ANTALYA BİLİM ÜNİVERSİTESİ

2024-2025 SPRING SEMESTER

Introduction to Robotics

-EE422/CS421-

WEEK 3

VISION LANGUAGE MODELS FOR ROBOTIC PERCEPTION

Why CLIP ?

In this week we looked at the some methods we can use in our project. OpenAI'S CLIP took our attention. We dig dive into this method. So finally we decide to use CLIP in our project as a VLA. These are the main reasons to why we choose CLIP ;

1-Connects images and texts in a meaningful way, enabling visual understanding through natural language.

2-CLIP trained on 400 million text and image pairs collected from the internet and CLIP is trained with **"contrastive learning"** , meaning it learns to match correct image-text pairs while distinguishing them from incorrect ones.

Also CLIP has multi language feature too. That means CLIP understands the languages(in text form) other than the english ie. German, Spanish...

How CLIP Works ?

CLIP has 2 main parts. Image and Text Encoder. Both encoders convert their input into a feature vector. (Image to vector and text to vector)

Than CLIP calculates the cosine similarity between these vectors.

What is Cosine Similarity ?

Cosine similarity measures how similar these vectors are based on the angle between them.

- If the angle between them is **0°**, they point in the **same direction** → similarity = **1**
- If the angle is **90°**, they are **completely different** → similarity = **0**
- If the angle is **180°**, they point in **opposite directions** → similarity = **-1**

How Image's and Text's turned into a vector ?

There are some tools for that. For turn a image to a vector ,used tools are ResNet and for turn a text in to a vector, Transformer is used.

What is ResNet and how ResNet transforms image to vector ?

ResNet is a deep convolutional neural network (CNN) that solves vanishing/exploding gradients in training very deep networks.As networks get deeper, gradients vanish (become too small) or explode, and layers forget how to learn.

To solve that problem ResNet uses "Residual Blocks". A Residual Block makes; instead of directly learning the complex function to be learnt, it learns its difference with respect to the input.

About Model

A small dataset is a collection of data that typically focuses on a specific task and contains fewer examples compared to large datasets. In robotic tasks, it is used in the process of fine-tuning and allows the model to better adapt to a specific task.

This dataset is included in the training phase of the project. For example, if a robot needs to place objects in a specific way, the small dataset includes examples specific to this task. By training the model with this data set, it can make more precise and accurate decisions.

1. Detection of Linguistic Commands: The user expresses the task that the robot needs to perform in natural language (e.g., "move red block to the right").
2. Interpretation of the Command: The model analyzes the given command and determines which objects, locations, and actions are relevant.
3. Action Planning: The robot determines the appropriate movements to carry out the command. At this stage, the accuracy of the fine-tuned model is of great importance.
4. Execution: The robot physically executes the specified actions.

The stack-block-pyramid-seq-seen-colors task is a task that aims to have the robot create a pyramid using blocks of specific colors. The demos.py file located in cliport/data is used to generate the training data for this task.

Here's how it works:

1. Identification of Blocks: The blocks to be used in the task are determined and their colors are assigned.
2. Placement Order: A pyramid is formed by stacking the blocks on top of each other in a certain order.
3. Linguistic Commands: The model uses linguistic commands to place blocks correctly (e.g., "put the red block on top of the blue block").
4. Training Data Generation: The demos.py file creates the data set required for this task and allows the model to learn.

