COO: Comic Onomatopoeia Dataset for Recognizing Arbitrary or Truncated Texts – Supplementary Material –

Jeonghun Baek, Yusuke Matsui, and Kiyoharu Aizawa

The University of Tokyo
{baek,matsui,aizawa}@hal.t.u-tokyo.ac.jp

The following materials are provided in the supplementary material:
Supplement A: We show more details of the visualization, annotation guideline, statistics, and analysis of our dataset COO.
Supplement B: We describe the details of text detection and link prediction methods in §3.
Supplement C: We provide more details of experimental settings.

A More Details of Comic Onomatopoeia Dataset

We have created the dataset COO to encourage studies on more irregular texts and to further improve text detection, recognition, and link prediction methods. Recognizing Japanese comic onomatopoeias is very difficult. If a model can recognize them, we expect that the model can also recognize other less difficult texts. A similar case exists in another task. Although Manga109, the base of COO, is a Japanese comic dataset, Manga109 [10] is widely used as a benchmark dataset in the super-resolution task [14,11]. If a model works not only in the major domain but also in minor sub-domain such as Japanese comics, we expect that the model has the potential to be generalized to various minor sub-domains.

A.1 Details of Annotation Guidelines

We define comic onomatopoeia as the texts that represent the sound or state of objects. In comics, the dialogue (line or quote) is usually in speech balloons. However, the dialogue is sometimes outside of speech balloons and written in similar fonts to onomatopoeias. For example, when the character shouts another character's name, the name is sometimes written in informal fonts. The dialogue written in informal fonts confused the annotators whether it should be regarded as an onomatopoeia. Following our guideline, they are not regarded as onomatopoeias because they do not represent the sound or state of objects.

When the annotators encountered ambiguous cases such as the dialogue written in informal fonts, annotation checker and we (authors) discussed how to handle them. To determine whether the text is onomatopoeia or not, we followed two main rules: 1) If the text is the part of the dialogue or similar to the dialogue, the text is not regarded as onomatopoeia. 2) If the text is not the part of

Table A. The intended meaning after connecting truncated texts based on each link. Each example is in Fig. 2 (b). Row and column denote the row and column of Fig. 2 (b), respectively. Text 1 and Text 2 are truncated texts.

(Row, Column)	Text 1	Text 2	Connected text	Intended meaning of connected text
(1, 1)	く	らっ	くらっ	State of feeling dizzy
(1, 2)	ガ—	—ッ	ガ——ッ	State of moving forward vigorously
(2, 1)	ガシャ	‥ン	ガシャ‥ン	Sound of breaking something
(2, 2)	と	ん	とん	Sound of putting something down

the dialogue and represents the sound or state of objects, the text is regarded as onomatopoeia. In addition, we take into account that the onomatopoeias are usually written in informal fonts.

For the link annotation, we take into account the order of reading truncated texts. For example, if the word "ばさ" is separated into "ば" and "さ", we annotate "ば" then "さ" rather than "さ" then "ば".

A.2 Data Preprocessing

In Japanese comics, there were also a few English onomatopoeias. The number of English onomatopoeias was only 148. They were excluded because they were not matched with our guideline. Most English onomatopoeias were verbs and did not represent the sound or state of objects.

All polygons in COO were validated by the function of `object.is_valid` in the python library shapely[1] [4]. Polygons that have unexpected intersections therein were corrected.

Based on the comic artist of each comic, we split 109 books in Manga109 [10] into training, validation, and test sets. In Manga109, there are multiple books written by the same comic artist. We split them into training and test sets. For example, the books "LoveHina_vol1" and "LoveHina_vol14" are written by the comic artist Akamatsu Ken. The books "ByebyeC-BOY" and "TotteokiNoABC" are written by the comic artist Aida Mayumi. Overall, Manga109 contains 30 books written by 15 comic artists (2 books per comic artist), 3 books written by one comic artist, and 4 books written by another comic artist.

A.3 Intended Meaning of Truncated Texts

To correctly capture intended meaning of truncated texts, we predict the link between truncated texts. With the predicted link, we connect truncated texts and capture the intended meaning. Table A shows the meaning of connected text after connecting truncated texts based on the link: each example is illustrated in Fig. 2 (b) of the main text.

