Bonus_Task_Generation

January 12, 2022

1 Generating Synthesis Visual Language Reasoning Data Set

1.1 Introduction

This note book uses a package named ShapeWorld to generate a visual language reasoning reasoning data set. The package ShapeWorld can be accessed from GitHub at the address here. The package generates abstract images consisted by shapes (square, circle, triangle, etc.) in different colors. A (language) caption about the image is also generated. The package can generate images in batches. We first generate the images and then a create manually curated text paragraph, question and answer choices based on each created images. The combined image, paragraph, question and answer choices could be like following. ## Sample Example Image:

Paragraph: If someone add one more circle to the image, there will be two circles in total in the image.

Question: Determine whether the passage is a valid argument for the given image.

Answer: A. True; B. False
Correct Answer: True

2 Generation Methodology

The manually created formats are used to generate text paragraph, question, answer choices. The formats can be categorized into five main types based on the methods to generate the texts from the image. 1. **Generation by adding one more shape** * With this generation method, we add one more quantity of particular shape to the image and generate two types of paragraph. One is correctly describe the the image marked with true after after the addition and the other is falsely describe the image marked with false. We created the paragraph by manually rephrasing the wording in many different ways. The question is to ask whether the paragraph is correct. 2. **Generation by adding one more color** * This generation method follows the same way as generation by adding one more shape but in a way to add one more color to the image. 3. **Generation by replacing one shape with another shape** * This generation method generates text by assuming that one shape is replaced by another shape in some quantities in the image. The text generation then follows different rephased formats that either give a true or false description of the image after replacing the shape. 4. **Generation by replacing one color with another color** * This generation method generates text by assuming that one color is replaced by another color in some quantities in the image. The text generation then follows different rephased formats that either give a true or false description of the image after replacing the shape. 5. **Generation using relational caption** * In this method, we use the caption ground truth generated by the

ShapeWorld package as a comparing target to generate kinds of true or false statement through various formats.

2.1 Note

In order to simplify the dataset generation process due to time limitation, all the answers in this generated data set has a uniform two-option format of ["True", "False"] instead of four-options case.

Before we can play with the ShapeWorld package, we first need clone the repository from GitHub.

[152]:
```
!git clone --recursive https://github.com/AlexKuhnle/ShapeWorld.git
```

```
fatal: destination path 'ShapeWorld' already exists and is not an empty
directory.
```

The package Pillow is used to convert array formats into images

[153]:
```
!pip install pillow
```

```
Requirement already satisfied: pillow in /usr/local/lib/python3.6/dist-packages
(7.0.0)
```

After cloning the ShapeWorld, we need to add the cloned repository to system path.

[154]:
```
import sys
from google.colab import drive

# This will prompt for authorization to mount Google Drive to this Colab␣
 ↪notebook.
drive.mount('/content/drive')

# Add the ShapeWorld repository to system path
if not 'ShapeWorld' in sys.path:
  sys.path += ['ShapeWorld']
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

We are now starting generating the data set.

[155]:
```
from shapeworld import Dataset
# number of worlds to generate
world_size = 100

# calling create (generate) function
dataset = Dataset.create(dtype='agreement', name='existential',␣
 ↪world_size=world_size)

# generated variable is a collection of the worlds (with properties) in json␣
 ↪format
```

```
generated = dataset.generate(n=100, mode='train', include_model=True)

###################### Some description and details about the world geenrated␣
↪#####################
print('world shape:', dataset.world_shape())
print('caption shape:', dataset.vector_shape(value_name='caption'))
print('vocabulary size:', dataset.vocabulary_size(value_type='language'))
print('vocabulary:', dataset.vocabulary)

# caption surface forms
print('first few captions:')
print('\n'.join(dataset.to_surface(value_type='language',␣
↪word_ids=generated['caption'][:5])))

# given to the image caption agreement model
batch = (generated['world'], generated['caption'], generated['caption_length'],␣
↪generated['agreement'])

# can be used for more specific evaluation
world_model = generated['world_model']
```

```
world shape: (100, 100, 3)
caption shape: (8,)
vocabulary size: 23
vocabulary: <bound method Dataset.vocabulary of
<shapeworld.datasets.agreement.existential.ExistentialDataset object at
0x7f4ed0ab96a0>>
first few captions:
there is a blue cross .
there is a square .
a shape is a cross .
a shape is a magenta circle .
there is a gray cross .
```

[0]:
```
data_set = []
```

[157]:
```
# Generate paragraph, questions and answers using Existential module in␣
↪ShapeWorld #
################################## Adding one shape to generate question␣
↪############################

import random
from PIL import Image
import numpy as np

print("========================Generating questions by adding one␣
↪shape=========================")
```

```
################### image_number_range ########################

image_number_range = [0, 30]

######################## Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):
  all_shapes = []
  all_colors = []
  number_of_shapes = {}
  number_of_colors = {}

