



Practicum 2 - Visual Question Answering

This is a group assignment.

Due: 17 November, 10:15 AM

In this practicum you will build a multimodal deep learning model trained on data that comes from two different sources - images and text. You will implement all the skills you learned on transfer learning for computer vision and natural language in the last 5 weeks.

Practicum 2 will put to the test the skills you have acquired in Module 2. The problem was designed so that you will need to do data processing and transfer learning jointly for both images and text. The dataset is large and this will help you in real life projects that you might encounter in the future.







We are deliberately not giving **specific instructions**, unlike the previous homework, so that the problem's overall structure and dynamics are understood and allowing you to be creative and investigative. This is a challenging task, where you will need to work hand in hand as a team, filling in a software development team's shoes.

Background: Visual Question Answering System

Deep learning models are generally designed to perform one task from one source of data (e.g. image classification or object detection from images in computer vision tasks; or text classification or question answering from text in natural language processing tasks). There are times when your dataset consists of both images and text that are related and you will want to combine computer vision and natural language tasks.

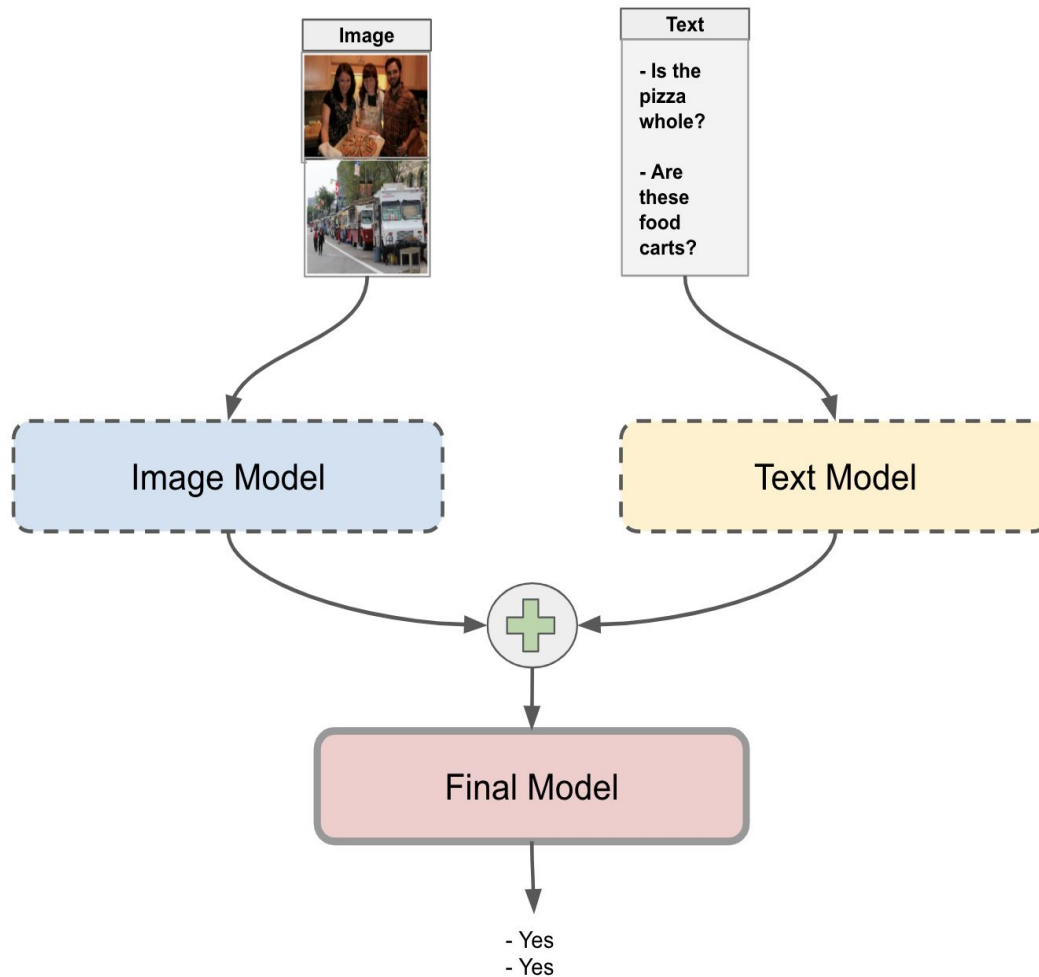
In Module 2 we covered transfer learning for various tasks in both computer vision and natural language. In this practicum you will use what you learned and create a combined model that performs both these tasks. The following images are from the VisualQA dataset and show images and corresponding question answers. **Your goal will be to use the image and question as inputs and predict the answer as output.**



		
Question: Is the pizza whole? Answer: yes	Question: What color is the sink? Answer: blue	Question: Are these food carts? Answer: yes
		
Question: How many stories is the building on the left? Answer: 3	Question: What color hat is the person on the left wearing? Answer: black	Question: What color is the ball? Answer: yellow

VisualQA Dataset [Reference](#)

You will need to build a multimodal model that can take both images and text questions as input and predict the answer. A high level flow of the required tasks for this practicum are given in the diagram below.



