Trabajo de Inteligencia Artificial (Grupo 2) 2012-13

Reconocimiento de imágenes usando redes neuronales

Este trabajo está basado en un trabajo que el profesor Tom Mitchell propone a sus alumnos del curso "Machine Learning" la Universidad Carnagie Mellon. También se describe también en la Seccion 4.7 de su libro "Machine Learning". El objeto fundamental del trabajo es la implementación de algoritmos de aprendizaje basados en redes neuronales y su aplicación al problema de reconocer (sobre imágenes en formato PGM), personas, posturas y gestos, a partir de imágenes de caras. Para realizar el trabajo es necesario conocer los contenidos que se dan en el tema 9 de la asignatura.

PARTE I: Lectura de las referencias

Se recomiendan las siguientes referencias para comprender bien el algoritmo de retropropagación. Además, en la primera de ellas (sección 4.7) se describe una aplicación análoga a la que se pide en este trabajo.

- Tema 4 del libro "Machine Learning" de Tom Mitchell (en la biblioteca del centro hay varios ejemplares para su préstamo y consulta). En la sección 4.5 se describe el algoritmo de retropropagación y en la 4.7 se dan detalles sobre la aplicación de redes neuronales al reconocimiento de imágenes.
- Sección 20.5 del capítulo 20 del libro *Inteligencia Atificial: un enfoque moderno* de S. Russell y P. Norvig (segunda edición en español, disponible en la biblioteca). En la tercera edición (sólo en inglés) las redes neuronales se tratan en la sección 18.7.

PARTE II: Implementación de algoritmos de aprendizaje de redes neuronales

Implementar en Python los siguientes algoritmos de aprendizaje para redes neuronales:

- Regla delta para entrenamiento del perceptrón simple
- Algoritmo de retropropagación
- Algoritmo de retropropagación con momentum (opcional)

En todos los casos, considerar el sigmoide como función de activación. En el caso del perceptrón multicapa, es suficiente con implementar el algoritmo para una y para dos capas ocultas, aunque es preferible que el algoritmo se haga de manera genérica y pueda ejecutarse con cualquier número de capas ocultas. En cualquier caso, el usuario debe poder introducir como argumento de entrada tanto el número de capas ocultas de la red, como el número de neuronas de cada capa.

NOTA: La implementación debe ser original, no permitiéndose usar ninguna parte de código o librerias (de redes neuronales) que no hayan sido desarrolladas . Se permite usar módulos de la librería estándar de Python.

PARTE III: Aprendizaje de redes neuronales para el reconocimiento del alfabeto en el lenguaje de signos

En esta parte se trata de construir redes neuronales que permitan reconocer, a partir de una imagen de la cara de una persona:

- Si está de frente, mirando hacia arriba o hacia abajo.
- Si tiene los ojos abiertos, cerrados o con gafas de sol.
- Si está sonriendo, serio o sacando la lengua.
- La persona de la que se trata, de entre cinco personas posibles.

Esta lista de problemas está en orden ascendente de dificultad. No es necesario resolver todos los problemas anteriores para conseguir la máxima calificación, pero se tendrá en cuenta el grado de dificultad del problema o problemas de la lista anterior que se resuelvan.

Las redes neuronales que se obtengan para cada uno de los problemas anteriores serán el resultado de la ejecución de un algoritmo de entrenamiento sobre un conjunto de entrenamiento, con una estructura de red dada y unos parámetros concretos. Nótese que esto puede suponer que se hagan varios experimentos, en los que se obtienen varias redes (variando la estructura de la red, el algoritmo, los parámteros, etc.), escogiendo la de mayor rendimiento en cada problema.

Será necesario también que la implementación contemple alguna manera de guardar de manera permanente (en un fichero) la red neuronal obtenida, de manera que no sea necesario repetir cada vez el proceso de entrenamiento.

Obtención del conjunto de entrenamiento

Una parte importante de este trabajo es la tarea de construir un conjunto de entrenamiento. Para ello se deben hacer numerosas fotos de la cara de cinco personas; cada foto debe combinar una postura, una posición de los ojos y un gesto concreto. No es necesario que las imágenes sean de una resolución alta, ni en color. Para mayor sencillez de la tarea de clasificación, se pueden hacer las fotos sobre fondo liso y claramente diferenciado.