  # Collecting all shapes and colors in a world image
  for entity in generated["world_model"][i]["entities"]:
    all_shapes.append(entity["shape"]["name"])
    all_colors.append(entity["color"]["name"])

  # Collecting the counts of each shape in a world image
  for s in all_shapes:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["shape"]["name"] == s:
        count += 1
    number_of_shapes[s] = count
  # Collecting the counts of each color in a world image
  for c in all_colors:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["color"]["name"] == c:
        count += 1
    number_of_colors[c] = count

  ########## generating the question ##########
  question_id = "1_shapeworld_addshape_"+"exist_"+str(i)
  shape = random.choice(all_shapes)
  generation_operation = ["add", "remove"]
  participle = {"add": "added", "remove": "removed"}
  prep = {"add": "to", "remove": "from"}
  key = {"add": "True", "remove": "False"}
  choices = ["True", "False"]
  operation = random.choice(generation_operation)
```

```
 paragraph = random.choice(["If we {} one more {} {} the given image, this␣
↪action will make the resuting image to have total {} counts of shapes being␣
↪a {} .".format(operation, shape, prep[operation], number_of_shapes[shape]+1,␣
↪shape),
                            "If someone {} one more {} {} the given image, this␣
↪will make the image to with total {} shapes being a {} .".format(operation,␣
↪shape, prep[operation], number_of_shapes[shape]+1, shape),
                            "If there will be one more {} {} {} the given␣
↪image, this change will make the resuting image to have total {} counts of␣
↪entities being a {} shape.".format(shape, participle[operation],␣
↪prep[operation],number_of_shapes[shape]+1, shape)])
 question = random.choice(["Determine whether the passage is a valid argument␣
↪for the given image.",
                            "Based on the image, check whether the statement in␣
↪the paragraph is true or not.",
                            "Tell if the statement about the image in the␣
↪paragraph is true or not."])
 correct_answer = key[operation]

 if i<=image_number_range[0]+first_n_samples:
   print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
 data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

 # Save the image using question_id as the image name
 im_array = np.array(generated["world"][i])
 im_array = im_array*255
 im_array = im_array.astype(np.uint8)
 im = Image.fromarray(im_array)
 im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
==========Generating questions by adding one
shape==========
Printing out first 5 generated samples!
{'id': '1_shapeworld_addshape_exist_0', 'paragraph': 'If there will be one more
pentagon added to the given image, this change will make the resuting image to
have total 2 counts of entities being a pentagon shape.', 'question': 'Tell if
the statement about the image in the paragraph is true or not.', 'choices':
['True', 'False'], 'correct_answer': 'True'}
{'id': '1_shapeworld_addshape_exist_1', 'paragraph': 'If someone remove one more
square from the given image, this will make the image to with total 2 shapes
being a square .', 'question': 'Tell if the statement about the image in the
paragraph is true or not.', 'choices': ['True', 'False'], 'correct_answer':
'False'}
{'id': '1_shapeworld_addshape_exist_2', 'paragraph': 'If there will be one more
```

```
cross added to the given image, this change will make the resuting image to have
total 2 counts of entities being a cross shape.', 'question': 'Determine whether
the passage is a valid argument for the given image.', 'choices': ['True',
'False'], 'correct_answer': 'True'}
{'id': '1_shapeworld_addshape_exist_3', 'paragraph': 'If we remove one more
triangle from the given image, this action will make the resuting image to have
total 2 counts of shapes being a triangle .', 'question': 'Based on the image,
check whether the statement in the paragraph is true or not.', 'choices':
['True', 'False'], 'correct_answer': 'False'}
{'id': '1_shapeworld_addshape_exist_4', 'paragraph': 'If we add one more circle
to the given image, this action will make the resuting image to have total 3
counts of shapes being a circle .', 'question': 'Determine whether the passage
is a valid argument for the given image.', 'choices': ['True', 'False'],
'correct_answer': 'True'}
{'id': '1_shapeworld_addshape_exist_5', 'paragraph': 'If we add one more circle
to the given image, this action will make the resuting image to have total 2
counts of shapes being a circle .', 'question': 'Determine whether the passage
is a valid argument for the given image.', 'choices': ['True', 'False'],
'correct_answer': 'True'}
```

[158]:
```python
################################### Adding one color to generate question␣
↪#################################
print("=========================Generating questions by adding one␣
↪color==========================")

################### image_number_range ######################

image_number_range = [31, 60]

###################### Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):
  all_shapes = []
  all_colors = []
  number_of_shapes = {}
  number_of_colors = {}

  # Collecting all shapes and colors in a world image
  for entity in generated["world_model"][i]["entities"]:
    all_shapes.append(entity["shape"]["name"])
    all_colors.append(entity["color"]["name"])
```

```
 # Collecting the counts of each shape in a world image
 for s in all_shapes:
   count = 0
   for entity in generated["world_model"][i]["entities"]:
     if entity["shape"]["name"] == s:
       count += 1
   number_of_shapes[s] = count
 # Collecting the counts of each color in a world image
 for c in all_colors:
   count = 0
   for entity in generated["world_model"][i]["entities"]:
     if entity["color"]["name"] == c:
       count += 1
   number_of_colors[c] = count