Dataset

The data for this practicum is from the [Visual Question Answering dataset](#).

VQA is a dataset containing open-ended questions about images. These questions require an understanding of vision, language and commonsense knowledge to answer. The dataset is a combination of the [Common Objects in Context](#) (COCO) dataset with question and answers added to them:

- Over 120,000 images (COCO)
- At least 3 questions per image with a total of over 600,000 questions
- 10 ground truth answers per question (we won't be using this)
- 1 given answer for each question

The size of the VQA dataset is large and hence we have provided a smaller version:



	Size	Download time (to Colab)
Full dataset	20GB	17 minutes
Small dataset (Reduced image size)	4GB	<3 minutes

The small dataset is a smaller version of the full dataset where the images have been resized to 224x224 to reduce the file sizes. The links to the various files in the datasets are given below.

VisualQA Small Dataset (4GB)

- [Training set images](#)
- [Training set questions](#)
- [Training set answers](#)

- [Validation set images](#)
- [Validation set questions](#)
- [Validation set answers](#)

VisualQA Full Dataset (20GB)

- [Training set images](#)
- [Training set questions](#)
- [Training set answers](#)

- [Validation set images](#)
- [Validation set questions](#)
- [Validation set answers](#)

Dataset Details

Images

There are **82,783** train image files and **40,504** validation image files once you extract train2014.zip and val2014.zip. Keep a note of the file name of the images as the numerical part of the file name is the image id and this is what you will use to combine with the questions and answers for each image.

Examples :

`'visualqa_2017/train2014/COCO_train2014_000000496617.jpg'`,



'visualqa_2017/train2014/COCO_train2014_000000226059.jpg',
'visualqa_2017/train2014/COCO_train2014_000000099956.jpg',
'visualqa_2017/train2014/COCO_train2014_000000072906.jpg'

In the shown example image files 496617, 226059, 99956, 72906 are the corresponding image ids.

Questions

The question can be read from the file v2_OpenEnded_mscoco_train2014_questions.json and v2_OpenEnded_mscoco_val2014_questions.json. The json file contains the questions corresponding to the images in the following format. Keep a note of the question id as that is what will be used to combine with the answers for each of the questions.

Examples :

```
{'image_id': 458752, 'question': 'What is this photo taken looking through?', 'question_id': 458752000}  
{'image_id': 458752, 'question': 'What position is this man playing?', 'question_id': 458752001}  
{'image_id': 458752, 'question': 'What color is the players shirt?', 'question_id': 458752002}
```

In the shown example the image of id 458752 has 3 question attached to it with the ids 458752000, 458752001, 458752002

Answers

The answers for each question can be read from v2_mscoco_train2014_annotations.json and v2_mscoco_val2014_annotations.json. Each question can have multiple answer types. But for the practicum you need to consider only the answers from the field **'multiple_choice_answer'** and treat that as the ground truth answer label. The annotation file has the link between the images, questions, and answers.

Examples :

```
{'answer_type': 'other',  
'answers': [{'answer': 'net', 'answer_confidence': 'maybe', 'answer_id': 1},  
{'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 2},  
{'answer': 'net', 'answer_confidence': 'yes', 'answer_id': 3},...],  
'image_id': 458752,  
'multiple_choice_answer': 'net',  
'question_id': 458752000,  
'question_type': 'what is this'}
```

```
{'answer_type': 'other',  
'answers': [{'answer': 'pitcher', 'answer_confidence': 'yes', 'answer_id': 1},  
{'answer': 'catcher', 'answer_confidence': 'no', 'answer_id': 2},  
{'answer': 'pitcher', 'answer_confidence': 'yes', 'answer_id': 3},...],  
'image_id': 458752,  
'multiple_choice_answer': 'pitcher',  
'question_id': 458752001,  
'question_type': 'what'}
```



```
{'answer_type': 'other',  
'answers': [{'answer': 'orange', 'answer_confidence': 'yes', 'answer_id': 1},  
{'answer': 'orange', 'answer_confidence': 'yes', 'answer_id': 2},  
{'answer': 'orange', 'answer_confidence': 'maybe', 'answer_id': 3},...],  
'image_id': 458752,  
'multiple_choice_answer': 'orange',  
'question_id': 458752002,  
'question_type': 'what color is the'}
```

In the shown example the image of id 458752 has a question with id 458752000 and the answer to the question is 'net'. Similarly the same image with id 458752 has another question with id 458752001 and the answer to the question is 'pitcher'. This should give you an idea on how to combine images with questions and answers to feed into your model.

