Machine Learning para el desarrollo de productos / Parcial I / 18-09-2020

Nombre(s) y Apellido(s): _____ Código:_____

En general, cuando se trabaja en proyectos de machine learning, una de las actividades que consume una gran cantidad de tiempo, es el pre-procesamiento de los datos. En particular en el caso de imágenes, normalmente, las imágenes recolectadas llegan desde diferentes fuentes así que para que sirvan a nuestro proyecto, estas deben ser al menos "limpiadas y estandarizadas". Existen pues, dependiendo del problema que se esté solucionando una variedad de técnicas que pueden ser usados para conseguir tal fin.

1. **Estandarizar imágenes.**

   Comúnmente las personas asocian la estandarización de imágenes al cambio de tamaño de las mismas. Sin embargo, la estandarización comprende varias técnicas que pueden ser aplicadas secuencialmente.

   a. <u>Cambio de tamaño (*resize*)</u>

   En algunos algoritmos de ML como es el caso de las CNN, es necesario cambiar el tamaño de las imágenes de entrada de tal forma que las dimensiones de las mismas sean unificadas. Aunque en principio parece una tarea sencilla. Existen varias técnicas.

   Para la siguiente imagen

   i. use la función ***cv2.resize()*** de ***opencv*** para cambiar el tamaño a la tercera parte del tamaño original. Use y compare al menos 3 métodos de interpolación.

      Ayuda: Una manera de comparar los métodos es usar el modo visual. Sin embargo existen métodos más formales. Uno de estos consiste en, una vez la imagen ha sido cambiada de tamaño, volverla a su tamaño original y entonces proceder a compararla con la imagen original calculando el **PSNR** (*peak signal to noise ratio*).

      Despliegue las imágenes e imprima el PSNR de cada una.

ii. ¿Qué sucede si su modelo necesita de imágenes cuadradas? En este punto existen dos posibilidades: una es mantener el ***aspect ratio***, la otra es adicionar ***padding*** a la nueva imagen. Para la siguiente imagen:

una vez escogido el mejor método de ***resize***. Realice como imágenes independientes, estas 2 operaciones y despliegue sus resultados en pantalla.

Ayuda: Para el caso de padding use la función ***copyMakeborder()*** de ***opencv***. Tenga en cuenta además que el padding "rellena" el faltante de la imagen con pixeles blancos o negros. Sin embargo en tareas de clasificación el ***padding*** usando la reflexión de la imagen parece dar mejores resultados.

iii. Haga una pequeña búsqueda acerca de esta pregunta y respóndala: ¿Que pasa si las imágenes que van a alimentar su sistema ML son todas de diferentes tamaños? ¿Cuál es la mejor opción para hacer ***resize***?

b. <u>Normalizar imágenes a nivel de pixeles.</u>

La normalización de las imágenes consiste en cambiar el rango de los valores de intensidad de los pixeles. Esta normalmente es usada para incrementar el contraste en las imágenes, lo que a su vez permite mejorar la extracción de características o la segmentación de imágenes en ML.

i. Use la función ***normalize()*** de ***opencv*** para normalizar la siguiente imagen:

ii. Estudie y use la función ***equalizeHist()*** de ***opencv***. Muestre las imágenes original, normalizada y equalizada e imprima los valores de media y desviación estándar para ellas, antes y después

de la normalización. Comente (de forma técnica) acerca del resultado obtenido.

2. ***Data augmentation*.**

Esta es una técnica que involucra aumentar la cantidad de imágenes existentes con dos objetivos: el primero, generar más datos cuando se cuenta con una cantidad limitada de datos a la entrada; el segundo, prevenir el ***overfitting***. Existen muchos tipos de transformaciones que pueden ser realizadas bajo esta técnica. A continuación algunas de las más usadas.

Use la siguiente imagen:

a. **<u>Rotaciones</u>**.
   i. Que posibles rotaciones puede lograr usando la función ***flip()*** de ***opencv***. Muestre los resultados en pantalla.
   ii. Usted puede obtener cualquier grado de rotación sobre sus imágenes haciendo uso de las funciones ***getRotationMatrix2D()*** y ***warpAffine()***. Haga uso de estas para obtener y mostrar en pantalla rotaciones de la imagen cada 45º.
   iii. Como usted pudo notar existe un problema a la hora de rotar las imágenes y es que estas parecen perder parte de los bordes de las imágenes según la rotación dada. Encuentre y muestre en pantalla una manera en que la imagen no quede recortada.