 # Changing one shape to generate question
 question_id = "2_shapeworld_addcolor_"+"exist_"+str(i)
 shape = random.choice(all_shapes)
 color = random.choice(all_colors)
 generation_operation = ["add", "remove"]
 participle = {"add": "added", "remove": "removed"}
 prep = {"add": "to", "remove": "from"}
 key = {"add": "True", "remove": "False"}
 choices = ["True", "False"]
 operation = random.choice(generation_operation)
 paragraph = random.choice(["If we {} one more {} {} {} the given image, this␣
↪action will make the resuting image to have total {} counts of shapes in {}␣
↪color .".format(operation, color, shape, prep[operation],␣
↪number_of_colors[color]+1, color),
                            "If someone {} one more {} {} the given image, this␣
↪will make the image to with total {} shapes being {} color.".
↪format(operation, color, shape, prep[operation], number_of_shapes[shape]+1,␣
↪color),
                            "If there will be one more {} {} {} {} the given␣
↪image, this change will make the resuting image to have total {} counts of␣
↪{} shape entities.".format(color, shape, participle[operation],␣
↪prep[operation],number_of_colors[color]+1, color)])
 question = random.choice(["Determine whether the passage is a valid argument␣
↪for the given image.",
                            "Based on the image, check whether the statement in␣
↪the paragraph is true or not.",
                            "Tell if the statement about the image in the␣
↪paragraph is true or not."])
 correct_answer = key[operation]
 if i<=image_number_range[0]+first_n_samples:
```

```
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
  data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

  # Save the image using question_id as the image name
  im_array = np.array(generated["world"][i])
  im_array = im_array*255
  im_array = im_array.astype(np.uint8)
  im = Image.fromarray(im_array)
  im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
=========================Generating questions by adding one
color=========================
Printing out first 5 generated samples!
{'id': '2_shapeworld_addcolor_exist_31', 'paragraph': 'If there will be one more
blue triangle removed from the given image, this change will make the resuting
image to have total 3 counts of blue shape entities.', 'question': 'Tell if the
statement about the image in the paragraph is true or not.', 'choices': ['True',
'False'], 'correct_answer': 'False'}
{'id': '2_shapeworld_addcolor_exist_32', 'paragraph': 'If someone remove one
more cyan circle the given image, this will make the image to with total from
shapes being 4 color.', 'question': 'Based on the image, check whether the
statement in the paragraph is true or not.', 'choices': ['True', 'False'],
'correct_answer': 'False'}
{'id': '2_shapeworld_addcolor_exist_33', 'paragraph': 'If there will be one more
red square removed from the given image, this change will make the resuting
image to have total 2 counts of red shape entities.', 'question': 'Determine
whether the passage is a valid argument for the given image.', 'choices':
['True', 'False'], 'correct_answer': 'False'}
{'id': '2_shapeworld_addcolor_exist_34', 'paragraph': 'If we add one more blue
circle to the given image, this action will make the resuting image to have
total 4 counts of shapes in blue color .', 'question': 'Based on the image,
check whether the statement in the paragraph is true or not.', 'choices':
['True', 'False'], 'correct_answer': 'True'}
{'id': '2_shapeworld_addcolor_exist_35', 'paragraph': 'If we add one more gray
cross to the given image, this action will make the resuting image to have total
3 counts of shapes in gray color .', 'question': 'Tell if the statement about
the image in the paragraph is true or not.', 'choices': ['True', 'False'],
'correct_answer': 'True'}
{'id': '2_shapeworld_addcolor_exist_36', 'paragraph': 'If someone add one more
gray semicircle the given image, this will make the image to with total to
shapes being 2 color.', 'question': 'Tell if the statement about the image in
the paragraph is true or not.', 'choices': ['True', 'False'], 'correct_answer':
'True'}
```

[159]:
```
################################## Replacing shape with another shape to␣
 ↪generate questions ##################################
print("========================Generating questions by replacing shape with␣
 ↪another shape=========================")

################### image_number_range ######################

image_number_range = [60, 90]

###################### Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):
  all_shapes = []
  all_colors = []
  number_of_shapes = {}
  number_of_colors = {}

  # Collecting all shapes and colors in a world image
  for entity in generated["world_model"][i]["entities"]:
    all_shapes.append(entity["shape"]["name"])
    all_colors.append(entity["color"]["name"])

  # Collecting the counts of each shape in a world image
  for s in all_shapes:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["shape"]["name"] == s:
        count += 1
    number_of_shapes[s] = count

  # Collecting the counts of each color in a world image
  for c in all_colors:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["color"]["name"] == c:
        count += 1
    number_of_colors[c] = count

  # Replacing shape to generate question
  question_id = "3_shapeworld_replaceshape_"+"exist_"+str(i)
  choices = ["True", "False"]
  if len(all_shapes) >1:
```

