales y los falsos positivos.

$$Recall = \frac{TP}{TP+FN}$$

F1 score

El puntaje de F1 se puede interpretar como un promedio ponderado de la precisión y la sensibilidad, donde un puntaje de F1 alcanza su mejor valor en 1 y el peor puntaje en 0.

$$F1Score = \frac{2*(Recall*Precision)}{(Recall+Precision)}$$

Matriz de confusión

En el campo de la inteligencia artificial una matriz de confusión es una herramienta que permite la visualización del desempeño de un algoritmo que se emplea en aprendizaje supervisado. Cada columna de la matriz representa el número de predicciones de cada clase, mientras que cada fila representa a las instancias en la clase real. Uno de los beneficios de las matrices de confusión es que facilitan ver si el sistema está confundiendo dos clases.

		Predicción	
		Positivos	**Negativos**
Observaciones	**Positivos**	Verdaderos positivos (VP)	Falsos negativos (FN)
	Negativos	Falsos positivos (FP)	Verdaderos negativos (VN)

Cuadro 3.1: Matriz de confusión.

Validación cruzada

La validación cruzada o cross-validation es una técnica utilizada para evaluar los resultados de un análisis estadístico y garantizar que son independientes de la partición entre datos de entrenamiento y prueba. Consiste en repetir y calcular la media aritmética obtenida de las medidas de evaluación sobre diferentes particiones. Se utiliza en entornos donde el objetivo principal es la predicción y se quiere estimar la precisión de un modelo que se llevará a cabo a la práctica. Es una técnica muy utilizada en proyectos de inteligencia artificial para validar modelos generados.

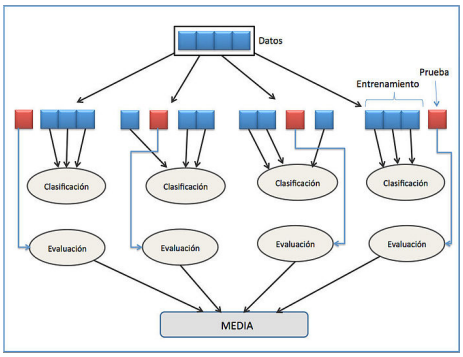


Figura 3.17: Esquema k-fold cross validation, con k=4 y un solo clasificador.

3.3. Lenguaje Python

Es un lenguaje de programación versátil multiplataforma y multiparadigma que se destaca por su código legible y limpio. Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License, que es compatible con la licencia pública general de GNU, se emplea en plataformas de alto tráfico como Google, YouTube o Facebook. Su objetivo es la automatización de procesos para ahorrar tanto complicaciones como tiempo, los dos pilares en cualquier tarea profesional. Dichos procesos se reducirán en pocas líneas de código que se insertan en una variedad de plataformas y sistemas operativos.

Phyton es ideal para trabajar con grandes volúmenes de datos porque favorece su extracción y procesamiento, siendo el elegido por las empresas de Big Data. A nivel científico, posee una amplia biblioteca de recursos con especial énfasis en las matemáticas para aspirantes a programadores en áreas especializadas. También es útil para crear videojuegos gracias a su dinamismo y simplicidad, aunque tratándose de un lenguaje de programación interpretado es más lento que Java, C++ o C#.

3.3.1. Historia

Python fue creado a finales de los ochenta por Guido van Rossum en el Centro para las Matemáticas y la Informática (CWI, Centrum Wiskunde Informatica), en los Países Bajos, como un sucesor del lenguaje de programación ABC, capaz de manejar excepciones e interactuar con el sistema operativo Amoeba.

El nombre del lenguaje proviene de la afición de su creador por los humoristas

británicos Monty Python. Actualmente, la evolución del lenguaje Python es gestionada por la Python Software Foundation, una sociedad sin ánimo de lucro dedicada a dar difusión al lenguaje y apoyar su evolución. Guido sigue totalmente involucrado en el desarrollo y en la toma de decisiones de diseño. Python está licenciado bajo licencia PSFL, derivada de BSD y compatible con GPL. Muchas empresas y organizaciones, como Google, Microsoft o Red Hat, hacen un gran uso de Python y tienen influencia en su evolución.

3.3.2. Características

Python es un lenguaje multiparadigma, esto significa que combina propiedades de diferentes paradigmas de programación. Principalmente es un lenguaje orientado a objetos, todo en Python es un objeto, pero también incorpora aspectos de la programación imperativa, funcional, procedural y reflexiva.

Una de las características más reseñables de Python es que es un lenguaje interpretado, esto significa que no se compila a diferencia de otros lenguajes como Java o C/C++, sino que es interpretado en tiempo de ejecución. Además, es de tipado dinámico, aunque opcionalmente desde la versión 3.5 podemos hacer uso de tipado estático.

Python es cross plataforma, es decir, se puede ejecutar en diferentes sistemas operativos como Windows o Linux simplemente usando el intérprete correspondiente. Se ha demostrado que es más lento en tiempo de ejecución que otros lenguajes compilados como Java o C/C++. Y es cierto, al tratarse de un lenguaje interpretado, sin embargo, las diferencias en velocidad son pequeñas y hoy en día el cuello de botella en los proyectos de desarrollo de software no está en la CPU. Gracias a avances como la computación en la nube se dispone de gran capacidad de cómputo a un coste muy asequible. El desafío está en acortar los tiempos de desarrollo, mejorando la mantenibilidad y calidad del código.

Scripting

Tradicionalmente Python ha tenido un uso muy extendido como herramienta de scripting, sustituyendo a scripts escritos en bash, otros lenguajes de script más

limitados o herramientas como AWK o sed. Por ello, Python ha sido adoptado por administradores de sistemas y equipos de operaciones.

Hoy en día, muchas de las herramientas punteras para gestión de despliegues e infraestructura usan o se basan en Python. Algunas de las más destacadas son Ansible, Salt o Fabric. Otra área en la que Python es pionero es en el mundo del scraping y el crawling, donde se extrae información de páginas web gracias a técnicas de "scraping", herramientas de Python como Scrapy son muy usadas en este contexto.

Desarrollo web

Otro de los campos en los que Python ha brillado en los últimos años es en el desarrollo de aplicaciones web, principalmente gracias a frameworks de desarrollo web muy potentes como Django, un framework completo o Flask, un microframework.

Sin embargo, en el ecosistema de desarrollo web existen muchas alternativas y frameworks muy maduros y asentados como Symfony para PHP, Spring para Java, Grails para Groovy o Rails para Ruby. Todos estos frameworks están continuamente tomando ideas entre ellos, inmersos en ofrecer las mejores alternativas para los desarrolladores. En este caso la ventaja que aporta Django, el principal framework para desarrollo web en Python, es la de ofrecer un marco de trabajo completo y de calidad para desarrollar aplicaciones web muy rápido. Como su leitmotiv dice es: "el framework para perfeccionistas con fechas de entrega".

Big Data, Data Science, AI

Sin embargo, al margen de todas las bondades que hemos comentado del lenguaje, en los últimos años ha ocurrido algo que ha revolucionado y extendido radicalmente el uso de Python. La generalización del Big Data en los últimos años, seguida de la explosión de la Inteligencia Artificial, Machine Learning, Deep Learning y el surgimiento de la ciencia de datos o data science como un nuevo área de trabajo con especialistas propios, ha revolucionado el panorama. Las nuevas herramientas que han surgido, y que son explotadas por ingenieros de datos y los científicos de datos, han sido desarrolladas en Python u ofrecen Python como la forma predilecta de interactuar con ellas.

Existe tecnología para Big Data como PySpark, de herramientas para Data Science como Pandas, NumPy, Matplotlib o Jupyter. De herramientas del procesamiento del lenguaje natural como NLTK, y por último el área de machine learning que tanto interés está despertando con herramientas como Tensorflow, MXNet o scikit-learn.