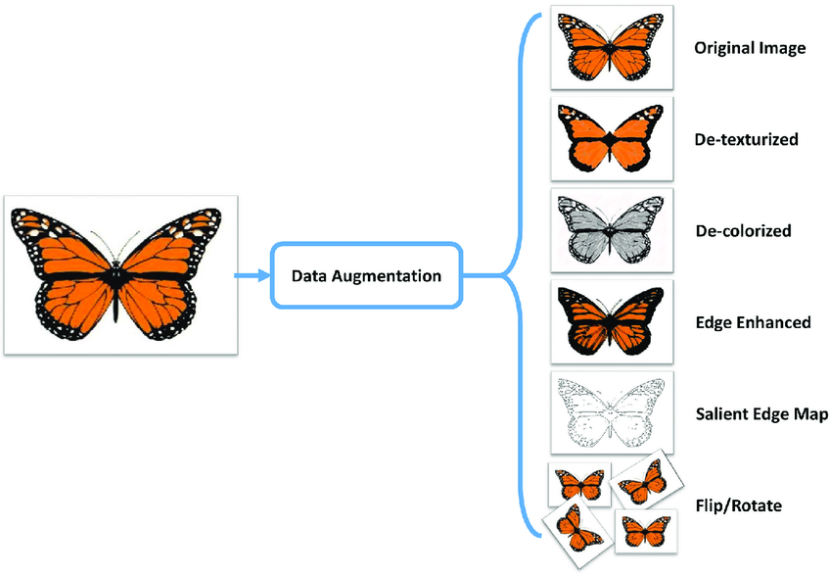
      Ayuda: Lea acerca de lo que significa ***CANVAS*** y busque como manipularlo con ***opencv***.

   iv. Haga una pequeña búsqueda acerca de esta pregunta y respóndala: ¿los grados de rotación de una imagen, dependerán del tipo de problema a resolver o se puede aplicar cualquier tipo de rotación aleatoriamente? ¿Porque?

b. <u>Traslaciones.</u>
   i. Partiendo de la siguiente imagen con altura ***H*** y ancho ***W***, cuyo pixel central es ***(x,y),*** Realice y muestre en pantalla las siguientes traslaciones: $\left(x-\frac{W}{2},y\right)$, $\left(x+\frac{W}{2},y\right)$, $\left(x,y-\frac{H}{2}\right)$, $\left(x,y+\frac{H}{2}\right)$

ii. Haga una pequeña búsqueda acerca de esta pregunta y respóndala: ¿Cuándo o porque es beneficioso usar la traslación dentro de un proyecto de ML?

c. Existen otras operaciones para realizar ***data augmentation***, como lo son: ***de-texturized***, ***de-colorized***, ***edge enhanced*** y ***salient edge map***. Observe la siguiente imagen como ayuda visual de los conceptos aquí expuestos y haga uso de las funciones aprendidas en clase, para realizar y mostrar en pantalla las operaciones antes mencionadas.



d. **Ruido**.

Muchas veces, se hace necesario un proceso preliminar de eliminación de ruido sobre las imágenes de entrada al sistema. Sin embargo, el caso contrario, es decir, adicionar ruido a una imagen es algo vital para darle un mayor grado de robustez a nuestro sistema, una vez este se encuentre en la etapa de producción. Haga uso de ***Python*** y ***OpenCV*** para generar dos tipos de ruidos sobre cualquier imagen que usted desee: ruido gaussiano y ruido de sal y pimienta. Muestre los resultados en pantalla.

Este parcial está basado en las ideas expresadas en la siguiente lectura:

https://machinelearningmastery.com/best-practices-for-preparing-and-augmenting-image-data-for-convolutional-neural-networks/

donde varios importantes autores muestran que técnicas usaron para preparar sus datos en tareas de clasificación de imágenes, sobre cuatro grandes modelos, como los son: SuperVision/AlexNet, GoogLeNet/Inception, VGG y ResNet.

Generalidades:

- Presente un archivo (código –texto) XY.ipynb. XY, reemplácelo por su nombre y apellido.
- El archivo debe mostrar claramente la división de los resultados para cada pregunta.
- El archivo debe ser autocontenido, cada estudiante debe cerciorarse que las imágenes queden “linkeadas” al programa.. Es decir, no es función del profesor cargar las imágenes, este simplemente se limitará a correr los diferentes códigos dentro del mismo.
- Cargue el archivo a classroom (no olvide darle enviar). .
- **Fecha de entrega: Lunes 21 de septiembre de 2020. Hora límite: 6:30 P.M**
- El archivo debe ser subido a classroom. (No debe enviarlo al correo del profesor, lo cual acarreara una rebaja de 2 puntos).
- La entrega de la tarea después de la hora límite, acarrea la pérdida de un punto por cada hora o fracción.