```
    all_shapes_copy = all_shapes.copy()
    shape_1 = random.choice(all_shapes)

    all_shapes_copy.remove(shape_1)
    shape_2_remove = random.choice(all_shapes_copy)
  if len(all_shapes) ==1:
    all_shapes_copy = all_shapes.copy()
    option_shapes = ["square", "circle", "triangle"]
    if all_shapes[0] in option_shapes:
      option_shapes.remove(all_shapes[0])
    candidate_shape = random.choice(option_shapes)
    all_shapes_copy.append(candidate_shape)
    number_of_shapes[candidate_shape] = 0
    shape_1 = candidate_shape
    shape_2_remove = all_shapes[0]

  color = random.choice(all_colors)
  random_number = random.choice([0, random.randint(1,3)])
  paragraph = random.choice(["Before we start counting the number of shapes in␣
↪the image, we first replace {} counts of the {} shapes with {} counts of the␣
↪{} shapes.".format(number_of_shapes[shape_2_remove], shape_2_remove,␣
↪number_of_shapes[shape_2_remove]+random_number, shape_1),
                             "Someone has made changes to the original image␣
↪which removed all the shpes of {} in the image and added {} counts of the {}␣
↪shapes.".format(shape_2_remove,␣
↪number_of_shapes[shape_2_remove]+random_number, shape_1),
                            "The image is modifed by someone and has {} counts␣
↪of {} shapes removed from the image and {} counts of additional {} shapes␣
↪added.".format(number_of_shapes[shape_2_remove], shape_2_remove,␣
↪number_of_shapes[shape_2_remove]+random_number, shape_1)])
  question = random.choice(["After the action described in the paragraph has␣
↪been taken, are there now {} counts of {} shapes in the image?".
↪format(number_of_shapes[shape_2_remove]+number_of_shapes[shape_1], shape_1),
                            "If the action described in the paragraph has␣
↪happend, are there now {} counts of {} shapes existing in the image?".
↪format(number_of_shapes[shape_2_remove]+number_of_shapes[shape_1], shape_1),
                            "In the image, are there now {} counts of {} shapes␣
↪after the action in the paragraph has happended?".
↪format(number_of_shapes[shape_2_remove]+number_of_shapes[shape_1], shape_1)])
  if random_number == 0:
    correct_answer = "True"
  if random_number != 0:
    correct_answer = "False"
  if i<=image_number_range[0]+first_n_samples:
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
```

```
 data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

 # Save the image using question_id as the image name
 im_array = np.array(generated["world"][i])
 im_array = im_array*255
 im_array = im_array.astype(np.uint8)
 im = Image.fromarray(im_array)
 im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
=========================Generating questions by replacing shape with another
shape=========================
Printing out first 5 generated samples!
{'id': '3_shapeworld_replaceshape_exist_60', 'paragraph': 'The image is modifed
by someone and has 2 counts of cross shapes removed from the image and 2 counts
of additional cross shapes added.', 'question': 'In the image, are there now 4
counts of cross shapes after the action in the paragraph has happended?',
'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '3_shapeworld_replaceshape_exist_61', 'paragraph': 'Before we start
counting the number of shapes in the image, we first replace 2 counts of the
semicircle shapes with 3 counts of the ellipse shapes.', 'question': 'After the
action described in the paragraph has been taken, are there now 3 counts of
ellipse shapes in the image?', 'choices': ['True', 'False'], 'correct_answer':
'False'}
{'id': '3_shapeworld_replaceshape_exist_62', 'paragraph': 'Someone has made
changes to the original image which removed all the shpes of ellipse in the
image and added 4 counts of the rectangle shapes.', 'question': 'If the action
described in the paragraph has happend, are there now 5 counts of rectangle
shapes existing in the image?', 'choices': ['True', 'False'], 'correct_answer':
'False'}
{'id': '3_shapeworld_replaceshape_exist_63', 'paragraph': 'Before we start
counting the number of shapes in the image, we first replace 2 counts of the
square shapes with 5 counts of the semicircle shapes.', 'question': 'In the
image, are there now 3 counts of semicircle shapes after the action in the
paragraph has happended?', 'choices': ['True', 'False'], 'correct_answer':
'False'}
{'id': '3_shapeworld_replaceshape_exist_64', 'paragraph': 'Before we start
counting the number of shapes in the image, we first replace 4 counts of the
triangle shapes with 4 counts of the cross shapes.', 'question': 'In the image,
are there now 5 counts of cross shapes after the action in the paragraph has
happended?', 'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '3_shapeworld_replaceshape_exist_65', 'paragraph': 'The image is modifed
by someone and has 1 counts of semicircle shapes removed from the image and 2
counts of additional pentagon shapes added.', 'question': 'If the action
described in the paragraph has happend, are there now 2 counts of pentagon
shapes existing in the image?', 'choices': ['True', 'False'], 'correct_answer':
'False'}
```

[160]:
```
################################## Replacing color with another color to␣
 ↪generate questions ##################################
print("========================Generating questions by replacing color with␣
 ↪another color=========================")

