PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE
ESCUELA DE INGENIERÍA
IIC-2613

TAREA 4: ALGORITMOS DE APRENDIZAJE DE MÁQUINA

LA TAREA ES INDIVIDUAL

Fecha de Entrega: 9/11, 18:00 hrs. (Siding).

1 Objetivo

Muchach@s, en esta tarea tendrán la oportunidad de poner en práctica los conocimientos que han aprendido en clases. En particular, probarán el rendimiento de diferentes clasificadores para reconocer imágenes de distintos tipos de escenas. Para esto usarán el set de datos MIT-67, el cual contiene imágenes de 67 tipos de escenas.

2 Información General

2.1 Espacio de Características (Feature space)

Como comentamos en clase, redes neuronales de aprendizaje profundo (deep learning) han surgido recientemente como una poderosa herramienta que permite aprender características (features) que permiten aumentar significativamente el rendimiento de diversas tareas de clasificación, tales como reconocimiento de voz, texto, o imágenes. Como también discutimos en clases, en general, el entrenamiento de estas redes requiere ajustar millones de parámetros, lo cual es un proceso lento y altamente demandante de recursos computacionales. Afortunadamente, gracias a los contactos de su profesor y su acceso al cluster del grupo de investigación en Inteligencia de Máquina (`grima.ing.puc.cl`), cada una de las imágenes usada en esta tarea fue previamente procesada por una red neuronal profunda para obtener un conveniente conjunto de atributos, lo cual facilitará la labor del clasificador utilizado. La red neuronal profunda utilizada se denomina Alex's Net, y pronto veremos en clases los detalles de su arquitectura. Adicionalmente, en la ayudantía Gabriel les contará un poco al respecto.

2.2 Preprocesamiento: Ventana deslizante y max-pooling

Una de las características relevantes del mundo visual es la alta coherencia local de los patrones visuales. Por ejemplo, al considerar imágenes de bares, existen estructuras espacialmente localizadas que se repiten, como pisos, mesones, áreas con botellas, etc. Para poder capturar esta información local se utiliza un estrategia denominada ventana deslizante (sliding window), que consiste en deslizar sobre la imagen una ventana espacial según un paso regular en la dirección horizontal y vertical. Así es posible obtener vectores de atributos de distintas áreas de cada imagen que denominaremos patches, tal como muestra la Figura 1.

Dado que las imágenes usadas en esta tarea tienen una resolución de 256x320 pixeles, al utilizar una ventana de 64x64 pixeles y un paso de 32 pixeles, se obtiene en cada imagen un total de 7x9=63 patches. Cada uno de estos patches se ingresa a Alex's net para obtener un vector (feature vector) de 4096 dimensiones por cada patch. Luego, para obtener el vector de atributos de la imagen completa, se integran los vectores de cada patch usando una estrategia denominada max-pooling. Ésta consiste en tomar la máxima respuesta para cada componente del vector de atributos. Por ejemplo, si en lugar de 4096 cada vector fuera de 5 dimensiones y tuvieramos 3 patches, el resultado de max-pooling para los siguientes 3 vectores sería: maxpool {(**10**, 3, 4, 5, 6); (3, **21**, 4, 5, 6); (1, 3, **80**, **100**, **50**)}=(10, 21, 80, 100, 50). En el caso de la tarea, para cada imagen el proceso de max-pooling entrega un vector de 4096 dimensiones.

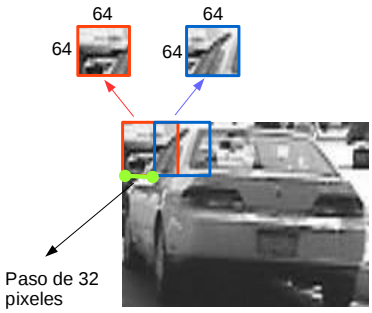

Figure 1: Ventana deslizante usando patches de 64x64 pixeles y un paso de 32 pixeles.

Para fines de la tarea probaremos con 2 set de datos, uno pequeño compuesto de sólo 10 clases, y el set completo con las 67 categorías. Los archivos utilizados estarán disponibles en el sitio web del curso, bajo los nombres: MIT67-Small-Train.zip, MIT67-Small-Test.zip, MIT67-Train.zip, y MIT67-Test.zip, según corresponda a entrenamiento o test.

