III. Model documentation and write-up

Information included in this section may be shared publicly with challenge results. You can respond to these questions in an e-mail or as an attached file. Please number your responses.

1. Who are you (mini-bio) and what do you do professionally?
   I'm a ML engineer with two years of experience working at a startup that builds mobile surveillance camera apps. My expertise is mainly on computer vision and image processing.

2. What motivated you to compete in this challenge?
   The main reason I will participate in this challenge is to challenge myself to solve challenging problems I normally don't encount in my day job. And the chance to work with models that deal with both image and text is pretty rare.

3. High level summary of your approach: what did you do and why?
   - First using OCR and inpaint model to find and remove the text from the image. This will improve the quality of both object detection and web entity detection.
   - Using the clean meme image to do bottom-up-attention feature extraction, web entity detection, human race detection. Those tags will give the transformer models much more diverse information to work with.
   - Train the following models:
     - extended VL-BERT
     - UNITER-ITM / VILLA-ITM
     - vanilla ERNIE-Vil
   - Average all predictions
   - Apply simple rule-base racism detection using meme text, race tag, entity tag, skin tone.

4. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.
   These are the 3 most impactful parts of my solution, but the code is not included because they cannot be formatted in a short and self-explanatory manner.
   - Explicitly link image regions to entity tags, text description by extending VL-BERT's language-vision embedding framework. This gave VL-BERT a chance to learn the relationship between entity tags and text by using the image region as anchor.
   - Reuse of UNITER's ITM pretrained binary classifier head gave me better performance over random initialization. By the characteristics of meme, normality when text description is matching the image to some degree it will have a higher chance of being non-hateful. So I use class-1(image and text match) of ITM head as a non-hateful meme class, class-0 as hateful meme.
   - Using inpainted meme images that don't have the text on it improves both OID detector and web entity detection's performance.

5. Please provide the machine specs and time you used to run your model.

`GCP n1-highmem-16` instance

- CPU (model): 16 Core Intel CPU
- GPU (model or N/A): 4x NVIDIA T4
- Memory (GB): 104
- OS: Ubuntu 18.04.5 LTS
- Train duration: 4 hours for all VL-BERT, 3 hours for UNTIER + VILLA, 4 hour for ERNIE-Vil
- Inference duration: Every model take less than 5 mins on a sample set of 2000.

6. Please list any external data or pre-trained models you used to develop your winning submission..
    - Res2Net FasterRCNN Patch Detector
    - DeepFillV2
    - InceptionV2 FasterRCNN OpenImageV4 object detector
    - FairFace race and gender classifier
    - VisualGnome FasterRCNN

7. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?
    - Adapter-transformer on VL-BERT
    - Oscar
    - Oscar fine-tune with image feature extractor(ResNet101 Faster-RCNN)
    - LXMERT fine-tune with image feature extractor
    - Apply entity tag on UNITER and Oscar
    - Apply super-resolution to image before extract feature
    - Throw more pretrain data at VL-BERT
    - SWA
    - ROC-Star loss
    - Contextual text augmentation
    - Text only toxic comment classifier combine with VL-BERT
    - VILLA's adversarial fine-tuning
    - Visual relationship detection as feature
    - Object attribute from the Vision Genome object detector as feature

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

I'm only using **Streamlit** to inspect paired meme images and text with corresponding entity tag and object bounding box.

9. How did you evaluate performance of the model other than the provided metric, if at all?

Not at all.

10. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

- Somehow when inference on the test set VL-BERT's DataLoader will require 4GB of shared memory. Not sure will this cause any issue on another machine beside GCE n1-highmem-16 VM.
- The Google Vision python package will randomly run into segmentation error if I process too many samples in one go. I currently have a workaround of using a shell script to restart the same python script every 20 samples. But I am not 100% confident that we won't encounter the same issue again.
- There is some non-deterministic behavior in the VL-BERT training pipeline and data preprocessing pipeline that may influence the final result.

11. Do you have any useful charts, graphs, or visualizations from the process?
    I don't have any visualizations that provide key insight that ends up improving the final score.

12. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?
    I was attempting to integrate various techniques from the file of open domain QA. Mainly using knowledge graph and GNN to help finding negative relations between entities appeared in the meme. Approach like MHGRN is promising research to directly combine pretrained multi-modal transformer models with knowledge graph. But there will be really challenge to extract reasonable sized sub-graph that contain key knowledge from source like wiki-data or ConceptNet.

13. Provide a link to the GitHub repository for your model and source code.
    https://github.com/HimariO/HatefulMemesChallenge

14. Provide a link to your published academic explanation of your solution.
    https://arxiv.org/abs/2012.08290