################### image_number_range ######################

image_number_range = [60, 90]

###################### Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):
  all_shapes = []
  all_colors = []
  number_of_shapes = {}
  number_of_colors = {}

  # Collecting all shapes and colors in a world image
  for entity in generated["world_model"][i]["entities"]:
    all_shapes.append(entity["shape"]["name"])
    all_colors.append(entity["color"]["name"])

  # Collecting the counts of each shape in a world image
  for s in all_shapes:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["shape"]["name"] == s:
        count += 1
    number_of_shapes[s] = count

  # Collecting the counts of each color in a world image
  for c in all_colors:
    count = 0
    for entity in generated["world_model"][i]["entities"]:
      if entity["color"]["name"] == c:
        count += 1
    number_of_colors[c] = count

  # Replacing color to generate question
  question_id = "4_shapeworld_replacecolor_"+"exist_"+str(i)
  choices = ["True", "False"]
  if len(all_colors) >1:
```

```
    all_colors_copy = all_colors.copy()
    color_1 = random.choice(all_colors)
    all_colors_copy.remove(color_1)
    color_2_remove = random.choice(all_colors_copy)

  if len(all_colors) ==1:
    all_colors_copy = all_colors.copy()
    option_colors = ['yellow', 'cyan', 'red']
    if all_colors[0] in option_colors:
      option_colors.remove(all_colors[0])
    candidate_color = random.choice(option_colors)
    all_colors_copy.append(candidate_color)
    number_of_colors[candidate_color] = 0
    color_1 = candidate_color
    color_2_remove = all_colors[0]

  color = random.choice(all_colors)
  random_number = random.choice([0, random.randint(1,3)])
  paragraph = random.choice(["There are {} shapes in {} color. We will paint␣
↪those shapes with a new color {}".format(number_of_colors[color_2_remove],␣
↪color_2_remove, color_1),
                             "A person decided to repaint the image. He painted␣
↪the shapes in {} color with another color {}.".format(color_2_remove,␣
↪color_1),
                            "The image is modifed by a painter in a way that {}␣
↪counts of shapes in {} are repainted with {} color.".
↪format(number_of_colors[color_2_remove], color_2_remove, color_1)])
  question = random.choice(["After the painting action described in the␣
↪paragraph, there are now {} shapes in {}. Is this true?".
↪format(number_of_colors[color_2_remove]+number_of_colors[color_1]+random_number,␣
↪color_1),
                            "If we count the number of shapes in {} color after␣
↪the action described in the paragraph has taken place, would we see {}␣
↪counts of shapes in {} color?".format(color_1,␣
↪number_of_colors[color_2_remove]+number_of_colors[color_1]+random_number,␣
↪color_1),
                            "Based on the information given in the paragraph,␣
↪can we expect to see {} counts of shapes in {} color in the image?".
↪format(number_of_colors[color_2_remove]+number_of_colors[color_1]+random_number,␣
↪color_1)])
  if random_number == 0:
    correct_answer = "True"
  if random_number != 0:
    correct_answer = "False"
  if i<=image_number_range[0]+first_n_samples:
```

```
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
  data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

  # Save the image using question_id as the image name
  im_array = np.array(generated["world"][i])
  im_array = im_array*255
  im_array = im_array.astype(np.uint8)
  im = Image.fromarray(im_array)
  im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
==========Generating questions by replacing color with another
color==========
Printing out first 5 generated samples!
{'id': '4_shapeworld_replacecolor_exist_60', 'paragraph': 'The image is modifed
by a painter in a way that 1 counts of shapes in blue are repainted with gray
color.', 'question': 'After the painting action described in the paragraph,
there are now 2 shapes in gray. Is this true?', 'choices': ['True', 'False'],
'correct_answer': 'True'}
{'id': '4_shapeworld_replacecolor_exist_61', 'paragraph': 'There are 1 shapes in
cyan color. We will paint those shapes with a new color green', 'question':
'Based on the information given in the paragraph, can we expect to see 6 counts
of shapes in green color in the image?', 'choices': ['True', 'False'],
'correct_answer': 'False'}
{'id': '4_shapeworld_replacecolor_exist_62', 'paragraph': 'A person decided to
repaint the image. He painted the shapes in red color with another color
yellow.', 'question': 'Based on the information given in the paragraph, can we
expect to see 7 counts of shapes in yellow color in the image?', 'choices':
['True', 'False'], 'correct_answer': 'False'}
{'id': '4_shapeworld_replacecolor_exist_63', 'paragraph': 'The image is modifed
by a painter in a way that 3 counts of shapes in yellow are repainted with
magenta color.', 'question': 'Based on the information given in the paragraph,
can we expect to see 6 counts of shapes in magenta color in the image?',
'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '4_shapeworld_replacecolor_exist_64', 'paragraph': 'The image is modifed
by a painter in a way that 2 counts of shapes in gray are repainted with magenta
color.', 'question': 'After the painting action described in the paragraph,
there are now 6 shapes in magenta. Is this true?', 'choices': ['True', 'False'],
'correct_answer': 'False'}
{'id': '4_shapeworld_replacecolor_exist_65', 'paragraph': 'A person decided to
repaint the image. He painted the shapes in gray color with another color
cyan.', 'question': 'Based on the information given in the paragraph, can we
expect to see 5 counts of shapes in cyan color in the image?', 'choices':
['True', 'False'], 'correct_answer': 'False'}
```

[161]:
```
######################## Using Relational caption type for world generation␣
↪########################
# number of worlds to generate
world_size = 100