The Visual Question Answering dataset is large and will require significant training time even on a GPU. We recommend you use Google Colab and enable GPU or use any computing environment that has GPUs. Here are some metrics for the small dataset on the average training time per epoch based in number of classes selected:

# of Classes (Filtered by top answer counts)	Train time per epoch (GPU allocated in Colab differed for each test run)	Val Accuracy/Loss (After 10 epochs)
10	13 minutes (1 GPU)	50% / 0.9
25	15 minutes (1 GPU)	47% / 1.12
50	12 minutes (1 GPU)	47% / 1.21
100	18 minutes (1 GPU)	44% / 1.4
200	22 minutes (1 GPU)	43% / 1.61

Objective of this practicum

We expect your team to develop at a minimum the following functionality:

- Build efficient data pipelines using the **small** (4GB) dataset for both images and text
- Use all training and validation images in the dataset
- Use the multiple choice answer for each question as your output labels
- There are 658,111 questions (train + validation) and over 29,000 unique answers in total. But you can reduce your questions by filtering only the top **10** answers across the train and validation set. Your top answers will be something like 'yes', 'no', '1', '2', 'white', etc.



- Build an image model and ensure you use transfer learning for the image task
- Build a language model and ensure you use transfer learning for the text task
- Build a multimodal model using both the image and language model and come up with a combined visual question answering model
- **Achieve an accuracy higher than 45%**
- Implement 3 additional features (details given below)

Recommended workflow

We suggest the following workflow as your team works on the problem. This will help you focus on various parts of the project separately but yet can integrate all together when your team is ready to put it together.

- Spend some time to understand the structure of the dataset. There are 3 components namely image files, question file, and answer file. The question and answers are in json format and you will need to build logic to combine them all.
- Start with a subset of the data and zip it similar to the original dataset
- Use the subset to build your data downloading and data processing pipelines
- Build your image model using this subset
- Build your text model using this subset
- Combine the models to make sure they work together
- Finally switch to the original dataset
- If you are using Colab store the dataset in a GCS Bucket so various team members can access and work on the same data
- Save your training metrics after each run of the model

Additional features

Choose 3 additional features from the following list.

Feature 1: Build the model using the full (20GB) dataset

For this feature you will need to use the original VisualQA Full Dataset of size 20GB. The links to the dataset were given in the Dataset section. The main challenge for this feature will be handling the large dataset size. Ensure your accuracy is still maintained and above the required baseline given in the practicum objectives.

Feature 2: Build any technique from the leaderboard of VQA

The leaderboard for the VisualQA dataset can be seen here: <https://visualqa.org/roe.html>. Some of the top performers have their architectures published as code or as papers. Implement any of their approaches (or a model inspired by their techniques) and get either a score of **85%** or higher for yes/no answers or **55%** or higher for numerical answers.



Feature 3: Build a second technique from the leaderboard of VQA

Implement another technique from the leaderboard for the VisualQA dataset. Follow their approach or a model inspired by their techniques and get either a score of **85%** or higher for yes/no answers or **55%** or higher for numerical answers.

Feature 4: Apply distillation to compress the final model

Distillation is a process where knowledge is transferred from a pre-trained teacher model to a student model by minimizing a loss function. This process is a model compression technique, where a smaller (student) model is trained from a larger pre-trained (teacher) model. Implement knowledge distillation to create a student model. Compare the original model size, accuracy, and inference times to the student model's size, accuracy, and inference times.

Feature 4: Apply quantization to compress the final model

Post-training quantization is a conversion technique that can reduce model size while also improving CPU and hardware accelerator latency, with little degradation in model accuracy. Implement model quantization and compare the original model size, accuracy, and inference times to the quantized model's size, accuracy, and inference times.

Feature 5: Apply pruning to compress the final model

Pruning is a model compression technique that optimizes a model by removing weights that do not contribute much to the results. Implement model pruning and compare the original model size, accuracy, and inference times to the quantized model's size, accuracy, and inference times.

Feature 6: Convert dataset to TFRecords

The [TFRecord format](#) is a simple format for storing a sequence of binary records called protocol buffers. Protocol buffers are Google's language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler. For this feature you can pre-process your data and store them as TF Records. Then during training you can directly read from the TFRecords and feed them to the model saving time on preprocessing.

Feature 7: Deploy model to TF Serving

[TensorFlow Serving](#) is a flexible, high-performance serving system for machine learning models, designed for production environments. For this feature we recommend you follow the instructions from the section [Install Tensorflow Serving using Docker](#). Once you have it up and running you will need to use [Postman](#) or a python client to test out your API's.

Feature 8: Build a frontend app

Build a frontend app that uses your model to predict on images and questions. For this you can use Docker containers to build out a frontend app and backend API's. The app can be a single web page which displays some test images and some questions for a user to pick from and a text box to type in a new question. On submission, use the image + question to feed it to the model and display the prediction results as the answer



What to submit

Practicum memo

The practicum memo should be a short write-up with screenshots of how you tackled the problem and how your solution meets the requirements. We expect 2 to 3 pages.

Video recording

The video recording should be less than 2 minutes long and should show your team's work. Show all the different features you have implemented.

Your code in the repo with a readme

A readme that lists out all the notebook files and any other code files that you have implemented and a very brief description of what each of them does.

How to submit

[Submission procedure for presentations, practicums and projects](#)

Grading system

- Minimum functionality : **10 points**
- 3 added features : **6 points**
- Practicum memo, video recording, code : **4 points**

Total: 20 points