2.3 scikit-learn

Para el entrenamiento y prueba de los clasificadores usaremos la librería scikit-learn de Python, disponible en: http:\www.scikit-learn.org. En la ayudantía de la tarea se explicará el funcionamiento de esta librería y se mostrarán algunos ejemplos. Además, de mostrará el código necesario para leer los archivos de entrada.

3 Actividades

3.1 Vecinos Cercanos

- Utilizando el set MIT67-Small-Train.zip pruebe el rendimiento de un clasificador de vecinos cercanos para clasificar las imágenes del set MIT67-Small-Test.zip. Para el clasificador utilice distancia euclideana y pruebe con distinto número de vecinos. Comente sus resultados.

  En esta actividad y las siguientes, reporten los resultados de clasificación utilizando una matriz de confusión (http://en.wikipedia.org/wiki/Confusion_matrix).

- Repita la actividad anterior, pero en esta oportunidad pruebe con los sets MIT67-Train.zip y MIT67-Test.zip. Compare sus resultados con los obtenidos en el caso anterior. Comente sus observaciones.

3.2 Redes Neuronales

- Utilizando el set MIT67-Train.zip entrene una red neuronal:
  - Para la capa de entrada utilice tantas neuronas como atributos, i.e., 4096.
  - Para la capa de salida pruebe con 2 posibles codificaciones de su elección.
  - Para la capa oculta ajuste empíricamente el número de neuronas.
  - Entrene las redes utilizando gradientes estocástico, i.e., el método incremental visto en clases.
- Pruebe el rendimiento del modelo obtenido utilizando el set MIT67-Test.zip. Reporte sus resultados.
- Compare el resultado obtenido para las 2 codificaciones seleccionadas para la capa de salida. Justifique sus observaciones.
- ¿Cómo determinó el número de neuronas de la capa oculta?

- ¿Qué criterio utilizó para terminar de iterar el proceso de entrenamiento?.
- ¿Cuál es la exactitud promedio en cada set, entrenamiento y test?.
- ¿Observa sobreajuste?, comente.
- ¿Qué categoría obtiene el mejor rendimiento?, ¿Cuál es el más bajo?. Indique razones que puedan justificar estos resultados. Para esto mire las imágenes reales, las cuales están en los archivos Image-Train.zip e ImageTest.zip.

3.3 Máquinas de Vectores de Soporte

- Utilizando el set MIT67-Train.zip entrene una máquina de vectores de soporte.
  - Utilice un kernel lineal.
  - Penalidad del tipo "l_2" para los pesos.
  - Penalidad el tipo "squared_hinge" para las variables slack.
  - Use la optimización en el espacio Dual.
  - Use el método de clasificación multiclase del tipo 1-contra-todos (1-vs-All).
- Pruebe el rendimiento del modelo obtenido utilizando el set MIT67-Test.zip. Reporte sus resultados.
- Pruebe distintos valores para el coeficiente de penalización de las variables slack. Comente sus resultados.
- Pruebe el rendimiento del clasificador en el set de entrenamiento. Nota diferencias respecto del resultado en el set de test, comente.
- ¿Qué categoría obtiene el mejor rendimiento?, ¿Cuál es el más bajo?. Indique razones que puedan justificar estos resultados. Para esto mire las imágenes reales, las cuales están en los archivos Image-Train.zip e ImageTest.zip.
- Finalmente, entrene una máquina de vectores de soporte en el set MIT67-Small-Train.zip y pruebe el modelo resultante en el set MIT67-Small-Test.zip. ¿Observa diferencias en el tiempo de ejecución y exactitud del modelo resultante?. ¿Cómo se compara con el rendimiento en el set completo de 67 clases?. ¿Qué sugiere para mejorar el rendimiento del clasificador en el set de 67 categorías?

3.4 Resultados Comparativos

Compare la operación y resultados de los distintos clasificadores evaluados en la tarea, fundamente sus observaciones. En su análisis considere lo siguiente:

- Facilidad de uso.
- Tiempo de proceso.
- Exactitud de la clasificación.
- Tipos de error.