# calling create (generate) function
dataset = Dataset.create(dtype='agreement', name='relational',␣
↪world_size=world_size)

# generated variable is a collection of the worlds (with properties) in json␣
↪format
generated = dataset.generate(n=100, mode='train', include_model=True)

######################## Some description and details about the world geenrated␣
↪########################
print('world shape:', dataset.world_shape())
print('caption shape:', dataset.vector_shape(value_name='caption'))
print('vocabulary size:', dataset.vocabulary_size(value_type='language'))
print('vocabulary:', dataset.vocabulary)

# caption surface forms
print('first few captions:')
print('\n'.join(dataset.to_surface(value_type='language',␣
↪word_ids=generated['caption'][:5])))

# given to the image caption agreement model
batch = (generated['world'], generated['caption'], generated['caption_length'],␣
↪generated['agreement'])

# can be used for more specific evaluation
world_model = generated['world_model']
```

```
world shape: (100, 100, 3)
caption shape: (15,)
vocabulary size: 49
vocabulary: <bound method Dataset.vocabulary of
<shapeworld.datasets.agreement.relational.RelationalDataset object at
0x7f4ed0e597b8>>
first few captions:
a triangle is not lighter than a cyan square .
a cyan shape is to the left of a square .
a magenta semicircle is not closer to the red circle than a red semicircle .
a green cross is not in front of a cyan cross .
a yellow shape is not the same shape as a cyan shape .
```

[162]:
```
# Generate question with type_1 format
```

```
print("========================Generating questions by using relational␣
 ↪reasoning (type_1 format)========================")

################### image_number_range ######################

image_number_range = [0, 30]

###################### Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):

  question_id = "5_1_shapeworld_reasoning_"+"rel1_"+str(i)
  choices = ["True", "False"]
  statement = dataset.to_surface(value_type='language',␣
 ↪word_ids=generated["caption"][i])
  if generated['agreement'][i] == 1:
    ground_truth = "True"
  if generated['agreement'][i] == 0:
    ground_truth = "False"

  judgement = random.choice(["True", "False"])
  paragraph = random.choice(["Mike is taking a reasoning test. In order to get␣
 ↪one point, he needs to make a correct classification about wether a␣
 ↪statement about the image is true or false. He think the statement '{}'␣
 ↪about the image is {}".format(statement, judgement),
                             "Mike think the statement '{}' about the image is␣
 ↪a {} one. If he is right, he will be arwarded one point.".format(statement,␣
 ↪judgement),
                             "There is statement about the image that '{}'.␣
 ↪Someone think the statement is {}. If his jedgement is correct, he will get␣
 ↪one point as award.".format(statement, judgement)])
  question = random.choice(["Is his judgement correct?", "Will he be awarded␣
 ↪one point regarding his judgement?", "Based on the real image, can one say␣
 ↪he is correct in judgement?"])
  if judgement == ground_truth:
    correct_answer = "True"
  if judgement != ground_truth:
    correct_answer = "False"
  if i<=image_number_range[0]+first_n_samples:
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
 ↪"choices": choices, "correct_answer": correct_answer})
```

