CC6204: Deep Learning
Tarea 2: Clasificación y búsqueda por similitud de sketches usando redes convolucionales

Profs. Josrge Pérez & José M. Saavedra

mayo 2018

1. Objetivo

Familiarizarse con el uso de redes convolucionales para clasificación y extracción de características de imágenes.

2. Descripción

La tarea consiste de dos etapas: 1) entrenar dos tipos de redes convolucionales, una clásica y una ResNet, para clasificar dibujos hechos a mano (sketches) y 2) evaluar un sistema de recuperación por similitud usando características aprendidas por los modelos convolucionales entrenados.

2.1. Construcción de conjuntos de datos

Para crear los conjuntos de datos (de entrenamiento y test) usaremos el conjunto de datos público *QuickDraw* `https://github.com/googlecreativelab/quickdraw-dataset`, el cual contiene aproximadamente 50 millones de dibujos repartidos en 345 categorías. Sin embargo, en esta tarea utilizaremos solamente 100 categorías, que serán elegidas aleatoriamente. Así, los conjuntos de datos a construir son:

1. **Conjunto de Entrenamiento**: Para cada una de las 100 categorías definidas se deberá seleccionar aleatoriamente 1000 dibujos, que generarán un conjunto de datos con 100.000 imágenes.

2. **Conjunto de Prueba**: Para cada una de las 100 categorías definidas se deberá seleccionar aleatoriamente 50 dibujos aleatoriamente, que generarán un conjunto de datos con 5.000 imágenes.

2.2. Modelos a entrenar

A continuación de muestra la arquitectura de cada uno de los modelos a entrenar:

2.2.1. skNet

```
conv1_1[64] conv1_2[64] maxpool1,
conv2_1[128], conv2_2[128], maxpool2,
conv3_1[128], conv3_2[128], maxpool3,
conv4_1[256], conv4_2[256], maxpool3,
fc_1[1024],
fc_2[100]
```

2.2.2. skResNet

```
conv1_1[64], conv1_2[64], maxpool_1
conv2_1[64], conv2_2[64], residual_1 = conv2_2 + maxpool_1
conv3_1[64], conv3_2[64], residual_2 = conv3_2 + residual_1
conv4_1[128], maxpool_2
conv5_1[128], conv5_2[128], residual_3 = conv5_2 + maxpool_2
conv6_1[128], conv6_2[128], residual_4 = conv6_2 + residual_3
conv7_1[256], maxpool_3
conv8_1[256], conv8_2[256], residual_5 = conv8_2 + maxpool_3
```

```
conv9_1[256], conv9_2[256], residual_6 = conv9_2 + residual_5
conv10_1[256], maxpool_4
fc_1[1024]
fc_2[100]
```

- **Imagen de entrada**: Redimensionar a 128x128
- **Capa convolucional:** Incluyen ReLU + Batch-Normalization. Todas usan filtros de 3x3. Para una capa convolucional definida como conv[N], N indica el número de canales o filtros.
- **Capa fully-connected:** Ocupan ReLU, excepto la capa final que no requiere no-linealidad.
- **Capa max-pooling**: Usa regiones de 3x3 y saltos (strides) de tamaño 2.

2.3. Búsqueda por Similitud

En esta parte se deberá utilizar cada modelo entrenado como extractor de características. Para este fin, se ocupará la salida de la capa fc_1 como el vector de características. Por lo tanto el tamaño del vector de características para un sketch de entrada será de 1024. Se evaluará el mAP sobre el conjunto de datos de prueba. En este caso, cada imagen de test participa como consulta y se deberá evaluar la efectividad en recuperar el resto de imágenes de la misma clase, esta estrategia es conocida como *leave one out*. Utilizar L2 como función de distancia.

3. Resumen

- Construir conjuntos de datos, para entrenamiento y pruebas.
- Entrenar dos modelos para clasificación de sketches: skNet y skResNet.
- Evaluar los modelos sobre el conjunto de pruebas. Presentar el *accuracy* alcanzado para datos de entrenamiento y prueba.
- Utilizar los modelos en un contexto de búsqueda por similitud usando la capa fc_1 como extractor de características.
- Evaluar mAP sobre el conjunto de prueba.
- Realizar informe según esquema mostrado en la siguiente sección.

4. Esquema de Informe

1. **Abstract o Resumen**: es el resumen del trabajo.
2. **Introducción**: se describe el problema y el contexto de aplicación. (10 %)
3. **Desarrollo**: se describe el diseño e implementación de la solución.(40 %)
4. **Resultados Experimentales y Discusión**: se debe presentar los resultados y hacer un análisis de los mismos. (40 %)
5. **Conclusiones** (10 %)

5. Restricciones y Condiciones

1. NO se aceptan tareas sin informe.
2. NO hay atrasos.
3. La tarea puede ser realizada por equipos de 2 estudiantes.
4. Poner mucho esfuerzo en la redacción del informe.
5. La implementación de los modelos deberá ser desarrollada en TensorFlow.

6. Entrega

La entrega se realiza por u-cursos hasta el domingo 17 de junio, 2018, 23:50 hrs. Se debe incluir:

1. Código fuente (en Python)
2. Modelos entrenados
3. Informe