MiniVQA - A resource to build your tailored VQA competition

Jean-Benoit Delbrouck
Stanford University
jeanbenoit.delbrouck@stanford.edu

Abstract

MiniVQA[1] is a Jupyter notebook to build a tailored VQA competition for your students. The resource creates all the needed resources to create a classroom competition that engages and inspires your students on the free, self-service Kaggle platform. “InClass competitions make machine learning fun!”[2].

1 Task overview

Beyond simply recognizing what objects are present, vision-and-language tasks, such as captioning (Chen et al., 2015) or visual question answering (Antol et al., 2015), challenge systems to understand a wide range of detailed semantics of an image, including objects, attributes, spatial relationships, actions and intentions, and how all these concepts are referred to and grounded in natural language. VQA is therefore viewed as a suitable way to evaluate a system reading comprehension.

“Since questions can be devised to query any aspect of text comprehension, the ability to answer questions is the strongest possible demonstration of understanding“ (Lehnert, 1977)

2 To teach NLP

A trained model cannot perform satisfactorily on the Visual Question Answering task without language understanding abilities. A visual-only system (i.e., processing only the image) will not have the visual reasoning skills required to provide correct answers (as it doesn’t process the questions as input). Two NLP challenges arise to build a multimodal model:

- The questions must be processed by the system as input. This is the opportunity to teach different features extractions techniques for sentences. The techniques can range from unsupervised methods (bag of words, tf-idf or Word2Vec/Doc2Vec (Mikolov et al., 2013) models) to supervised methods (Recurrent Neural Networks (Hochreiter and Schmidhuber, 1997) or Transformers (Vaswani et al., 2017) neural networks).
- The extracted linguistic features must be incorporated into the visual processing pipeline. This challenge lies at the intersection of vision and language research. Different approaches, such as early and late fusion techniques, can be introduced.

All implemented techniques can be evaluated on the difficult task that is VQA. Because answering questions about a visual context is a hard cognitive task, using different NLP approaches can lead to significant performance differences.

3 Resource overview

The source code is split into several sections, each section containing tunable parameters to modulate the dataset to match your needs. For example, you can choose the number of possible answers, the sample size per answer or the balance of labels between splits.

MiniVQA built upon two datasets, the VQA V2 dataset (Goyal et al., 2017) and the VQA-Med dataset (Ben Abacha et al., 2020). You can choose to create a competition based on natural images or medical images. MiniVQA proposes 443k questions on 82.7k images and 4547 unique answers for the former, and 7.4k unique image-question pairs and 332 answers for the latter [3].

Finally, MiniVQA provides a second Jupyter notebook that trains and evaluates a baseline VQA

[1] https://github.com/jbdel/miniVQA
[2] https://www.kaggle.com/
[3] As of 2021

Proceedings of the Fifth Workshop on Teaching NLP, pages 55–58
June 10–11, 2021. ©2021 Association for Computational Linguistics

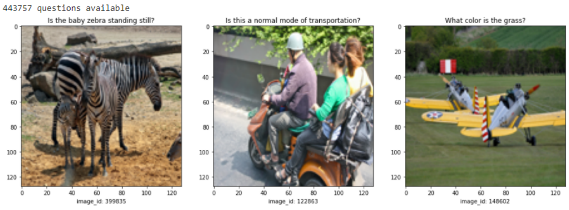
model on any dataset you create. You are free to share it or not amongst your class.

4 Resource presentation

The following sections present the features of MiniVQA. We illustrate these features using the VQA v2.0 dataset as a matter of example. The same can be applied to the VQA-Med dataset.

4.1 Automatic download

The resource automatically downloads annotations (questions, image_ids and answer)s from the official datasets websites. Any pre-processing that must be done is carried out. The number of possible questions and unique answers are printed to the user, along with random examples.



```
443757 questions available
```

4.2 Decide the volume of your dataset

Using MiniVQA, you can choose the size of your dataset according to several settings.