[1] https://shapely.readthedocs.io/en/stable/manual.html

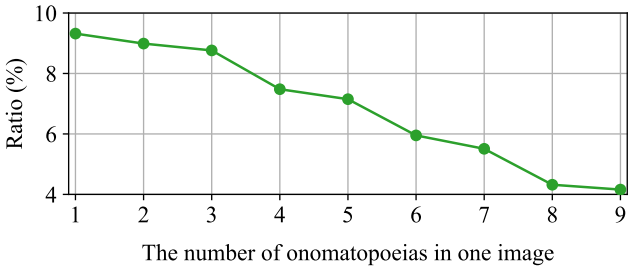

Fig. A. The ratio of the number of onomatopoeias in one image

Table B. More statistics of COO dataset: (a) Top 10 vocabularies in COO, sorted by the frequency, (b) The number of onomatopoeia according to its length of text

(a) Top 10 vocabularies in COO

Vocabulary	Count	Ratio (%)
オ	891	1.4
ザワ	749	1.2
パチ	687	1.1
ワ	677	1.1
ドキ	365	0.6
ン	347	0.6
はっ	346	0.6
?	333	0.5
ゴ	309	0.5
はあ	308	0.5

(b) Count by the length of text

Length of text	Count	Ratio (%)
1	5,845	9.5
2	26,064	42.4
3	15,909	25.9
4	6,713	10.9
5	3,384	5.5
6	2,018	3.3
7	747	1.2
8	410	0.7
9	183	0.3
10	75	0.1

Some truncated texts contain special characters such as “!”, “?”, “~”, and so on. We consider that they are also needed to capture the intended meaning, and they have a link with other truncated text.

A.4 More Statistics and Analysis

The dataset COO can be used to translate Japanese comics or analyze Japanese comics or onomatopoeias. In this subsection, we show more analysis of onomatopoeias in Japanese comics.

Fig. A shows the ratio of the number of onomatopoeias in one image. About 47.7% images have more than four onomatopoeias. About 18.3% images have only one or two onomatopoeias. About 17.8% images have no onomatopoeias, and about 20.6% images have more than nine onomatopoeias (both are not listed in the graph). These results indicate that Japanese comic images usually contain many onomatopoeias.

Table B (a) shows top 10 vocabularies sorted by the frequency. Overall, all top 10 vocabularies are short (less than 3 characters) and the sound or state

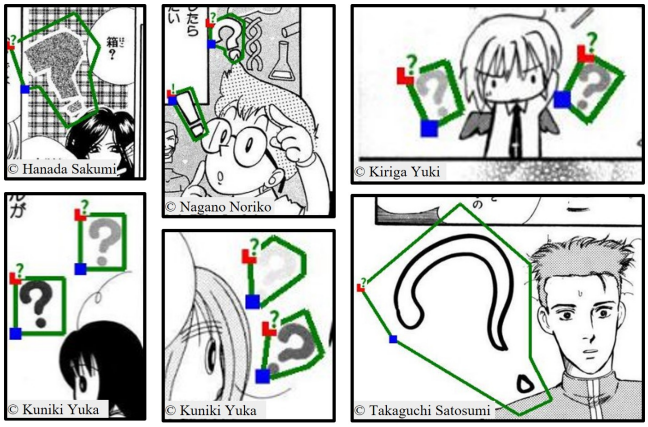

Fig. B. Question marks are also regarded as onomatopoeias because they represent the state where objects (human) wonder about something

of people are frequently used. “オ” and “ワ” represent the sound of yelling by people. “ザワ” and “ゴ” represent the state of the scene or atmosphere. “パチ” is the sound of applause by people. “ドキ” represents the state or sound of the heart beating. “はっ” represents the sound of noticing something. “はあ” represents the sound of a sigh. In COO, question mark “?” is also regarded as onomatopoeia because they represents the state in which people wonder about something. “?” is usually written in informal fonts as shown in Fig. B.

Table B (b) shows the number of onomatopoeias according to its length of text. The length of most onomatopoeias is two (42.4%) or three (25.9%) characters. It indicates that there are many short onomatopoeias in Japanese comics. COO consists of many short onomatopoeias and some long onomatopoeias. Even though many are short, it is still difficult to detect or recognize them because they are written in informal fonts, arbitrary-shaped, placed at unexpected position, or occluded, as shown in Fig. C, D, and E.

A.5 More Visualization of COO

Fig. C, D, and E show various onomatopoeias, such as arbitrary texts and truncated texts with link annotations. Some are transparent texts that look similar to background objects, some are overlapped with other onomatopoeias, and others are occluded by objects or frames.