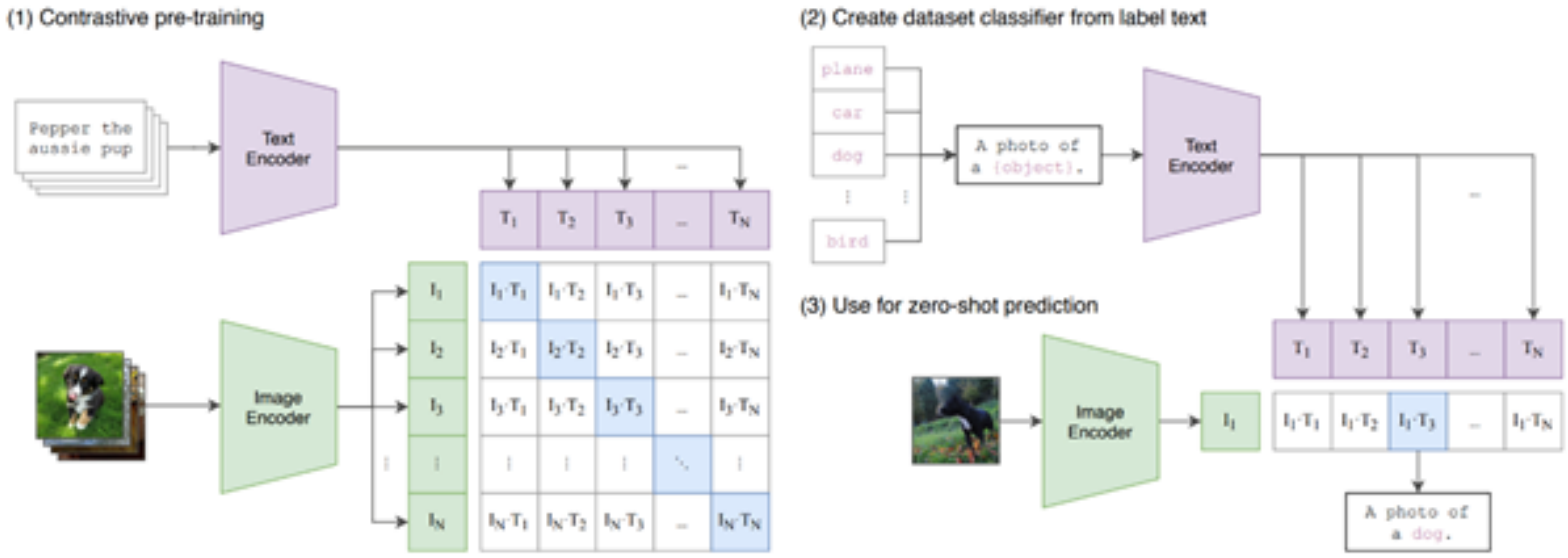


Figure 1. Summary of our approach. While standard image models jointly train an image feature extractor and a linear classifier to predict some label, CLIP jointly trains an image encoder and a text encoder to predict the correct pairings of a batch of (image, text) training examples. At test time the learned text encoder synthesizes a zero-shot linear classifier by embedding the names or descriptions of the target dataset's classes.

3. How does the CLIP model perform zero-shot classification?

Question: How can the CLIP model classify a zero-shot image without any training data?

Answer: As shown in steps (2) and (3):

1. First, text descriptions are created from class labels ("dog", "plane", etc.): "A photo of a {label}".
2. These texts are passed through the text encoder to obtain vector representations.
3. The image in the test is passed through the image encoder.
4. The cosine similarity between the image vector and the text vectors is calculated.
5. The text with the highest similarity is estimated as the class.

4. How does the CLIP model differ from traditional image classification models?

Question: How is CLIP different from standard image classification models?

Answer: Traditional models use a linear classifier that is trained for predetermined classes. CLIP, on the other hand, dynamically generates this classifier with text descriptions. Thus, CLIP can also recognize classes that have never been seen (zero-shot).

5. What does the matrix in the image represent?

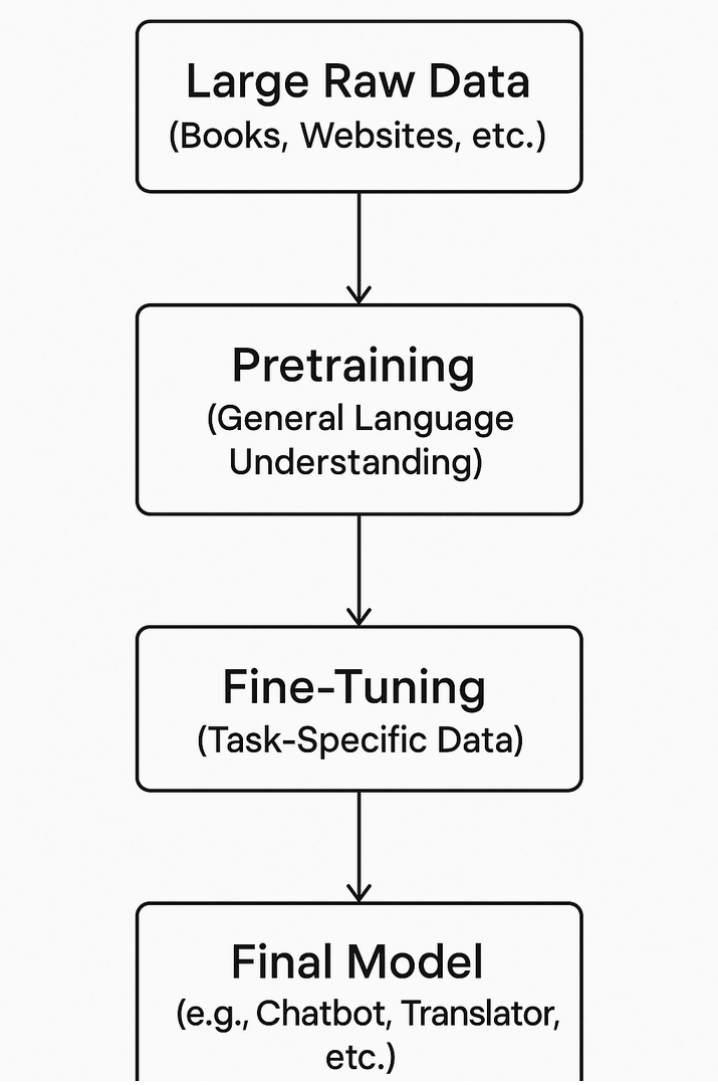
Question: The large square matrix in the image ($I_1 \cdot T_1$, $I_1 \cdot T_2$, ...) What does it refer to?

Answer: This matrix shows the inner product (dot product) or similarity between each image vector (I_1, I_2, ...) and each text vector (T_1, T_2, ...). During training, these values are maximized for correct (visual, text) pairs, and minimized for false matches.

Agah
Pretraining is a widely used method to improve machine learning models by training them on massive datasets such as books and websites. The purpose of this phase is to provide general language and structural understanding. Once the model is pretrained, it is fine-tuned on task-specific datasets to perform more accurately on defined tasks.

In this project, we are not limited to textual data. We will also integrate Vision AI and Spoke AI modules to analyse visual and audio inputs, respectively. Vision AI will handle image-based data such as facial expressions or environmental objects, allowing for deeper contextual awareness. Spoke AI will process voice data, converting spoken language into text and interpreting its content for further analysis.

This multi-modal architecture allows the system to go beyond simple text-based processing, enabling more accurate, flexible, and intelligent interactions across different types of input.


Large Raw Data
(Books, Websites, etc.)
Pretraining
(General Language Understanding)
Fine-Tuning
(Task-Specific Data)
Final Model
(e.g., Chatbot, Translator, etc.)