```
  data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

  # Save the image using question_id as the image name
  im_array = np.array(generated["world"][i])
  im_array = im_array*255
  im_array = im_array.astype(np.uint8)
  im = Image.fromarray(im_array)
  im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
=========================Generating questions by using relational reasoning
(type_1 format)=========================
Printing out first 5 generated samples!
{'id': '5_1_shapeworld_reasoning_rel1_0', 'paragraph': "There is statement about
the image that 'a triangle is not lighter than a cyan square .     '. Someone
think the statement is True. If his jedgement is correct, he will get one point
as award.", 'question': 'Will he be awarded one point regarding his judgement?',
'choices': ['True', 'False'], 'correct_answer': 'False'}
{'id': '5_1_shapeworld_reasoning_rel1_1', 'paragraph': "Mike is taking a
reasoning test. In order to get one point, he needs to make a correct
classification about wether a statement about the image is true or false. He
think the statement 'a cyan shape is to the left of a square .    ' about the
image is True", 'question': 'Will he be awarded one point regarding his
judgement?', 'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '5_1_shapeworld_reasoning_rel1_2', 'paragraph': "Mike think the statement
'a magenta semicircle is not closer to the red circle than a red semicircle .'
about the image is a False one. If he is right, he will be arwarded one point.",
'question': 'Will he be awarded one point regarding his judgement?', 'choices':
['True', 'False'], 'correct_answer': 'True'}
{'id': '5_1_shapeworld_reasoning_rel1_3', 'paragraph': "There is statement about
the image that 'a green cross is not in front of a cyan cross .   '. Someone
think the statement is True. If his jedgement is correct, he will get one point
as award.", 'question': 'Based on the real image, can one say he is correct in
judgement?', 'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '5_1_shapeworld_reasoning_rel1_4', 'paragraph': "Mike is taking a
reasoning test. In order to get one point, he needs to make a correct
classification about wether a statement about the image is true or false. He
think the statement 'a yellow shape is not the same shape as a cyan shape .  '
about the image is False", 'question': 'Is his judgement correct?', 'choices':
['True', 'False'], 'correct_answer': 'False'}
{'id': '5_1_shapeworld_reasoning_rel1_5', 'paragraph': "Mike is taking a
reasoning test. In order to get one point, he needs to make a correct
classification about wether a statement about the image is true or false. He
think the statement 'a cross is not a different color from a square .    ' about
the image is False", 'question': 'Based on the real image, can one say he is
correct in judgement?', 'choices': ['True', 'False'], 'correct_answer': 'False'}
```

[163]:
```
# Generate question with type_2 format
print("=========================Generating questions by using relational␣
↪reasoning (type_2 format)=========================")

################### image_number_range ######################

image_number_range = [31, 60]

####################### Generating ############################

first_n_samples = 5
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):

  question_id = "5_2_shapeworld_reasoning_"+"rel2_"+str(i)
  choices = ["True", "False"]
  statement = dataset.to_surface(value_type='language',␣
↪word_ids=generated["caption"][i])
  if generated['agreement'][i] == 1:
    ground_truth = "True"
  if generated['agreement'][i] == 0:
    ground_truth = "False"

  judgement = random.choice(["True", "False"])
  paragraph = random.choice(["Regarding the image, there is statement '{}' made␣
↪by John. ".format(statement),
                             "David made a state {} regarding the image.".
↪format(statement),
                             "A statement is made by someone regarding the␣
↪image. It describes some relations in the image that '{}'".
↪format(statement)])
  question = random.choice(["Is he making a correct statement about the image?
↪", "Is his statement correct?", "Is the statement made by his a true␣
↪statement?"])
  correct_answer = ground_truth
  if i<=image_number_range[0]+first_n_samples:
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
  data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

  # Save the image using question_id as the image name
  im_array = np.array(generated["world"][i])
```

```
  im_array = im_array*255
  im_array = im_array.astype(np.uint8)
  im = Image.fromarray(im_array)
  im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
 ↪image_collection/"+question_id+".png")
```

```
=========================Generating questions by using relational reasoning
(type_2 format)=========================
Printing out first 5 generated samples!
{'id': '5_2_shapeworld_reasoning_rel2_31', 'paragraph': "A statement is made by
someone regarding the image. It describes some relations in the image that 'a
red triangle is behind a blue pentagon .      '", 'question': 'Is his statement
correct?', 'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '5_2_shapeworld_reasoning_rel2_32', 'paragraph': "A statement is made by
someone regarding the image. It describes some relations in the image that 'a
green ellipse is to the right of a cyan shape .   '", 'question': 'Is his
statement correct?', 'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '5_2_shapeworld_reasoning_rel2_33', 'paragraph': 'David made a state a
magenta triangle is closer to the red rectangle than a yellow shape .  regarding
the image.', 'question': 'Is the statement made by his a true statement?',
'choices': ['True', 'False'], 'correct_answer': 'True'}
{'id': '5_2_shapeworld_reasoning_rel2_34', 'paragraph': "A statement is made by
someone regarding the image. It describes some relations in the image that 'a
circle is above a yellow shape .       '", 'question': 'Is he making a correct
statement about the image?', 'choices': ['True', 'False'], 'correct_answer':
'True'}
{'id': '5_2_shapeworld_reasoning_rel2_35', 'paragraph': "A statement is made by
someone regarding the image. It describes some relations in the image that 'a
red shape is not the same shape as a yellow shape .  '", 'question': 'Is he
making a correct statement about the image?', 'choices': ['True', 'False'],
'correct_answer': 'False'}
{'id': '5_2_shapeworld_reasoning_rel2_36', 'paragraph': 'David made a state a
magenta shape is not the same shape as a cyan shape .   regarding the image.',
'question': 'Is the statement made by his a true statement?', 'choices':
['True', 'False'], 'correct_answer': 'True'}
```

[164]:
```
# Generate question with type_3 format
print("=========================Generating questions by using relational␣
 ↪reasoning (type_3 format)=========================")

#################### image_number_range ########################

image_number_range = [61, 90]