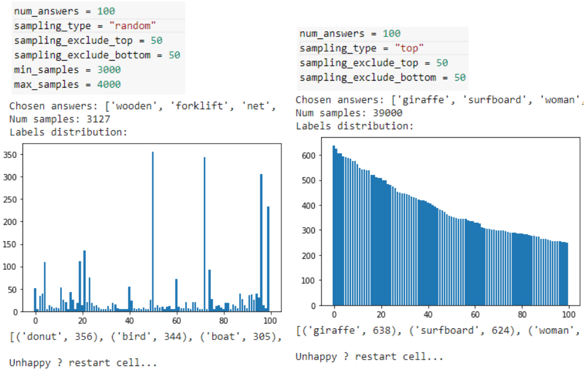
num_answers the number of possible different answers (i.e., how many classes).

sampling_type how to select samples, choose between "top" or "random". Value "top" gets the 'num_answers' most common answers in the dataset.

sampling_exclude_top and **sampling_exclude_bottom** you can choose to exclude the n most popular or least popular answers (the most popular answers is "no" and contains 80.000 examples).

min_samples and **max_samples** if sampling_type is random, you can choose a minimum and maximum number of samples with min_samples and max_samples.

This section outputs a bar graph containing the label distributions and some additional information. Figure 1 shows two examples.



```
num_answers = 100
sampling_type = "random"
sampling_exclude_top = 50
sampling_exclude_bottom = 50
min_samples = 3000
max_samples = 4000
```

```
Chosen answers: ['wooden', 'forklift', 'net',
Num samples: 3127
Labels distribution:
```

```
[('donut', 356), ('bird', 344), ('boat', 305),

Unhappy ? restart cell...
```

```
num_answers = 100
sampling_type = "top"
sampling_exclude_top = 50
sampling_exclude_bottom = 50
```

```
Chosen answers: ['giraffe', 'surfboard', 'woman',
Num samples: 39000
Labels distribution:
```

```
[('giraffe', 638), ('surfboard', 624), ('woman',

Unhappy ? restart cell...
```

Figure 1: sampling_type random (left) and sampling_type top (right).

4.3 Create dataset files

This section creates the chosen samples a json format. Two more parameters are available.

sample_clipping set to select n maximum samples per answer. This setting is particularly handy if you chose the "top" sampling_type in the previous section.

im_download you can choose to download the images of the selected samples directly through http requests. Though rather slow, this allows the user not to download the images of the full dataset.

resize if im_download is set to True, images are squared-resized to n pixels. For your mini-VQA project, you might want to use lower resolution for your images (faster training). $n = 128$ is a good choice.

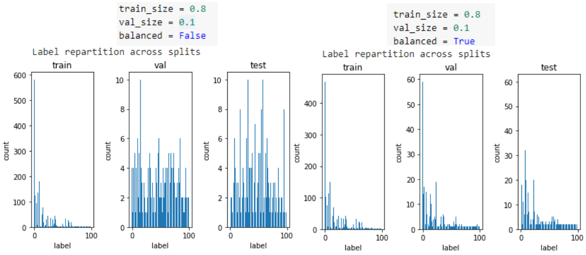
4.4 Create splits

As in any competition, participants are provided a train and validation set with ground-truth answers, and a test-set without these answers.

train_size and **valid_size** fraction of the total examples selected to populate the splits. 0.8 and 0.1 are usually good values. The rest (0.1) goes in the test set.

balanced whether or not labels are homogeneously distributed across splits.

Figure 2 shows an example of the balanced setting effect:



```
train_size = 0.8
val_size = 0.1
balanced = False
```

```
train_size = 0.8
val_size = 0.1
balanced = True
```

Figure 2: balance set to True (left) and to False (right).

Regardless of the balanced value, at least one sample of each label is put in each split.

4.5 Explore the question embedding space

It is possible to compute questions embedding using pre-trained transformer models. Each representation is then reduced into a two-dimensional point using t-SNE algorithm (van der Maaten and Hinton, 2008). These embeddings can also be used for section 5. MiniVQA plots two projections, one for the 5 most popular question types and one with randomly chosen question types.

