Asignatura

Sistemas Interactivos Inteligentes

Práctica 2

Unidad IV

CLIP vs PRE-CLIP

MEMORIA

LUCÍA NISTAL PALACIOS

El objetivo de esta práctica era hacer una comparativa entre dos modelos (CLIP y Pre-CLIP) para relaciones entre imágenes y texto. El primer método, Pre-CLIP que utiliza modelos independientes entrenados por separado para procesar por un lado las imágenes y por otro los textos o descripciones. El segundo modelo es CLIP (Contrastive Language-Image Pre-training) que es un modelo desarrollado por OpenAI que aprende las relaciones entre imagenes y textos mediante un entrenamiento de estas.

Para realizar la práctica, primero se creó un **dataset** de **20 imágenes** divididas en **4 categorías** diferentes: Animales, Colores, Ropa y Colegio. En un primer momento 1 de las categorías fue Estaciones habiendo paisajes de primavera verano otoño e invierno pero no funcionaba adecuadamente ya que los paisajes se confundían con los fondos de los animales y con los colores, no demasiado pero se cambió la categoría para hacerlo más sencillo. Se eligieron estas para que hubiera variedad y diferencias semánticas entre los grupos y para que fuera más facil el análisis posterior de cómo funciona cada modelo. Y todas las imágenes fueron descargadas de pexels.

La categoría **Animales** incluye cinco imágenes diferentes: un pájaro, un gato naranja marrón y blanco, un perro negro grande, una serpiente con escamas amarillas y negras, y un conejo marrón claro. La categoría **Colores** presenta cinco imágenes simples con fondos amarillo, rojo, rosa, verde y azul. Estas imágenes son muy útiles para evaluar si los modelos pueden relacionar correctamente la parte visual “sencilla” con su etiqueta correspondiente de color. La categoría **Ropa** contiene objetos cotidianos de vestir: una chaqueta naranja, zapatos azul oscuro, una camiseta negra, un sombrero beige con cinta negra, y pantalones verdes. Esta categoría permite evaluar cómo los modelos manejan objetos con formas y texturas específicas. Finalmente, la categoría **Colegio** incluye material escolar común: una calculadora científica, una mochila negra, un lápiz de madera naranja y rojo, un reloj con números de varios colores, y una goma de borrar.

Para cada una de las 20 imágenes, hay una descripción en inglés. El objetivo era crear **captions** que reflejaran las imagenes sin ser ni demasiado genéricos ni demasiado detallados. Algunos ejemplos de descripciones fueron estas, para el pájaro: "A colorful bird, green, blue, and purple with brown wings and blurred background", y para el gato: "an orange, brown and white cat on top of a wooden table", que especifica tanto los colores como donde esta el gato. En el caso de los colores, las descripciones son muy simples como "a photo of the color yellow", ya que estas fotos son directas. Para los objetos más complejos como la ropa o el material escolar, hay características visuales relevantes como "A pair of dark blue sneakers on a white surface" o "A scientific calculator on a wooden surface".

Pre-CLIP, se llevó a cabo con dos modelos. Para el procesamiento de imágenes ResNet18, una red neuronal pre-entrenada en ImageNet. ResNet18 es una versión sencilla de la ResNet pero suficiente para extraer características significativas.

Para el procesamiento de texto, utilicé el modelo recomendado sentence-transformers/distiluse-base-multilingual-cased, un transformer optimizado para generar embeddings de oraciones.

El proceso de extracción de **embeddings** de **imagen** comenzó cargando ResNet18 pre-entrenado desde torchvision. Para cada imagen del dataset, Para los **embeddings** de **texto**, el proceso fue más directo. Cargué todas las descripciones del CSV y las pasé en un batch único al modelo de sentence-transformers usando su método encode() que maneja la tokenización, el procesamiento por el transformer y la generación de los embeddings. Y el resultado es un tensor de forma (20, 512) con un embedding por cada caption.