3.3.3. Tendencia

El crecimiento en el uso del lenguaje está siendo espectacular debido a las nuevas tecnologías de Data Science y Machine Learning, donde junto con el lenguaje R son pioneros. Sin embargo, R es un lenguaje más de nicho que proviene del mundo de la estadística. Python, por otro lado, es un lenguaje de propósito general y su uso está mucho más extendido.

En la siguiente gráfica 3.18 se muestra una proyección para los próximos años de Stackoverflow sobre el número de visitas que espera recibir en función de los principales lenguajes de programación.

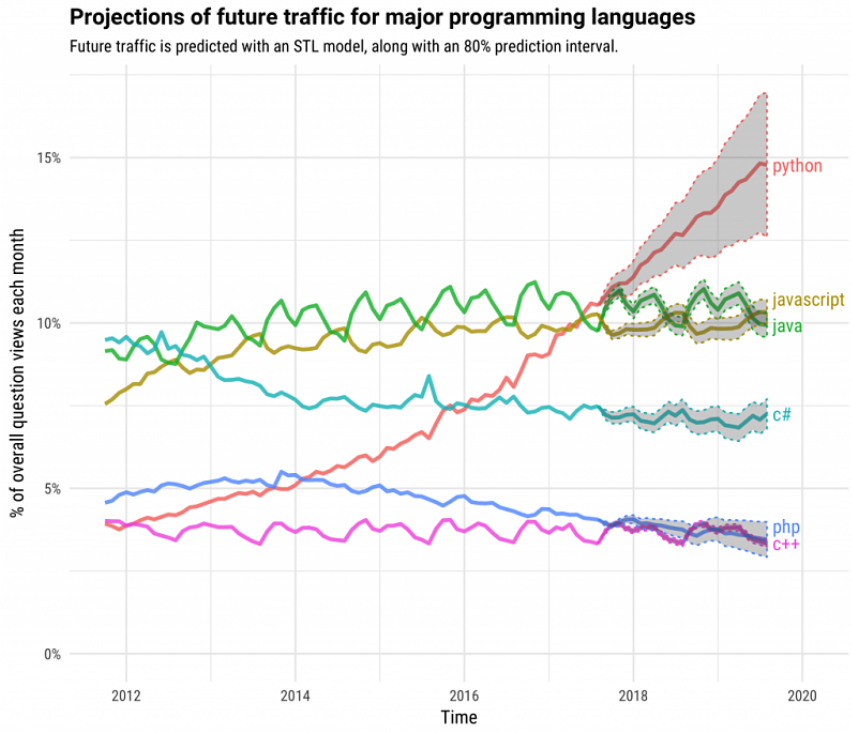


Figura 3.18: Proyección a futuro de los leguajes de programación con más tráfico.

3.4. Librería OpenCV

Desarrollada por Intel, OpenCV es una biblioteca libre de visión artificial que desde 1999 se viene empleando en todo tipo de aplicaciones que requieren incorporar el reconocimiento de objetos. Es así como sus más de 7 millones de descargas, revelan la trascendencia de sus más de 2.500 algoritmos ya que son estos los encargados de hacer posible encontrar imágenes similares, identificar rostros, redes neuronales artificiales, soporte de maquinas vectoriales, calibrar cámaras, clasificar acciones humanas en vídeo y extraer modelos 3D entre muchas otras cosas más.

3.4.1. Historia

OpenCV se inició en Intel en 1999 por Gary Bradsky y el primer lanzamiento salió en 2000. Vadim Pisarevsky se unió a Gary Bradsky para administrar el equipo OpenCV del software ruso de Intel. En 2005, OpenCV se utilizó en Stanley, el vehículo que ganó el Gran Desafío DARPA 2005. Más tarde, su desarrollo activo continuó bajo el apoyo de Willow Garage, con Gary Bradsky y Vadim Pisarevsky liderando el proyecto. En este momento, OpenCV admite muchos algoritmos relacionados con la Visión por Computador y el Aprendizaje Automático y se está expandiendo día a día.

Actualmente, OpenCV admite una amplia variedad de lenguajes de programación como C ++, Python, Java, etc. y está disponible en diferentes plataformas, incluyendo Windows, Linux, OS X, Android, iOS, etc. Además, las interfaces basadas en CUDA y OpenCL también están en desarrollo activo para alta velocidad de las operaciones de GPU. OpenCV-Python es la API de Python de OpenCV. Combina las mejores cualidades de OpenCV C ++ API y el lenguaje Python.

3.4.2. Características

La biblioteca cuenta con más de 2500 algoritmos optimizados, que incluye un conjunto completo de algoritmos de aprendizaje por ordenador y de visión artificial tanto clásicos como de vanguardia. Estos algoritmos se pueden usar para detectar y reconocer rostros, identificar objetos, clasificar acciones humanas en vídeos, rastrear movimientos de cámara, rastrear objetos en movimiento, extraer modelos 3D

de objetos, producir nubes de puntos 3D desde cámaras estéreo, unir imágenes para producir una alta resolución imagen de una escena completa, encuentre imágenes similares de una base de datos de imágenes, elimine los ojos rojos de las imágenes tomadas con flash, siga los movimientos de los ojos, reconozca paisajes y establezca marcadores para superponerlos con realidad aumentada, etc. OpenCV tiene más de 47 mil personas de usuarios Comunidad y número estimado de descargas que superan los 14 millones. La biblioteca se utiliza ampliamente en grupos de investigación, organismos gubernamentales junto con empresas bien establecidas como Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota que emplean la biblioteca, hay muchas empresas nuevas como Applied Minds, VideoSurf y Zeitera, que hacen un uso extensivo de OpenCV.

Tiene interfaces C ++, Python, Java y MATLAB y es compatible con Windows, Linux, Android y Mac OS. OpenCV se inclina principalmente hacia aplicaciones de visión en tiempo real y aprovecha las instrucciones MMX y SSE cuando están disponibles. Actualmente se están desarrollando activamente las interfaces CUDA y OpenCL. Hay más de 500 algoritmos y aproximadamente 10 veces más funciones que componen o admiten esos algoritmos. OpenCV está escrito de forma nativa en C ++ y tiene una interfaz de plantilla que funciona perfectamente con los contenedores STL.

3.4.3. OpenCV-Python

Python es un lenguaje de programación de propósito general iniciado por Guido van Rossum, que se hizo muy popular en poco tiempo principalmente debido a su simplicidad y legibilidad de código. Permite al programador expresar sus ideas en menos líneas de código sin reducir la legibilidad.

En comparación con otros lenguajes como C / C ++, Python es más lento. Pero otra característica importante de Python es que se puede extender fácilmente con C / C ++. Esta característica ayuda a escribir códigos de computación intensiva en C / C ++ y crear un contenedor de Python para que se pueda utilizar envoltorios como módulos de Python. Esto brinda dos ventajas: primero, el código es tan rápido como el código original de C / C ++ (ya que es el código real de C ++ que funciona en segundo plano) y, segundo, es muy fácil de codificar en Python. Así es como funciona

OpenCV-Python, es un envoltorio de Python alrededor de la implementación original de C ++.

Con el apoyo de Numpy hace que la tarea sea más fácil. Numpy es una biblioteca altamente optimizada para operaciones numéricas. Da una sintaxis de estilo MATLAB. Todas las estructuras de matrices de OpenCV se convierten y forman matrices de Numpy. Entonces, independientemente de las operaciones que se realizan en Numpy, pueden ser combinadas con OpenCV. Además de eso, varias otras bibliotecas como SciPy, Matplotlib que soporta Numpy pueden usarse. Por lo tanto, OpenCV-Python es una herramienta adecuada para la creación rápida de prototipos de problemas de visión de computadora.

