Text2Scene Generation Generator

Abstract

1 Dataset

Table 1: Keyword occurrence

Occurrence	1	2	> 2
# keyword	114	22	23
# subject	0	2	10
# object	88	9	7
# action	26	11	6

Table 2: Layer occurrence

Occurrence	1	2	> 2
# layer	151	14	9
# background	0	0	1
# accessory	11	0	2
# character	74	9	3
# surrounding	66	5	3

2 Metric

2.1 Layer-layer similarity

We use a mean IoU to measure the similarity between two layers. A layer is typically composed of several objects and verbs. These two types will be treated differently. For object keywords, we use the original IoU. Specifically, suppose the object sets in two layers are $\mathcal{O}_1 = \{o_i\}$ and $\mathcal{O}_2 = \{o_j\}$ respectively. The similarity is

$$\mathrm{sim}_o = \frac{|\mathcal{O}_1 \cap \mathcal{O}_2|}{|\mathcal{O}_1 \cup \mathcal{O}_2|}.$$

Duplicate keywords will not treated individually when performing intersection and union. For instance, if the object set in layer 1 is $\{$"man", "man"$\}$ and the object set in layer 2 is $\{$"man"$\}$, then the intersection set will be $\{$"man"$\}$ and the union set will be $\{$"man", "man"$\}$, thus $\mathrm{sim}_o = 1/2$.

For verbs, we use a soft IoU, considering that verbs are often flexible. Suppose the verb sets in two layers are $\mathcal{A}_1 = \{a_i\}$ and $\mathcal{A}_2 = \{a_j\}$ respectively. The similarity is

$$\mathrm{sim}_a = \frac{\sum \mathcal{A}^*_\cap}{|\mathcal{A}_1| + |\mathcal{A}_2| - |\mathcal{A}^*_\cap|},$$

where

$$\mathcal{A}^*_\cap = \{r(a_i, a_j)\}_{i,j<\min(|\mathcal{A}_1|,|\mathcal{A}_2|)}.$$

Here $r(a_1, a_2)$ is the relatedness between two keywords. The intersection set is build greedily, such that the most related keywords in the two sets are first selected, then the less related keywords, until one set is exhausted. For instance, if two verb sets are {"walk", "talk"} and {"run"}, and suppose the relatedness between "walk" and "run" is 0.5, then $\mathrm{sim}_a = 1/4$. This similarity is appropriate for sets of different but similar keywords.

The overall similarity between two layers is the weighted sum of three types of keywords. Here, three types refer to verbs, objects and subjects. Subjects will fit the metric for objects. Therefore, we can write

$$\mathrm{sim}(l_1, l_2) = \alpha_s \mathrm{sim}_o(\mathcal{S}_1, \mathcal{S}_2) + \alpha_a \mathrm{sim}_a(\mathcal{A}_1, \mathcal{A}_2) + \alpha_o \mathrm{sim}_o(\mathcal{O}_1, \mathcal{O}_2),$$

where $\mathcal{S}$, $\mathcal{A}$ and $\mathcal{O}$ refer to the sets of subjects, verbs and objects in a layer. Currently, the weights are selected such that $\alpha_s = 0.5$, $\alpha_a = 0.2$, $\alpha_o = 0.3$.

2.2 Picture-picture similarity

To quantitatively evaluate the generation, we needs a metric to measure the similarity between the generated picture and the true picture. A picture is composed of several layers. And the generated picture may not have an equal number of layers against the true picture. One simple metric is the averaged pairwise similarities between layers. But this metric will discourage the number of generated layers in a picture because layers in different categories are distinct. Here we borrow the idea of soft IoU in layer-layer similarity. Specifically,

$$\mathrm{sim}(p_1, p_2) = \frac{\sum \mathcal{L}^*_\cap}{|\mathcal{L}_1| + |\mathcal{L}_2| - |\mathcal{L}^*_\cap|},$$

and

$$\mathcal{L}^*_\cap = \{\mathrm{sim}(l_i, l_j)\}_{i,j<\min(|\mathcal{L}_1|,|\mathcal{L}_2|)}.$$

The intuition here is to choose the most similar layers in two pictures as intersections.

