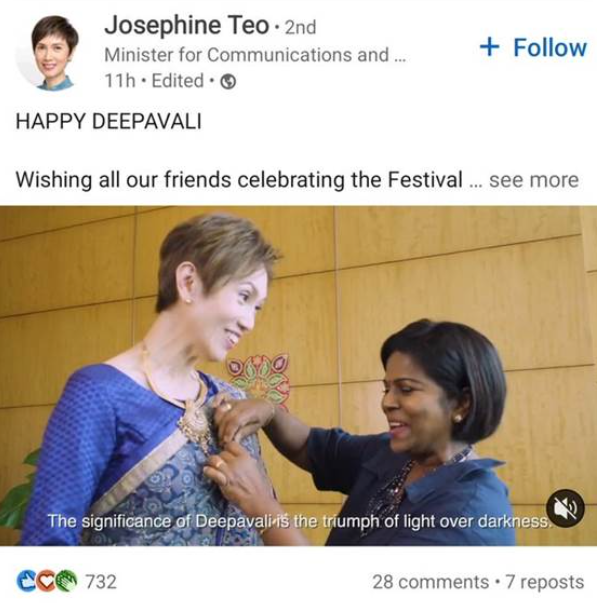
**Task Outline (just for my own reference)**

Requirements of task: Build an AI framework to

1. Identify target ***in***(??) the image (Person, Organisation, Country, Race, Religion)
2. Classify sentiments (Positive, Negative or Neutral)

Writeup should detail:

* Details on workflow, pipeline diagram, identify inputs and outputs in the modelling approach
* Include names of techniques (i.e. Face Recognition with FaceNet/InsightFace model, 3-class classification with Support Vector Machines using Scikit-Learn package)
* A short write-up on your thought-process, ideation or methodology on why you think this would work.

**Base Assumptions**

1. All inputs will be of the form of a screenshot
2. All inputs are given in the form of text for the caption and an image for the actual post, such that we do not need to use OCR on the image to detect where the caption ends and where the image starts. This also implies no videos as inputs. In short, we assume the data has already been pre-processed.
3. Part 1 is interpreted as what the image is talking about, what its topic is or what it is commenting on.

Input: Text (from caption), Image (from post)

Output: Keywords about target based on the given list in dictionary/json format (Part 1), followed by “positive”, “negative” or “neutral” (Part 2) separated by a comma.

**Part 1**

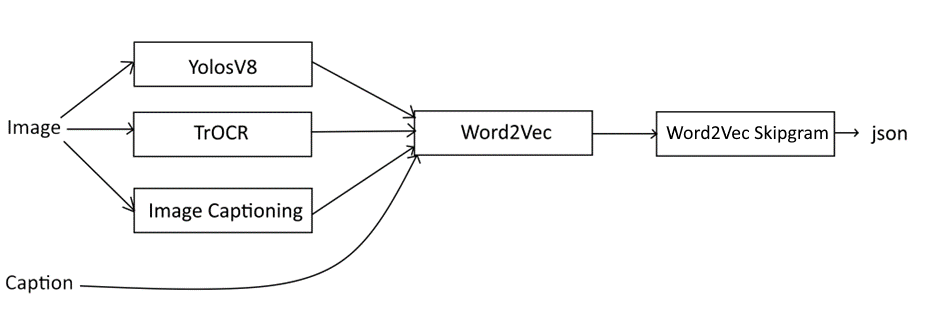
**Thought process**

Since I started work on the second part first, naturally I would try inputting the image into Gemini and observing if it could tell me what the post’s target was. After messing around with the prompt, I arrived at the prompt “This post consists of an image and the caption "{caption}". I have supplied you with the image below. Please describe the target/topic/person that is being referenced in the post, in the format (Person, Organisation, Country, Race, Religion), with one-word answers. For example, if the image was one of a cat eating meat, and the caption was "love cats", you should respond with (Cat lover, None, None, None, None).”, which will be input into the API along with the image, followed by the prompt "{output}. Given this text, remove the top part and the bottom part such that only the part with person:, orginsation:, country:, race: and religion: is left. Then, change it into a json format”. This returns a string in json format, which can be printed out or manually converted into a dictionary.