Capítulo 4

Propuesta

En esta sección se presenta el trabajo que se realizó para construir un clasificador de imágenes, el método que se utilizó, así como las librerías, algoritmos y conjunto de imágenes que se emplearon.

4.1. Modelo BOVW

Como se ha demostrado y desarrollado el modelo de bag of visual words en el capítulo anterior, se muestra un diagrama de flujo con los algoritmos que se emplearán para este clasificador de imágenes de mano alzada.

	Conjunto de imágenes	**Detección y Descripción de imágenes**	**Algoritmo de agrupamiento**	**Clasificador**
	Caltech 101 Figuras Números Símbolos	SIFT	EM Kmeans Kmeans+EM	SVM, KNN ANN, RF RT, Adaboost NBayes
Bibliotecas empleadas	Numpy cPickle	OpenCV CV2	Sklearn Scipy	Sklearn

Cuadro 4.1: Panorama general

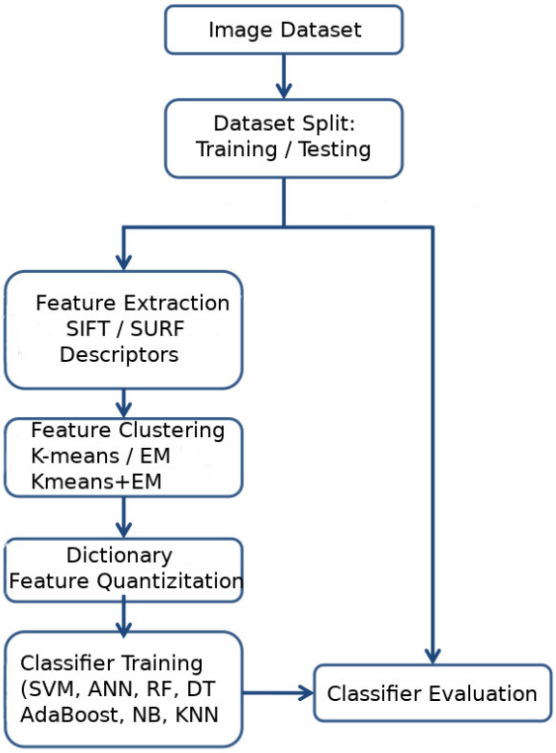


Figura 4.1: Diagrama de flujo BOVW

4.2. Conjunto de imágenes

Se utilizarán 4 conjuntos de imágenes (datasets), uno contiene cuatro clases que pertenece al conjunto de imágenes Caltech 101 que es muy popular entre los trabajos de visión por computadora. También se crearon 3 conjuntos de imágenes hechos a mano en un lienzo por computadora, obteniendo así tres categorías: figuras, símbolos y números.

4.2.1. Caltech 101

El conjunto de datos de Caltech101 contiene 101 clases o categorías, para este caso solo se seleccionaron cuatro clases, la categoría pirámide contiene 42 elementos para entrenamiento y 15 para evaluación, la categoría de revólver contiene 67 elementos para entrenamiento y 15 artículos para evaluación, la categoría de caballitos de mar contiene 42 artículos para entrenamiento y 15 elementos para la evaluación,

estegosaurus contiene 44 elementos para la entrenamiento y 15 elementos para la evaluación.

Figura 4.2: Caltech 101

4.2.2. Creación de conjunto de imágenes a mano alzada

Para obtener estos conjuntos de imágenes se programó en python un lienzo para dibujar las formas que se requieren para entrenar y evaluar el clasificador, también se utilizó como método de entrada para ingresar nuevas imágenes para su clasificación en tiempo real.

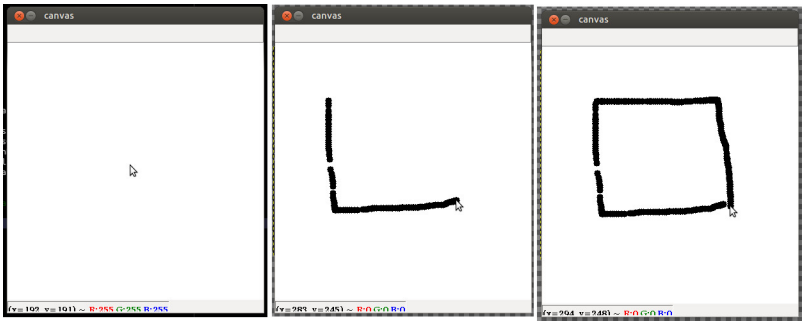


Figura 4.3: Ejemplo de creación de una clase a mano alzada

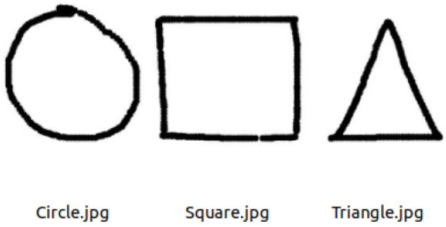


Figura 4.4: Conjunto de imágenes de Figuras

El conjunto de datos de las figuras se construyó desde cero, cada categoría contiene 100 elementos, 80 para entrenamiento y 20 para evaluación.

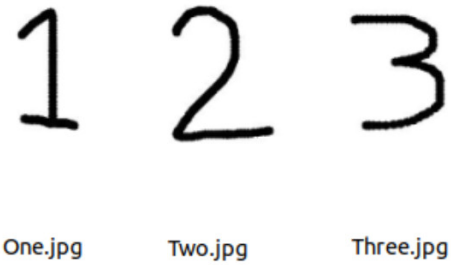


Figura 4.5: Conjunto de imágenes de Números

El conjunto de datos de números se construyó desde cero, cada categoría contiene 100 elementos, 80 para entrenamiento y 20 para evaluación.

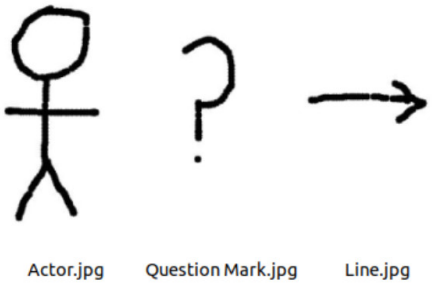


Figura 4.6: Conjunto de imágenes de Símbolos

El conjunto de datos de símbolos se construyó desde cero, la categoría de persona contiene 17 elementos para la entrenamiento y 7 para las pruebas, la categoría de signo de interrogación contiene 24 elementos para la entrenamiento y 16 para

las pruebas, la categoría de línea contiene 48 elementos para la entrenamiento y 9 elementos para la prueba.

4.2.3. Detección y descripción de la imagen

En la detección y descripción de imágenes se emplea el uso del algoritmo de SIFT (Scale-invariant feature transform), se utiliza la biblioteca de OpenCV, a continuación se muestra un fragmento del código para la detección.

```
import numpy as np
import cv2 as cv

K=[], D=[]
descriptor = cv.ORB_create()
img = cv.imread('test.jpg')
gray= cv.cvtColor(img,cv.COLOR_BGR2GRAY)
sift = cv.ORB_create()
kp = sift.detect(gray,None)
img=cv.drawKeypoints(gray,kp,img)
cv.imwrite('sift_keypoints.jpg',img)
kpts=detector.detect(gray)
kpts,des=descriptor.compute(gray,kpts)
K.append(kpts)
D.append(des)
```

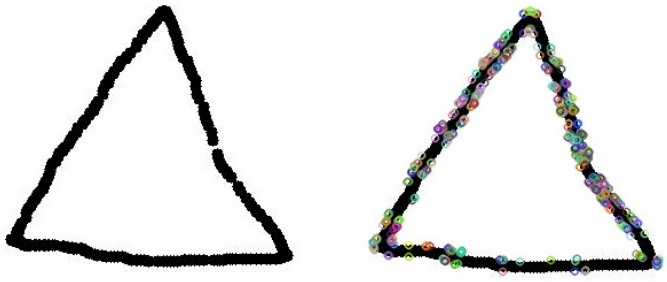


Figura 4.7: Detección de puntos de interés

4.3. Agrupamiento de datos

En esta sección es donde se forman las palabras visuales a partir de los vectores que son la descripción de las imágenes, se utiliza un algoritmo de agrupamiento K means por defecto, en este proyecto se propone utilizar el algoritmo EM (Expectación máxima).