3 Evaluation

Currently we have 90 pairs of pictures and descriptions. Let us randomly choose 70 as the training set, and the rest as test set. We do not have a learning process in our pipeline. The difference between training and test set is that the layers of pictures in training set are extracted as material. Therefore the true layers in training set are seen by the model, while the true layers in test set are unobservable. In other words the model needs to select consistent layers in the training set to compose the pictures in the test set.

Using the above metric to measure the similarity between generated pictures and true pictures, the overall average performance on the training set is 0.496. In contrast, the performance on the test set is 0.333. The prominent gap means the model needs to improve the ability to utilize existing materials.

Table 3: Generation score

	Random picture		Model	
	train	test	train	test
mIoU	0.22	0.20	0.39	0.24

The following lists some specific issues of the model reflected from the test results.

1. Entities in the description are not included in the layer base
2. Entities in the picture are not mentioned in the description. Background and most accessories will never be explicitly mentioned.

3. Token in the description and keyword in the description referring to the same object mismatch
4. Improper partition
5. Measure word not considered

Let us show some test pairs with poor performance.

1. `Frozen` - Issue 1
   (a) **Description:** `There is a snowman.`
   (b) Picture: `#alien(have[snowman])`
   (c) Generated: `#home(have[bookcase,apple,wall_clock])`
2. `prototyping_process` - Issue 2 & 1
   (a) **Description:** `A woman is standing in front of the drawing board with a pen.`
   (b) Picture:
   - `#background; #accessory(have[leaf])`
   - `#chart(have[pad,diagram]); #other(have[drawing_board])`
   - `#woman(stand,hold[pen])`

   (c) Generated: `#woman(hold[pen],point_to)`
3. `team_work` - Issue 4 & 2
   (a) **Description:** `A man is sitting in front of the leaves, and a woman is standing.`
   (b) Picture:
   - `#background`
   - `#wild(have[leaf,sun,ground])`
   - `#man(sit); #woman(stand)`

   (c) Generated: `#group(have[woman(stand),man(stand)])`
4. `Game_day` - Issue 2 & 5
   (a) **Description:** `Two man sat on the sofa celebrating a game.`
   (b) Picture:
   - `#background; #accessory`
   - `#home(have[sofa,toy,chips])`
   - `#group(have[man(drink[beer],sit,raise[arm]),man(sit,raise[arm],eat[chips`

   (c) Generated: `#man(lie_on[sofa],watch[movie])`
5. `work_chat` - Issue 5 & 4
   (a) **Description:** `Two men and a woman are standing by the message.`
   (b) Picture:
   - `#background`
   - `#other(have[circle,message])`
   - `#woman(stand); #man(stand); #man(stand,put_up)`

   (c) Generated:
   - `#chart(have[message,leaf])`
   - `#group(have[woman(stand),man(stand)])`
6. `weather` - Issue 3 & 1
   (a) **Description:** `A man is standing to give the weather report.`
   (b) Picture:
   - `#background; #accessory(have[plant]);`
   - `#other(have[bulletin])`
   - `#man(stand,point_to)`

   (c) Generated:
   - `#wild(have[rock,sign])`
   - `#man(collect[data])`

4 Runtime

Most of the time is spent on querying the similarity between keywords. Each time we query a similarity, the dictionary will be opened and closed, which consumes much time in file I/O.

now 0.021

Table 4: Runtime profile in a hierarchy. The number below each module indicates the cumulative time spent on it, on seconds.

Generator 1.133				
Grounding (layer) 0.714		Grounding (keyword) 0.418		...
Layer similarity 0.714	...			
Keyword similarity 0.711	...	Keyword similarity 0.312	...	