But since I am unsure whether this count as being an AI framework, I will be attempting to propose another method. (I am running out of time so I am unable to do this for part 2)

**Description of workflow**

Instead, I have come up with this:



First, we use YolosV8 from ultralytics on github to detect objects in the image. We now have a list of objects in the image. Next, we use TrOCR model on HuggingFace to recognise any text in the image. Finally, we then use pycocotools library to generate a description of the image.

We repeat the process of capturing information from the image so many times since it is possible for details to be missed out. For example, a sarong could be detected as a skirt from the YolosV8 model, or not even picked up at all, so hopefully the image description from pycocotools will capture it.

We then pipe all of these outputs and the image caption we have been provided earlier into the pretrained word2vec on gensim, word by word. We now have a list of vectors, which are the words we had before.

Finally, we then input every combination of the vectors in the list and the vectorised words in (Person, Organisation, Country, Race, Religion) into skipgram from gensim’s word2vec to produce the top 5 most correlated words.

After that, we iterate through every produced word and check if they are considered as strongly correlated (using consine similarity) to any of the words in (Person, Organisation, Country, Race, Religion) by the method we chose from the step before. If there is a match, we append the word to a dictionary with the key of whatever matches, for example dictionary[‘religion’].append(word). This gives us the first part of our output.

**Assumptions**

This assumes that:

1. The pretrained word2vec has all the words we input in their database
2. We can somehow find data to train the neural network or can find that difference for related vs unrelated words

**Part 2**

**Thought process**

My initial idea was to find a model (maybe Image Captioning Pytorch) to describe the image and input the output from the first model into a second one (possibly N-gram Statistics Package) to find correlation words in the first output and the caption that have a strong correlation with both the post caption and the image description, which will serve as the topic. Then based on the topic perform sentiment analysis on both the image and the caption to output positive, negative the AND operator on them, returns -1 by default when outputs do not match unless it is 0 and 1, which returns 1 (-1 for negative, 0 for neutral, 1 for positive) and output the corresponding word for each number. There are likely many errors here, but one of the more obvious ones is that something that could return positive and negative could be captions being “I hate XXX for being so silly” and the image is of the two of them posing for a photo, which would be evaluated as positive and negative, which returns negative but should return positive.

Only after a bit of searching did I realise that finding the models that fits our exact purpose were quite infeasible, especially for the sentiment analysis based on topic since that likely requires me to train my own rather than take one that is pretrained, and there are LLMs that could do that way better since they are trained on large amounts of data so they can detect language features such as sarcasm, as well as being able to perform sentiment analysis. Sentiment analysis is also quite broad and more specific models for your own use require you to train on your own, and I have no clue where to get vast amounts of training data from. Based on the little testing I could do, Gemini is able to detect the tone of the post (positive, negative or neutral).

**Description of workflow**

With that in mind, we will use the Gemini Advanced model for this task. We upload the image into the same folder as the script and open it using the PIL library. We input that along with the text into Gemini API in the format “The post below was captioned {caption}. Is this post positive, negative or neutral?” From generated output, obtain every instance of “positive”, “negative” and “neutral” and add into list. Return the last element of the list. Alternatively, another way to determine the output is to sum up all instances of each word (positive, negative, neutral) and return the one with most occurrences. This gives us the second part of our output.

**Assumptions**

This then assumes that:

1. The final judgement is at the last lines of the text generated. This is due to posts such as example 2 where there is sarcasm so with the caption by itself, it can be deemed as positive and the generated text will also describe it as such, hence adding an instance of “positive” to the generated output. The final judgement was only given at the end of the generated output.
2. The topic does not involve the keywords, those being positive, negative or neutral (for example, a post talking about positive/negative charge ions).