4.3.1. Relación entre Kmeans y EM

La agrupación en clúster es un método de aprendizaje no supervisado, donde cada punto de datos o agrupación se agrupa en un subconjunto o agrupación, que contiene un tipo similar de puntos de datos.

K-Means Clustering

Es un algoritmo, que clasifica muestras en función de los atributos o características en K número de grupos. La agrupación o agrupación de muestras se realiza minimizando la distancia entre la muestra y el centroide. es decir, asignar el centroide y optimizar el centroide en función de las distancias de los puntos a él. Esto se denomina Asignación difícil, es decir, se está seguro de que ciertos puntos pertenecen a un centroide en particular y, luego, en función del método de distancia por mínimos cuadrados, optimizará la ubicación del centroide.

Ventajas de K-means

- Tiempo de ejecución.
- Mejor para datos de alta dimensión.
- Facil de implementar e interpretar.

Desventajas de K-means

- Asume que los grupos son esféricos, por lo que no funciona de manera eficiente con datos con formas geométricas complejas (en su mayoría no lineales).
- La asignación difícil puede llevar a una agrupación incorrecta.

EM clustering

En lugar de asignar datos duros a un clúster, si no se está seguro de los puntos de datos a los que pertenecen a qué grupo, se utiliza este método. Utiliza la probabilidad de una muestra para determinar la viabilidad de que pertenezca a un grupo.

Ventajas

- No asume que los clusters sean de ninguna geometría. Funciona bien con distribuciones geométricas no lineales también.
- No modifica los tamaños del clúster para que tengan estructuras específicas como lo hace K-Means (Circular).

Desventajas

- Utiliza todos los componentes a los que tiene acceso, por lo que la inicialización de los clústeres será difícil cuando la dimensionalidad de los datos sea alta.
- Difícil de interpretar.

Uso de Keans con EM

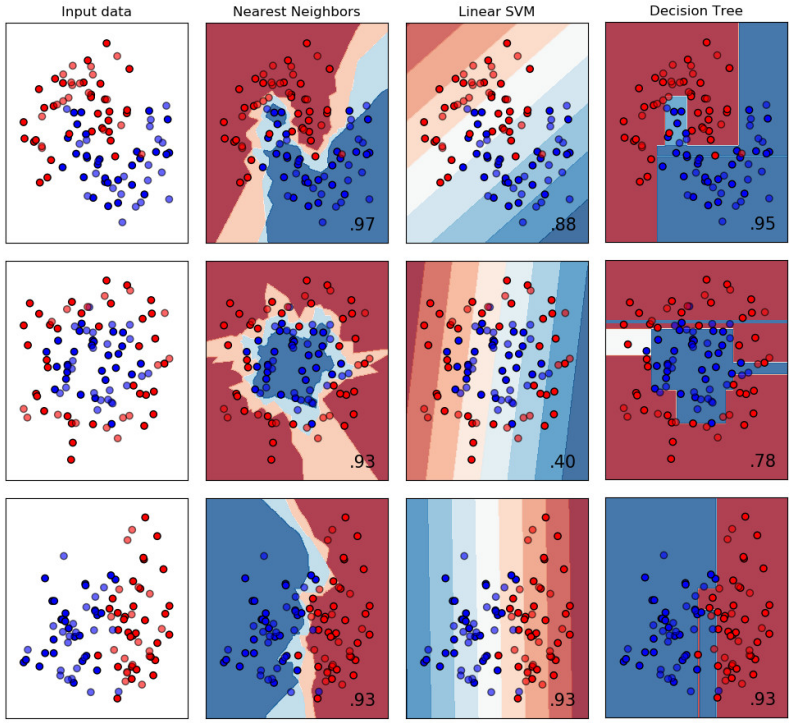
Dentro de la biblioteca de Scikit learn, donde se encuentran estos algoritmos de agrupamiento (kmeans y EM) se tiene una función de utilizar el algoritmo EM con K means, el método utilizado para inicializar los pesos, los medios y las precisiones. Debe ser uno de:

`init_params: {kmeans, random}`

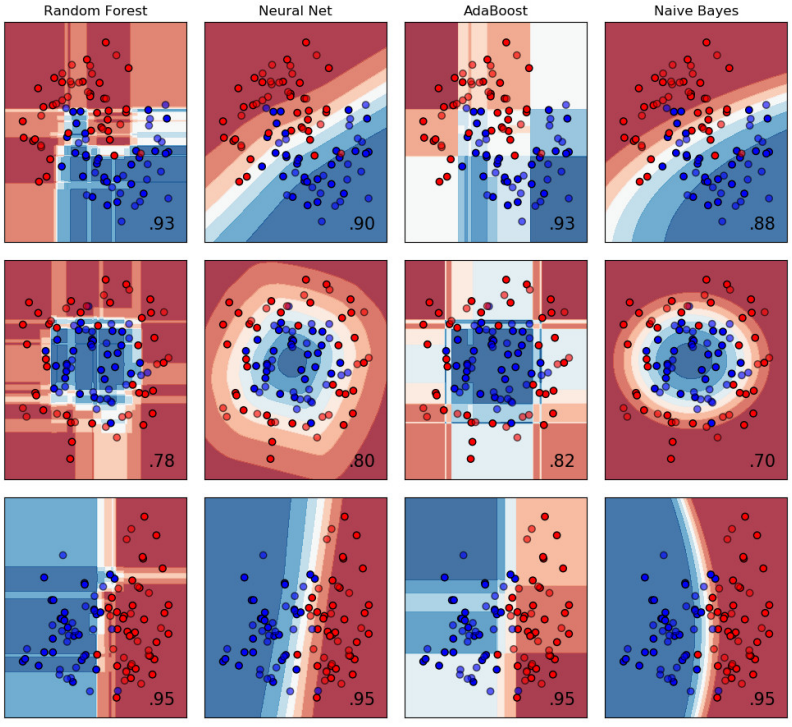
kmeans: las responsabilidades se inician utilizando kmeans.
random: las responsabilidades se inicializan aleatoriamente.

4.4. Clasificadores

Se hizo uso de 7 clasificadores con el objetivo de hacer un sistema robusto, dado a que no siempre un clasificador puede tener la misma precisión con diferentes grupos de datos como se puede apreciar en la figura



(a)



(b)

Figura 4.8: Comparación de clasificadores

Capítulo 5

Resultados

Los resultados se muestran en las tablas a continuación, solo se presentan los mejores resultados de clasificación, el mejor clasificador y el método de agrupación en valores de k (diccionario) para cada conjunto de datos.

Cada combinación, como se muestra en la tabla 4.1 es ejecutada 5 veces en relación de el tamaño de k: 100, 200, 300, 400 y 500 palabras, en cada conjunto de entrenamiento, empezando por Caltech 101 después Figuras, Números y Símbolos, con el fin de encontrar el tamaño óptimo del diccionario de palabras.

5.1. Caltech 101

En la tabla 5.1 contiene los mejores resultados para cada tamaño en k (diccionario) y el clasificador más adecuado, el método de agrupamiento en el conjunto de datos Caltech 101. A primera vista, k = 300, EM y el clasificador SVM parece ser el método más adecuado, aunque su precisión es mayor, se puede observar que la sensibilidad, f1 y la validación cruzada son menores que los resultados encontrados en k = 500, Clasificador KNN y Kmeans.