Una vez obtenidos ambos conjuntos de embeddings, implementé el cálculo de similitud coseno en el script compute_preclip_similarity.py, que sirve para comparar vectores en espacios de alta dimensión y se calcula como el producto punto entre dos vectores normalizados.

Para garantizar la estabilidad numérica, se normalizan ambos conjuntos de embeddings dividiendo cada vector por su norma. Luego se calcula la matriz de similitud completa mediante multiplicación matricial, obteniendo una matriz 20x20 donde cada elemento (i,j) representa la similitud entre la imagen i y el caption j.

Es importante decir que en Pre-CLIP, los embeddings de imagen y texto provienen de espacios vectoriales **completamente distintos**, entrenados con objetivos diferentes y en datos distintos. Esta es la razón por la que los resultados no son buenos

Para el enfoque **CLIP**, utilicé el modelo recomendado openai/clip-vit-base-patch32 disponible en la biblioteca Transformers de HuggingFace. Una característica fundamental de CLIP es que tanto la imagen como el de texto fueron entrenados conjuntamente sobre un dataset de millones de imágenes y textos recopilados de internet. Este entrenamiento conjunto es lo que permite a CLIP proyectar ambas modalidades en un espacio común donde la similitud vectorial tiene significado semántico real. Esta es también la explicación de los resultados finales, ya que son muy buenos.

En el archivo extract_image_embeddings.py, se procesa cada imagen usando processor(images=image, return_tensors="pt")y luego se extraen sus features mediante model.get_image_features(). De forma que devuelve directamente los embeddings y el proceso funciona igual para el texto en extract_text_embeddings.py, se usa processor(text=texts, return_tensors="pt", padding=True) y luego de model.get_text_features().

El cálculo de **similitud** en CLIP sigue el mismo proceso que en Pre-CLIP (similitud coseno mediante normalización y producto matricial), pero con la diferencia de que aquí existe una similitud que tiene significado semántico real y se debe principalmente a que CLIP fue entrenado explícitamente para que pares imagen-texto correctos tengan alta similitud coseno, mientras que pares incorrectos tengan baja similitud.

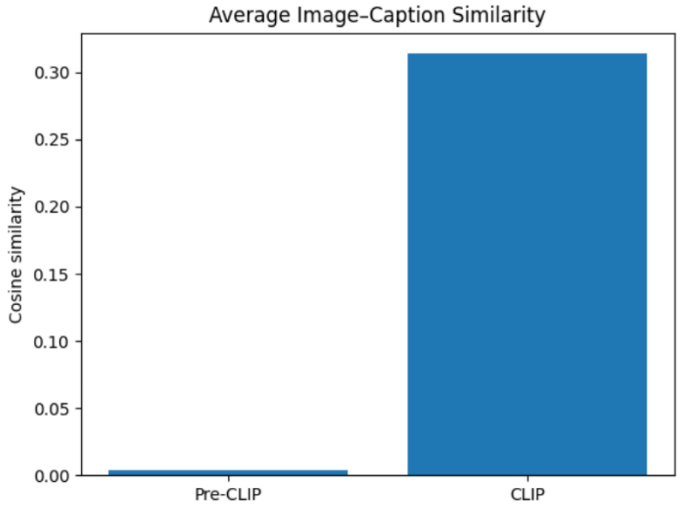
Los **resultados** obtenidos demuestran claramente la **superioridad de CLIP sobre Pre-CLIP**. La similitud media fue de 0.004 para Pre-CLIP y 0.314 para CLIP.

```
root@aa3b7019f296:/opt/project/src# python evaluate.py
 Mean similarity (image-caption pairs):
Pre-CLIP: 0.004
CLIP:     0.314
Saved plot to ../results/similarity_comparison.png
root@aa3b7019f296:/opt/project/src# ls ../results
clip_similarities.csv        preclip_similarities.csv   text_embeddings_preclip.pt
image_embeddings_clip.pt     similarity_comparison.png
image_embeddings_preclip.pt  text_embeddings_clip.pt
root@aa3b7019f296:/opt/project/src#
```

En esta captura se ve cómo además de generar la **imagen similarity_comparison.png,** después miro que se generó en mi **carpeta de results**, y ahí se encuentran todos los archivos generados por el sistema, los embeddings de imagen y texto cada modelo, las similitudes y una evaluación entre ambos.