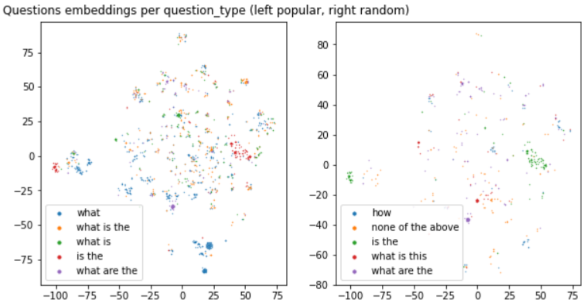


Figure 3: Questions embedding using a pretrained bert-base-nli-mean-tokens model.

4.6 Download files

Finally, you can download the dataset file in json, the splits, and optionally, the images.

{train, val, sample_submission}.csv csv files containing question_id, label.

test.csv must be given to students. They must fill it with their systems predictions formatted like sample_submission.csv which contains random predictions.

answer_key.csv is the ground-truth file that has to be stored on Kaggle (see Appendix A).

answer_list.csv maps the label to the answer in natural language (i.e., label 0 is answer at line 1 in answer_list, etc.).

image_question.json maps an image_id to a list of questions (that concerns image_id). Each question is a tuple (question_id, question).

5 Baseline Model

The provided baseline consists of a dataloader that opens images and resize them to 112 × 112 pixels and that embeds questions to a feature vector of size 768 from a pre-trained DistilBERT (Sanh et al.).

The network consists a modified Resnet (He et al., 2016) that takes as input an RBG image of size 112 × 112 and outputs a feature map of size 512 × 4 × 4 that is flattened and then concatenated with the question representation of size 768. Finally, a classification layer projects this representation to probabilities over answers. The network is trained until no improvements is recorded on the validation set (early-stopping).

References

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)*.

Asma Ben Abacha, Vivek V. Datla, Sadid A. Hasan, Dina Demner-Fushman, and Henning Müller. 2020. Overview of the vqa-med task at imageclef 2020: Visual question answering and generation in the medical domain. In *CLEF 2020 Working Notes*, CEUR Workshop Proceedings, Thessaloniki, Greece. CEUR-WS.org <http://ceur-ws.org>.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. 2015. Microsoft coco captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Wendy Lehnert. 1977. Human and computational question answering. *Cognitive Science*, 1(1):47–73.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc.

Victor Sanh, Lysandre DEBUT, Julien CHAUMOND, Thomas WOLF, and Hugging Face. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

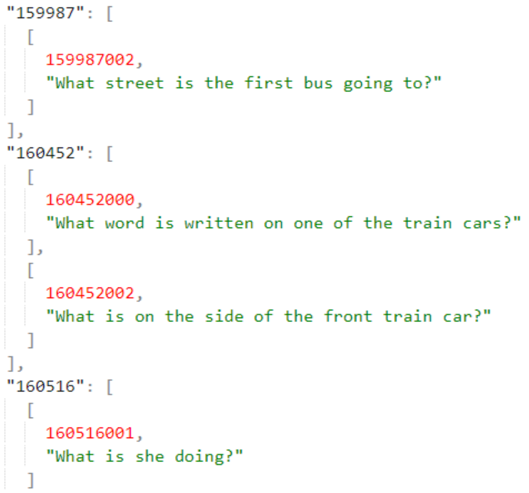
Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

A Create a competition on Kaggle

Navigate to `http://www.kaggle.com/inclass`. Follow the instructions to setup an InClass competition. Upload files train.csv, val.csv, test.csv, answer_key.csv, and sample_submission.csv when prompted.

B image_question.json structure



```
"159987": [
  [
    159987002,
    "What street is the first bus going to?"
  ]
],
"160452": [
  [
    160452000,
    "What word is written on one of the train cars?"
  ],
  [
    160452002,
    "What is on the side of the front train car?"
  ]
],
"160516": [
  [
    160516001,
    "What is she doing?"
  ]
```

Figure 4: Maps an image_id to a list of questions (that concerns image_id). Each question is a tuple (question_id, question).