Debido a una mayor tasa de precisión y resultados de validación cruzada, la mejor combinación para el conjunto de datos Caltech 101 es KNN, Kmeans y k = 500. Se puede observar en la matriz de confusión que están mejor clasificadas las clases.

k	**Clustering**	**classifier**	**Confusion Matrix**				**Presicion**	**Recall**	**F1 Score**	**Cross Validation**
100	EM	SVM	11	0	1	3	66	65	65	70
			1	11	1	2				
			1	0	10	4				
			2	2	4	7				
200	EM	SVM	7	0	4	4	70	66	66	68
			2	9	0	4				
			1	0	13	1				
			0	0	3	12				
300	EM	SVM	6	1	4	4	74	69	69	68
			0	11	0	4				
			1	0	13	1				
			0	0	3	12				
400	EM	KNN	7	2	5	1	71	69	69	68
			1	11	1	2				
			1	0	12	2				
			0	1	2	12				
500	K means	KNN	9	3	3	0	72	71	71	73
			2	12	0	1				
			3	0	11	1				
			1	1	2	11				

Cuadro 5.1: Resultados de Caltech 101.

5.2. Figuras geométricas

En la tabla 5.2 contiene los mejores resultados para cada tamaño en k (diccionario), el clasificador más adecuado y el método de agrupamiento en el conjunto de datos figuras geométricas. En este caso, k = 200, Kmeans + EM clustering y AdaBoost son los métodos más precisos.

k	**Clustering**	**classifier**	**Confusion Matrix**			**Presicion**	**Recall**	**F1 Score**	**Cross Validation**
100	K means	ANN	18	2	0	90	90	90	93
			0	18	2				
			0	0	20				
200	K means EM	AdaBoost	20	0	0	96	96	96	91
			0	18	2				
			0	0	20				
300	K means EM	ANN	20	0	0	93	93	93	93
			0	17	3				
			0	1	19				
400	K means EM	ANN	18	2	0	92	91	91	93
			0	17	3				
			0	1	19				
500	K means	SVM	20	0	0	93	93	93	95
			0	18	2				
			0	2	18				

Cuadro 5.2: Resultados de figuras geométricas.

5.3. Números

En la tabla 5.3 contiene los mejores resultados para cada tamaño en k (diccionario), el clasificador más adecuado y el método de agrupamiento en el conjunto de datos de Números. Para este conjunto de datos, k = 100, Kmeans + EM clustering y RF son los métodos más precisos.

k	Clustering	classifier	Confusion Matrix			Presicion	Recall	F1 Score	Cross Validation
100	K means EM	RF	19	0	1	87	86	86	83
			0	15	5				
			0	2	18				
200	EM	NB	7	12	1	83	71	69	76
			0	20	0				
			0	4	16				
300	K means	ANN	15	1	4	80	73	73	88
			0	10	10				
			0	1	19				
400	K means	ANN	14	6	0	86	81	82	88
			0	19	1				
			0	4	16				
500	K means	ANN	14	0	6	83	75	75	88
			0	12	8				
			1	0	19				

Cuadro 5.3: Resultados de Números.

5.4. Símbolos

En la tabla 5.4 contiene los mejores resultados para cada tamaño en k (diccionario), el clasificador más adecuado y el método de agrupamiento en el conjunto de datos de Símbolos. Para este conjunto de datos, k = 100, el algoritmo de agrupamiento EM y el clasificador SVM son los métodos más precisos. Aunque la precisión, sensibilidad y f1 de GNB obtienen un puntaje perfecto de 100, la validación cruzada es la decisión crítica.

k	**Clustering**	**classifier**	**Confusion Matrix**			**Presicion**	**Recall**	**F1 Score**	**Cross Validation**
100	EM	SVM	7	0	0	100	100	100	96
			0	16	0				
			0	0	9				
200	K means	NB	7	0	0	100	100	100	87
			0	16	0				
			0	0	9				
300	EM	SVM	6	0	1	97	96	96	96
			0	16	0				
			0	0	9				
400	EM	SVM	6	0	1	97	96	96	96
			0	16	0				
			0	0	9				
500	K means EM	ANN	7	0	0	97	96	96	100
			1	15	0				
			0	0	9				

Cuadro 5.4: Resultados de Símbolos.

5.5. Resumen

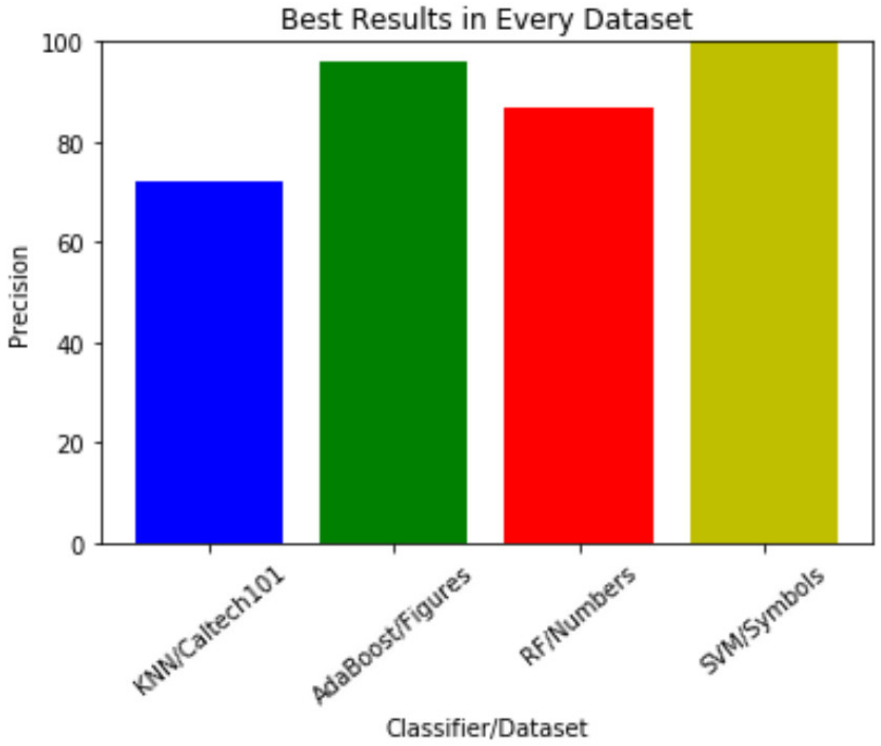


Figura 5.1: Mejor rendimiento de clasificador en cada dataset

k	Clustering	Clasifier	Dataset	Confusion Matrix				Presicion	Recall	F1 Score	Cross Validation
500	K means	KNN	Caltech 101	9	3	3	0	72	71	71	73
				2	12	0	1				
				3	0	11	1				
				1	1	2	11				
200	K means EM	AdaBoost	Figuras Geométricas	20	0	0		96	96	96	91
				0	18	2					
				0	0	20					
100	K means EM	RF	Números	19	0	1		87	86	86	83
				0	15	5					
				0	2	18					
100	EM	SVM	Símbolos	7	0	0		100	100	100	96
				0	16	0					
				0	0	9					

Cuadro 5.5: Mejores resultados en cada conjunto de imágenes

Capítulo 6

Conclusiones

En Este proyecto se hizo el uso de una biblioteca del lenguaje de programación python: Scikit-Learn. Dentro de esta biblioteca hay una comparación de varios clasificadores en conjuntos de datos sintéticos. En espacios de alta dimensión, los datos se pueden separar fácilmente de forma lineal y los clasificadores, como los Bayes ingenuos y los SVM lineales, pueden tener un mejor desempeño que otros clasificadores. En este experimento, se ejecutaron diferentes combinaciones de agrupación en clústeres y clasificadores y los resultados demuestran que no solo la combinación de clasificador SVM y agrupación en Kmeans es la mejor opción para el Modelo de Bolsa de Palabras Visuales, de lo contrario, un proceso híbrido de clasificación y agrupación llevará a un mejor rendimiento para diferentes conjuntos de datos .