######################## Generating ############################

first_n_samples = 5
```

```
print("Printing out first {} generated samples!".format(first_n_samples))

#################################################################

for i in range(image_number_range[0], image_number_range[1]):

  question_id = "5_3_shapeworld_reasoning_"+"rel3_"+str(i)
  choices = ["True", "False"]
  statement = dataset.to_surface(value_type='language',␣
↪word_ids=generated["caption"][i])
  if generated['agreement'][i] == 1:
    ground_truth = "True"
  if generated['agreement'][i] == 0:
    ground_truth = "False"

  paragraph = statement
  question = random.choice(["The paragraph has a description about the image,␣
↪is this description true based on the image?",
                            "A statement, describing the information in the␣
↪image, is given in the paragraph. Is it true?",
                            "The paragraph describes one relationship regarding␣
↪the shapes in the image. Judge whether the description is a true one?"])
  correct_answer = ground_truth
  if i<=image_number_range[0]+first_n_samples:
    print({"id": question_id, "paragraph":paragraph, "question": question,␣
↪"choices": choices, "correct_answer": correct_answer})
  data_set.append({"id": question_id, "paragraph":paragraph, "question":␣
↪question, "choices": choices, "correct_answer": correct_answer})

  # Save the image using question_id as the image name
  im_array = np.array(generated["world"][i])
  im_array = im_array*255
  im_array = im_array.astype(np.uint8)
  im = Image.fromarray(im_array)
  im.save("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
↪image_collection/"+question_id+".png")
```

```
=========================Generating questions by using relational reasoning
(type_3 format)=========================
Printing out first 5 generated samples!
{'id': '5_3_shapeworld_reasoning_rel3_61', 'paragraph': 'a yellow pentagon is in
front of a blue pentagon .    ', 'question': 'A statement, describing the
information in the image, is given in the paragraph. Is it true?', 'choices':
['True', 'False'], 'correct_answer': 'True'}
{'id': '5_3_shapeworld_reasoning_rel3_62', 'paragraph': 'a yellow square is to
the left of a cyan square .    ', 'question': 'The paragraph has a description
about the image, is this description true based on the image?', 'choices':
```

```
['True', 'False'], 'correct_answer': 'False'}
{'id': '5_3_shapeworld_reasoning_rel3_63', 'paragraph': 'an ellipse is not
lighter than a cyan semicircle .     ', 'question': 'The paragraph has a
description about the image, is this description true based on the image?',
'choices': ['True', 'False'], 'correct_answer': 'False'}
{'id': '5_3_shapeworld_reasoning_rel3_64', 'paragraph': 'a yellow shape is the
same shape as a gray shape .   ', 'question': 'The paragraph describes one
relationship regarding the shapes in the image. Judge whether the description is
a true one?', 'choices': ['True', 'False'], 'correct_answer': 'False'}
{'id': '5_3_shapeworld_reasoning_rel3_65', 'paragraph': 'a rectangle is not
below an ellipse .       ', 'question': 'The paragraph has a description about
the image, is this description true based on the image?', 'choices': ['True',
'False'], 'correct_answer': 'False'}
{'id': '5_3_shapeworld_reasoning_rel3_66', 'paragraph': 'a green shape is not to
the right of a cyan semicircle .  ', 'question': 'The paragraph has a
description about the image, is this description true based on the image?',
'choices': ['True', 'False'], 'correct_answer': 'False'}
```

[165]:
```
import json
import pandas as pd
# Saving data_set as json file named shapeworld_dateset on Google Drive
with open("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
 ↪shapeworld_dateset.json", "w") as fp:
    json.dump(data_set, fp)
print("Generated data_set has been saved as json file named shapeworld_dataset.
 ↪json")
# saving data_set as csv file named shapeworld_dateset on Google Drive
df = pd.DataFrame(data_set)
df.to_csv("/content/drive/My Drive/CSE576_NLP/bonus_dataset_creation/
 ↪shapeworld_dataset.csv")
print("Generated data_set has been saved as csv file named shapeworld_dateset.
 ↪csv")
```

```
Generated data_set has been saved as json file named shapeworld_dataset.json
Generated data_set has been saved as csv file named shapeworld_dateset.csv
```

[166]:
```
df
```

[166]:

	id	…	correct_answer
0	1_shapeworld_addshape_exist_0	…	True
1	1_shapeworld_addshape_exist_1	…	False
2	1_shapeworld_addshape_exist_2	…	True
3	1_shapeworld_addshape_exist_3	…	False
4	1_shapeworld_addshape_exist_4	…	True
..	…	…	…
202	5_3_shapeworld_reasoning_rel3_85	…	False
203	5_3_shapeworld_reasoning_rel3_86	…	False
204	5_3_shapeworld_reasoning_rel3_87	…	False

```
205  5_3_shapeworld_reasoning_rel3_88  …          False
206  5_3_shapeworld_reasoning_rel3_89  …          False

[207 rows x 5 columns]
```