La **diferencia entre CLIP y Pre-CLIP** no es solo un dato sino que nos da información muy importante. En Pre-CLIP, las similitudes son casi indistinguibles del ruido, de hecho muchos valores en la matriz son negativos o casi cero, lo que indica que los espacios vectoriales de imagen y texto no tienen ninguna alineación significativa. Una imagen puede tener similitud 0.03 con su caption correcto pero también 0.05 con un caption completamente irrelevante. Y mientras CLIP muestra un patrón claro donde todas las similitudes en la diagonal son positivas y bastante más altas que la mayoría de las similitudes fuera de la diagonal Y además al examinar las matrices de similitud **por categorías**, se observan patrones interesantes. En la categoría Colores, CLIP logra similitudes particularmente altas, esto tiene sentido ya que los colores son conceptos bastante simples con una correspondencia imagen y texto directa. La categoría Ropa muestra similitudes buenas pero no increibles y hay algunas confusiones con la categoría Colores, especialmente para prendas con colores distintivos lo que tiene sentido ya que los captions de ropa mencionan colores, y puede crear una especie de superposición semántica. Finalmente, la categoría Colegio presenta resultados muy buenos también. La calculadora y el lápiz obtienen buenas similitudes, y la goma de borrar logra la similitud diagonal más alta de todo el dataset con 0.319, posiblemente porque su descripción es más específica.

La **comparación visual** mediante el gráfico de barras generado por evaluate.py es muy importante. La barra de Pre-CLIP es prácticamente nula comparada con la de CLIP, reflejando la gran diferencia que tienen de 0.004 vs 0.314 en similitud.



Esta diferencia no llama tanto la atención teniendo en cuenta que **funcionan de maneras diferentes.** Pre-CLIP utiliza modelos entrenados independientemente con objetivos distintos, ResNet fue entrenado para clasificación de imágenes en 1000 categorías de ImageNet, mientras que el modelo de texto fue entrenado para capturar similitudes semánticas entre oraciones. Por lo que no se pueden esperar grandes resultados. CLIP, por otro lado, fue diseñado exactamente para esto y su entrenamiento le enseñó que los conceptos relacionados están cerca y conceptos no relacionados están lejos.

Siguiendo las recomendaciones de la práctica ecidí utilizar modelos completamente pre-entrenados y sin realizar fine-tuning, ya que el objetivo era comparar las capacidades de estos modelos en su forma comun, además de que requeriría un dataset mucho más grande. Además los modelos pre-entrenados, especialmente CLIP, ya demuestran grandes capacidades sin necesidad de esa adaptación específica. Luego organicé el código en archivos separados para cada parte del proyecto, como la extracción de embeddings de imagen y texto, el cálculo de similitudes y la evaluación. Esta lo hice para que no fuera lioso y tiene varias ventajas que son que permite ejecutar cada parte de forma findependiente y además facilita la reutilización de embeddings sin tener que volver a calcularlos. Además, hay un archivo utils.py que tiene funciones como la cosine_sim que hace el cálculo de similitud coseno de forma robusta y evitando divisiones por cero.

Esta práctica demuestra **la ventaja de CLIP, sobre Pre-CLIP.** También es cierto que había ciertas limitaciones como el uso de solo 20 imágenes que esta bien para trabajar pero realmente no es un dataset demasiado grande, si fuera más grande quizás se podría hacer un análisis comparativo más robusto.

Los **resultados** muestran que no basta con tener embeddings de alta calidad para imágenes y texto por separado para conseguir una alineación multimodal efectiva**.**