Bibliografía

Aitkin, M. and Aitkin, I. (1996). A hybrid em/gauss-newton algorithm for maximum likelihood in mixture distributions. *Statistics and Computing*, 6(2):127–130.

Arman, F. (1993). Cad-based vision: Object recognition in cluttered range images using recognition strategies. *Computer Vision and Image Understanding*, 58(1):33–48.

Bay, H., T. T. and Van Gool, L. (2006). Surf: Speeded up robust features.

Bhat, C. R. (1997). An endogenous segmentation mode choice model with an application to intercity travel. *Transportation Science*, 31(1):34–48.

Boyles, R. A. (1983). On the convergence of the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 45(1):47–50.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22.

Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., and Zisserman, A. (2009). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338.

Grauman, K. and Darrell, T. Efficient image matching with distributions of local invariant features. *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05).*

Hamerly, G. and Elkan, C. (2002). Alternatives to the k-means algorithm that find better clusterings. *Proceedings of the eleventh international conference on Information and knowledge management - CIKM 02.*

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. *Procedings of the Alvey Vision Conference 1988.*

Ke, Y. and Sukthankar, R. (2004). Pca-sift: a more distinctive representation for local image descriptors. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*

Koh, J. E., Ng, E. Y., Bhandary, S. V., Hagiwara, Y., Laude, A., and Acharya, U. R. (2018). Automated retinal health diagnosis using pyramid histogram of visual words and fisher vector techniques. *Computers in Biology and Medicine*, 92:204–209.

Lazebnik, S., Schmid, C., and Ponce, J. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR06).*

Leibe, B. and Schiele, B. (2003). Interleaved object categorization and segmentation. *Procedings of the British Machine Vision Conference 2003.*

Lewis, D. D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. *Machine Learning: ECML-98 Lecture Notes in Computer Science*, page 4–15.

Lowe, D. (1999). Object recognition from local scale-invariant features. *Proceedings of the Seventh IEEE International Conference on Computer Vision.*

Mclachlan, G. J. and Krishnan, T. (2008). The em algorithm and extensions, 2e. *Wiley Series in Probability and Statistics.*

Mikolajczyk, K. and Schmid, C. A performance evaluation of local descriptors. *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*

Niebles, J. C., Wang, H., Wang, H., and Fei-Fei, L. (2006). Unsupervised learning of human action categories using spatial-temporal words. *Procedings of the British Machine Vision Conference 2006.*

Pentland, A. P. (1987). Recognition by parts.

Ruud, P. A. (1991). Extensions of estimation methods using the em algorithm. *Journal of Econometrics*, 49(3):305–341.

Sivic and Zisserman (2003). Video google: a text retrieval approach to object matching in videos. *Proceedings Ninth IEEE International Conference on Computer Vision.*

Sivic, J., Russell, B. C., Efros, A. A., Zisserman, A., and Freeman, W. T. (2005). Discovering objects and their location in images. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 1, pages 370–377. IEEE.

Swain, M. and Ballard, D. (1990). Indexing via color histograms. *[1990] Proceedings Third International Conference on Computer Vision.*

Train, K. E. (2008). Em algorithms for nonparametric estimation of mixing distributions. *Journal of Choice Modelling*, 1(1):40–69.

Turk, M. A. and Pentland, A. P. (1991). Eigenfaces for recognition. *Intelligent Robots and Computer Vision IX: Algorithms and Techniques.*

Von Ahn, L. and Dabbish, L. (2004). Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM.

Weeks, D. E. and Lange, K. (1989). Trials, tribulations, and triumphs of the em algorithm in pedigree analysis. *Mathematical Medicine and Biology*, 6(4):209–232.

Wong, K., Casey, R., and Wahl, F. (1982). *Document Analysis System*. Research reports // IBM. IBM Thomas J. Watson Research Division.

Wu, C. F. J. (1983). On the convergence properties of the em algorithm. *The Annals of Statistics*, 11(1):95–103.

Yang, J., J. Y. H. A. and Ngo, C. (2007). Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, volume 1, page 206. ACM.

Zhao, T. and Nevatia, R. (2004). Tracking multiple humans in crowded environment. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, pages 406–413, Washington, DC, USA. IEEE Computer Society.

Apéndice A

Publicaciones

Figura A.1: Publicación en el Congreso Mexicano de Inteligencia Artificial Mérida 2018

CIRCULATION IN COMPUTER SCIENCE
An International, Professionally Refereed Scholarly Journal
www.ccsarchive.org

Ref: CCS/53AT58/2018
ISSN 2456-3692

Date: July 18, 2018

Certificate of Publication

To:
Baldemar Zurita
Master in Computer Systems
Apizaco Technological Institute
Apizaco, Mexico

We have pleasure to announce the publication of your article in *Circulation in Computer Science* journal **Volume 3 – Number 4 (June 2018 Edition)**.

Article Title: Hybrid Classification in Bag of Visual Words Model

Uniform Resource Identifier: *http://www.ccsarchive.org/archive/volume3/number4/ccs-2018-252-85*

Full-Text Link: *http://www.ccsarchive.org/articles/volume3/number4/ccs-2018-252-85.pdf*

CrossRef DOI: *https://doi.org/10.22632/ccs-2018-252-85*

Author(s): Baldemar Zurita, Luís Luna, José Hernández and José Ramírez

Thanking you,

Regards,
Editorial Office,
Circulation in Computer Science
editor@ccsarchive.org

CCS Archive is published by
CSL Press
2 River Terrace, Suite #23, NY, New York 10282, USA

Figura A.2: Publicación en revista Circulation in Computer Science, ISSN 2456-3692, Vol. 3, Num. 4

JOURNAL OF COMPUTER – JOC

ISSN: 2518-6205

Impact Factor: 0.337

JoC

Certificate of Publication

This certificate is awarded to Baldemar Zurita as an appreciation for publishing a paper entitled "BOVW FOR CLASSIFICATION OF GEOMETRIC SHAPES" in Journal of Computer - JoC

Editor, JoC

www.journal.computer

Volume: 3

Issue: 6

Month, Year: June, 2018

Figura A.3: Publicación en la revista Journal of Computer, ISSN 2518-6205, Vol. 3, No.6

Apéndice B

Estancias

SEP

TECNOLÓGICO NACIONAL DE MÉXICO
Instituto Tecnológico de Apizaco
División de Estudios de Posgrado e Investigación

"Año del Centenario de la Promulgación de la Constitución Política de los Estados Unidos Mexicanos"

Apizaco, Tlax., 24/Mayo/2017

Oficio: DEPI/165/17

ASUNTO: **Presentación para Realizar Estancia**

M.C. OSCAR ATRIANO PONCE
DIRECTOR GENERAL DE LA EMPRESA
"SMARTSOFT AMERICA BUSINESS
APPLICATION S.A. DE C.V.",
PRESENTE.