Se recomienda usar el formato PGM con una escala de grises no muy amplia. PGM (Portable GreyMap) es un formato de imagen muy simple, en el que cada pixel está codificado por un número (usualmente un byte), que indica su escala de gris. Para esta aplicación no deberíamos necesitar imágenes de más de 30x30 píxeles y con una amplitud de grises no mayor de 200. En wikipedia se puede encontrar más información sobre PGM y la libreria Netpbm que permite pasar de unos formatos de imagen a otros (Netpbm está disponible para numerosas plataformas). Otra posibilidad es usar el programa IrfanView, que permite conversión entre numerosos formatos de imagen.

NOTA: Aunque el trabajo (la implementación) es individual, la obtención del conjunto de entrenamiento puede hacerse en grupos de hasta TRES alumnos, indicándolo claramente.

PARTE IV: Documentación del experimento

La implementación debe estar documentada (como comentarios dentro del fichero). Debe describirse el trabajo realizado, los problemas que se han resuelto y la experimentación que se ha llevado a cabo para obtener las redes neuronales que sirven para realizar las distintas tareas de clasificación propuestas anteriormente. Se debe describir, al menos:

- Proceso de obtención del conjunto de entrenamiento.
- Distintas posibilidades intentadas, en cuanto a estructura de la red (número de capas ocultas, número de unidades en cada capa, etc.)
- Resultados según los distintos algoritmos de entrenamiento (regla delta, retropropagación, momentum)
- Estructura de datos usada para representar una red neuronal.
- Generación de los pesos iniciales, factores de aprendizaje,...
- Distintos criterios de parada.

PARTE V: Envío del trabajo

Antes de la hora y fecha marcada para la entrega, el trabajo debe ser enviado por correo electrónico al profesor (dirección `jruiz` en la Universidad de Sevilla). Se debe enviar sólamente la implementación en Python (Atención: NO enviar las imágenes usadas como conjunto de entrenamiento)

El trabajo no se considerará entregado si no se ha recibido acuse de recibo por parte del profesor en el plazo de un día desde que se envía.

PARTE V: Defensa

Se recomienda tener preparada la presentación de la defensa en un portátil propio. Si no se dispone de portátil para la defensa, contactar previamente con el profesor. En cualquier caso, el código usado para la demostración deberá ser el mismo que se envíe antes de la fecha tope de entrega.

En la presentación se ha de mostrar el conjunto de entrenamiento usado y dar ejemplos sobre cómo clasifican las redes neuronales finalmente obtenidas para cada problema. Los ejemplos que se usen para mostrar la clasificación deben ser variados y no formar parte del conjunto de entrenamiento.

También podrá pedirse que se muestre la ejecución de los algoritmos de entrenamiento. Obviamente, las redes finalmente obtenidas son el resultado de un proceso de entrenamiento realizado previamente, que no hay que realizar en la defensa. Sin embargo, se debe traer preparada una pequeña demostración en la que se vea la ejecución de los distintos algoritmos implementados (por ejemplo, con una versión modificada en la que se obtenga alguna salida que indique cómo van actualizándose los pesos).

Finalmente, se podrá pedir que se expliquen con calridad y soltura algunas partes del código implementado.

Copias

Los trabajos son individuales. La discusión y el intercambio de información de carácter general con los compañeros se permite (e incluso se recomienda), pero NO al nivel de código. Cualquier plagio o compartición de código que se detecte significará automáticamente la calificación de cero en la asignatura para TODOS los alumnos involucrados. Por tanto a estos alumnos NO se les conserva, para futuras convocatorias, ninguna nota que hubiesen obtenido hasta el momento.

Criteros de evaluación

- Calidad de los resultados en la identificación de signos
- Documentación
- Eficiencia en la ejecución
- Claridad y buen estilo de programación.
- Presentación realizada el día de la defensa.

En la página de la asignatura en WEBCT se ha abierto un foro poder comentar cualquier cuestión sobre el trabajo. Se recomienda que las consultas se realicen a través del foro. Este foro también se utilizará para publicar cualquier actualización o aclaración al trabajo.

Inteligencia artificial (Ingeniería Informática - Tecnologías Informáticas)
José Luis Ruiz Reina
Dpto. de Ciencias de la Computación e Inteligencia Artificial
Escuela Superior de Ingeniería Informática
Universidad de Sevilla