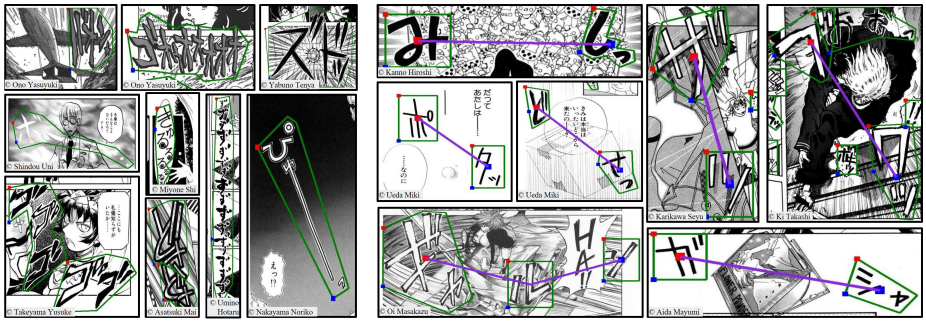

Fig. C. Each example shows diversity of onomatopoeias. Some are on the objects, some are written in informal fonts, and others lie across multiple frames in comics. Red and blue squares denote the start and end points of each annotation, respectively. Purple lines denote the link between truncated texts

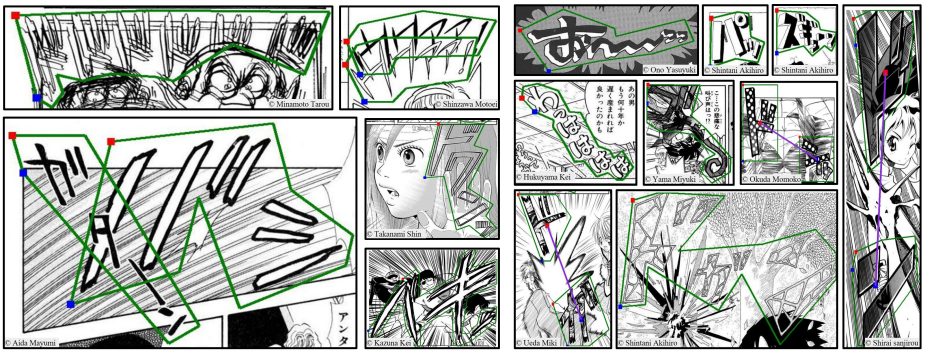

Fig. D. Each example shows diversity of onomatopoeias. Some are overlapped with other onomatopoeias and others are transparent texts that look similar to background objects

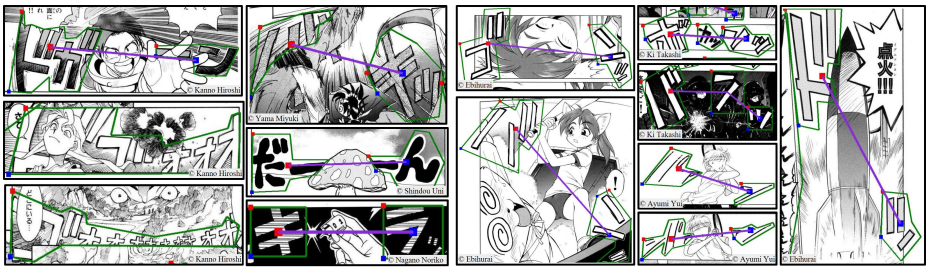

Fig. E. Each example shows diversity of onomatopoeias. Some are occluded by objects or frames, and others are separated into several parts

B More Details of Methods

B.1 Text Detection

As described in §3, we use ABCNet v2 [9] and MTS v3 [7]. We use only the text detection part of them and use only the loss function corresponding to the text detector part.

Regression-based methods for text detection usually find some points that represent each text region. ABCNet v2 finds eight control points for each text region. Eight control points represent two Bezier curves. One Bezier curve draws the top line of the text region, and another Bezier curve draws the bottom line. ABCNet v2 is trained with the regression loss for finding the coordinate of eight control points.

For MTS v3, the official code[2] [7] has the option for "train detection only," and we used this option. In this option, the model does not use the heads on the region of interest (RoI heads) and the model is trained with only segmentation loss for each text region.

B.2 Link Prediction

For M4C-COO model, we use two visual features (FRCN and bbox) and two semantic features (fastText [3] and PHOC [1])

FRCN. It is the visual (appearance) feature extracted from Faster RCNN [12]. In M4C [5] model, the feature of the fc6 layer in Faster RCNN is used. However, because we do not use RoI heads in MTS v3, we cannot use the feature of the fc6 layer for our M4C-COO model. Instead, we use the feature of the last layer, which is the segmentation map. We use the segmentation map, expecting that the segmentation map suppresses the elements that are irrelevant to texts and grasps the shape of texts. We pool the regions of proposals in the segmentation map into 32×32. After reshaping them into 1024 dimensions ($32 \times 32 = 1024$), we use them as FRCN.

bbox. It is the visual (location) feature. Each bbox is 4-dimensional relative bounding box coordinates and is calculated as follows.

$$bbox = \left[\frac{x_{min}}{W_{im}}, \frac{y_{min}}{H_{im}}, \frac{x_{max}}{W_{im}}, \frac{y_{max}}{H_{im}} \right] \tag{1}$$

where W_{im} and H_{im} denote width and height of an input image, respectively.

fastText. It is the semantic feature, a word embedding method that considers subword information. In other words, when training the word embedding model, we also use the n-gram (subword) of each word as training data. For example, if we use trigram as the subword information for the word "comics", we also use "<co", "com", "omi", "mic", "ics", and "cs>" as the training data. < and > are boundary symbols that denote at the beginning and end of the word. Because the fastText model is trained with subwords, fastText can handle out-of-vocabulary words if their subwords are used in training.