Por este conducto me dirijo a usted para presentar al C. **Ing. Baldemar Zurita Islas** alumno de segundo semestre de la Maestría en Sistemas Computacionales con Nº de control M16370022, quien tiene interés en realizar una ***Estancia*** en la empresa que usted dignamente dirige, teniendo como **Objetivo:** Diseñar e implementar un sistema de reconocimiento de formas utilizando un algoritmo EM, en el periodo comprendido del 17 de Abril al 17 de Octubre de 2017 los días Miércoles y Jueves en un horario de 09:00 a 18:00 hrs. realizando las siguientes actividades: 1. **Mes de Abril-Mayo, Construcción del editor de prototipos de alto nivel;** 2. **Mes de Junio-Julio, Interfaz de wizard de transformación;** 3. **Mes de Agosto-Septiembre, Administrar las pantallas dentro de un prototipo de alto nivel;** 4. **Mes de Octubre, Permitir que los colaboradores agreguen comentarios a los prototipos.** Cabe mencionar que el alumno cuenta con la incorporación de estudiantes al Seguro Facultativo del Régimen del Seguro Social, teniendo el número de **Seguridad Social 6105862094**1.

Sin otro particular por el momento y agradeciendo de antemano la atención a la presente, me es grato quedar de usted.

ATENTAMENTE
PENSAR PARA SERVIR, SERVIR PARA TRIUNFAR

SECRETARÍA DE EDUCACIÓN PÚBLICA
TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
DIRECTOR

DIRECCIÓN

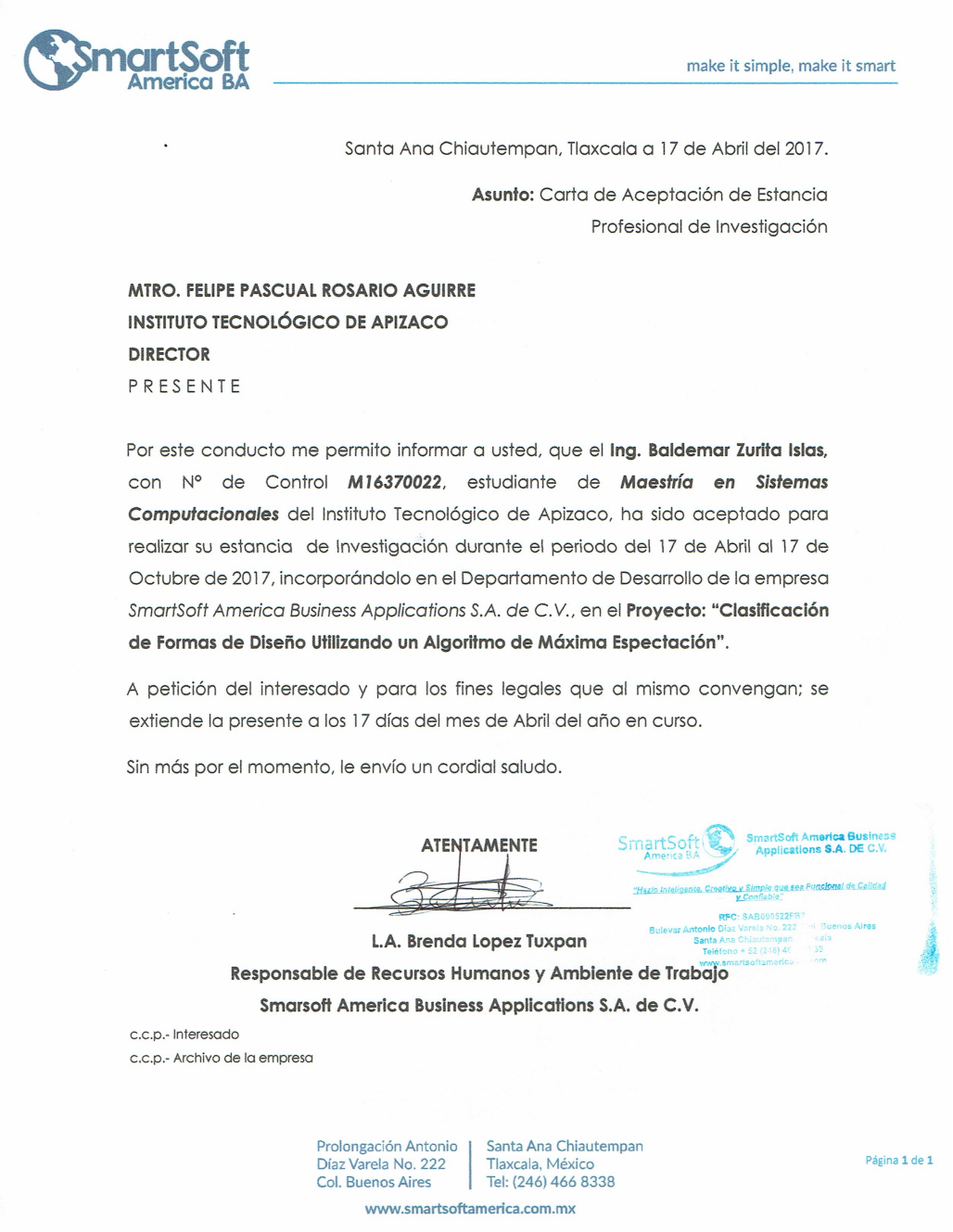
Recibí Original
19/6/2017
Brenda Lopez Texpan
RRH y AT

SmartSoft America Business Applications S.A. de C.V.

FPRA/MAGM/IEGV/MJSG/mebr

Carretera Apizaco – Tzompantepec, esquina con Av. Instituto Tecnológico S/N, Comunidad Apizaco - Xonmantepec, Tlaxcala, Mex. C.P. 90300 Apizaco, Tlaxcala
Tel. (01) 241 417 20 10, Ext. 146, 246 e-mail: posgrado@itapizaco.edu.mx

Figura B.1: Carta de presentación para realizar estancias



SmartSoft America BA

make it simple, make it smart

Santa Ana Chiautempan, Tlaxcala a 17 de Abril del 2017.

Asunto: Carta de Aceptación de Estancia
Profesional de Investigación

MTRO. FELIPE PASCUAL ROSARIO AGUIRRE
INSTITUTO TECNOLÓGICO DE APIZACO
DIRECTOR
P R E S E N T E

Por este conducto me permito informar a usted, que el **Ing. Baldemar Zurita Islas,** con N° de Control ***M16370022***, estudiante de ***Maestría en Sistemas Computacionales*** del Instituto Tecnológico de Apizaco, ha sido aceptado para realizar su estancia de Investigación durante el periodo del 17 de Abril al 17 de Octubre de 2017, incorporándolo en el Departamento de Desarrollo de la empresa *SmartSoft America Business Applications S.A. de C.V.*, en el **Proyecto: "Clasificación de Formas de Diseño Utilizando un Algoritmo de Máxima Espectación"**.

A petición del interesado y para los fines legales que al mismo convengan; se extiende la presente a los 17 días del mes de Abril del año en curso.

Sin más por el momento, le envío un cordial saludo.

ATENTAMENTE

SmartSoft America BA — SmartSoft America Business Applications S.A. DE C.V.

"Hazlo Inteligente, Creativo y Simple que sea Funcional de Calidad y Confiable"

RFC: SAB090522FB1
Bulevar Antonio Díaz Varela No. 222 [illegible] Buenos Aires
Santa Ana Chiautempan [illegible] Tlaxcala
Teléfono + 52 (246) 4[illegible]
www.smartsoftamerica[illegible].com

L.A. Brenda Lopez Tuxpan
Responsable de Recursos Humanos y Ambiente de Trabajo
Smarsoft America Business Applications S.A. de C.V.

c.c.p.- Interesado
c.c.p.- Archivo de la empresa

Prolongación Antonio Díaz Varela No. 222 Col. Buenos Aires | Santa Ana Chiautempan Tlaxcala, México Tel: (246) 466 8338

www.smartsoftamerica.com.mx

Página 1 de 1

Figura B.2: Carta de aceptación para realizar estancias

SmartSoft America BA

make it simple, make it smart

Santa Ana Chiautempan, Tlax., 30 de Octubre del 2017

Asunto: Carta de Liberación de Estancia Profesional

Mtro. Felipe Pascual Rosario Aguirre
Director del Instituto
Tecnológico de Apizaco

PRESENTE

La que suscribe ***Lic. Brenda López Tuxpan*** , Responsable de Recursos Humanos y Ambiente de Trabajo de la *empresa SmartSoft America Business Applications S.A. de C.V.*, por medio de la presente se *hace constar* que el ***Ing. Baldemar Zurita Islas,*** Realizó satisfactoriamente su estancia profesional en nuestra empresa, **del 17 de Abril de 2017 al 17 de Octubre del 2017,** incorporándola en el Departamento de Desarrollo, en el proyecto**: "Clasificación de Formas de Diseño Utilizando un Algoritmo de Máxima Espectación".**

A petición del interesado y para fines legales que al mismo convengan; se extiende la presente constancia a los 30 días del mes de Octubre del año 2017.

ATENTAMENTE

SmartSoft America BA — SmartSoft America Business Applications S.A. DE C.V.

"Hazlo inteligente, Creativo y Simple que sea Funcional de Calidad y Confiable"

RFC: SAB090522FB7
Bulevar Antonio Díaz Varela No. 222, Col. Buenos Aires
Santa Ana Chiautempan, Tlaxcala
Teléfono + 52 (246) 46 6 83 38
www.smartsoftamericamx.com

L.A. Brenda López Tuxpan
Responsable de Recursos Humanos y Ambiente de Trabajo

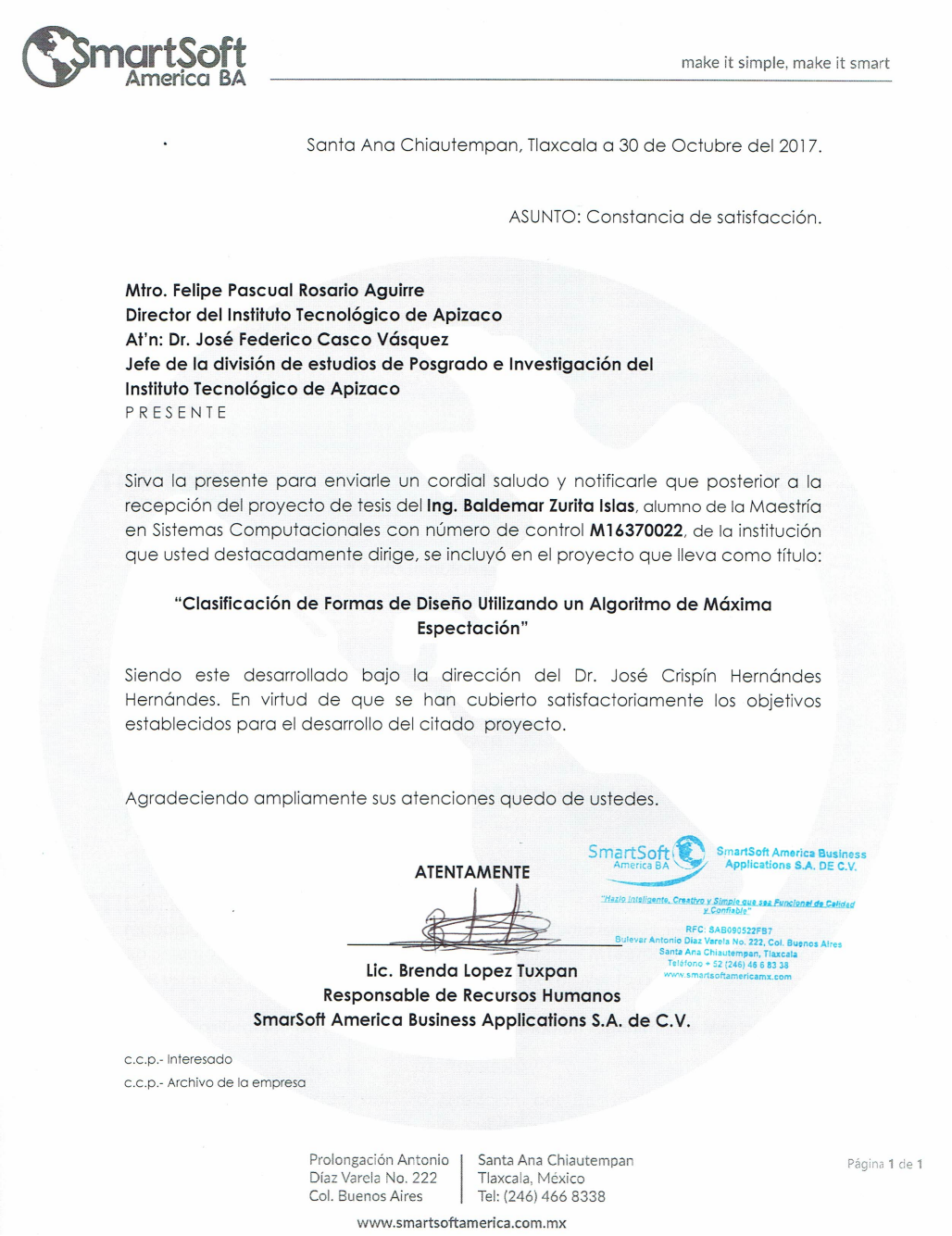
c.c.p.- Interesado
c.c.p.- Archivo de la empresa

Prolongación Antonio Díaz Varela No. 222
Col. Buenos Aires
Santa Ana Chiautempan
Tlaxcala, México
Tel: (246) 466 8338

www.smartsoftamerica.com.mx

Página 1 de 1

Figura B.3: Carta de liberación de estancias en SmartSoft de América



SmartSoft America BA

make it simple, make it smart

Santa Ana Chiautempan, Tlaxcala a 30 de Octubre del 2017.

ASUNTO: Constancia de satisfacción.

Mtro. Felipe Pascual Rosario Aguirre
Director del Instituto Tecnológico de Apizaco
At'n: Dr. José Federico Casco Vásquez
Jefe de la división de estudios de Posgrado e Investigación del Instituto Tecnológico de Apizaco
PRESENTE

Sirva la presente para enviarle un cordial saludo y notificarle que posterior a la recepción del proyecto de tesis del **Ing. Baldemar Zurita Islas**, alumno de la Maestría en Sistemas Computacionales con número de control **M16370022**, de la institución que usted destacadamente dirige, se incluyó en el proyecto que lleva como título:

"Clasificación de Formas de Diseño Utilizando un Algoritmo de Máxima Espectación"

Siendo este desarrollado bajo la dirección del Dr. José Crispín Hernándes Hernándes. En virtud de que se han cubierto satisfactoriamente los objetivos establecidos para el desarrollo del citado proyecto.

Agradeciendo ampliamente sus atenciones quedo de ustedes.

ATENTAMENTE

SmartSoft America BA
SmartSoft America Business Applications S.A. DE C.V.
"Hazlo inteligente, Creativo y Simple que sea Funcional de Calidad y Confiable"
RFC: SAB090522FB7
Bulevar Antonio Díaz Varela No. 222, Col. Buenos Aires
Santa Ana Chiautempan, Tlaxcala
Teléfono + 52 (246) 46 6 83 38
www.smartsoftamericamx.com

Lic. Brenda Lopez Tuxpan
Responsable de Recursos Humanos
SmarSoft America Business Applications S.A. de C.V.

c.c.p.- Interesado
c.c.p.- Archivo de la empresa

Prolongación Antonio Díaz Varela No. 222
Col. Buenos Aires
Santa Ana Chiautempan
Tlaxcala, México
Tel: (246) 466 8338
www.smartsoftamerica.com.mx

Página 1 de 1

Figura B.4: Carta de satisfacción emitida por la empresa SmartSoft de América