[2] https://github.com/MhLiao/MaskTextSpotterV3

PHOC. It is the semantic feature, the pyramidal histograms of characters (PHOC) for each word. Fig. F illustrates the concept of PHOC for English characters. PHOC is the concatenation of multiple binary histograms. If we embed the word "comics" as a binary histogram of characters, we will get a histogram like as L1 in Fig. F. Each dimension of the histogram represents whether the word "comics" contains a character or not. However, this embedding method has a problem: words "comics" and "cosmic" share the same histogram. To avoid this problem, the pyramid version of the histogram of characters is proposed by [1]. PHOC counts characters in the part of the word instead of the whole word. For example, at level 2, the word "comics" split into the first half of the word "com" and the second half of the word "ics". After that, we construct 2 histograms for each half word, like as L2 in Fig. F. At level 3, we split the word "comics" into three parts "co", "mi", and "cs", and then conduct the same thing. The concatenation of these binary histograms is the PHOC representation. In practice, levels 2, 3, 4, and 5 leading to a histogram of $(2 + 3 + 4 + 5) \times 36 = 504$ (36 is the sum of 26 lower-case alphabets and 10 digits) are used for English characters. In addition, the 50 most common English bigrams with level 2 leading to 100 dimensions ($2 \times 50 = 100$) is also used. As a result, PHOC is a 604-dimensional histogram for English. In COO, we use 182 characters for Japanese Hiragana, Katakana, and symbols. As a result, PHOC is a 2648-dimensional histogram: $(2 + 3 + 4 + 5) \times 182 + 2 \times 50 = 2648$.

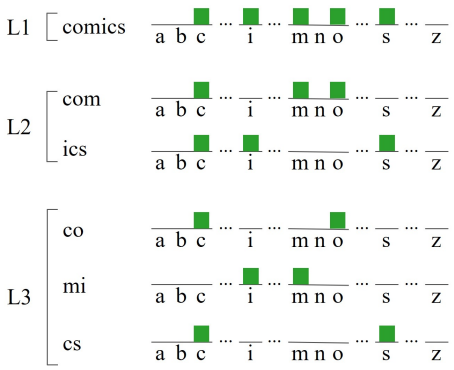

Fig. F. Pyramidal histogram of characters (PHOC). (*L1*), (*L2*), and (*L3*) are histograms of a word at levels 1, 2, and 3, respectively

C More Details of Experimental Setting

C.1 Text Detection

For ABCNet v2, two Bezier curves are created based on the points of each polygon region. Then, we generate eight points that represent the two Bezier curves. We use coordinates of these eight points as training data. When the number of points of the onomatopoeia region is four, we should have interpolated additional two points in the top line and bottom line as the authors of ABCNet v2 [9] did. As a result, the top and bottom lines have three points, respectively. When we did not interpolate, two Bezier curves were incorrectly created from four points.

To detect onomatopoeia regions, we conduct fine-tuning ABCNet v2 and MTS v3 on our dataset COO by using the pretrained models on the dataset

CTW1500 [8]. With four NVIDIA Tesla V100 GPUs, training of ABCNet v2 and MTS v3 take about 21 and 33 hours, respectively.

C.2 Text Recognition

According to Baek *et al.* [2], most text recognition methods have been trained on synthetic data because the number of real data was too small. They also shows that if we have enough real data, we can train a text recognizer only with real data. In our case, we have enough data to train a text recognizer, and thus we did not use synthetic data. Following Baek *et al.* [2], we use Adam [6] optimizer and an one-cycle learning rate schedule [13] for faster training and better performance. With one NVIDIA Tesla V100 GPU, training of TRBA with heights of the input image 32, 64, and 100 takes about 13, 24, and 35 hours, respectively.

C.3 Link Prediction

For distance-based method, we calculate the average distance from one truncated text to another truncated text. In the case of training data, the average distance is 266.2. For each feature of M4C-COO, the dimensions of FRCN, bbox, fastText, and PHOC are 1024, 4, 300, and 2648, respectively. With one NVIDIA Tesla V100 GPU, training of M4C-COO takes about 2 hours.

References

1. Almazán, J., Gordo, A., Fornés, A., Valveny, E.: Word spotting and recognition with embedded attributes. TPAMI (2014) 6, 7
2. Baek, J., Matsui, Y., Aizawa, K.: What if we only use real datasets for scene text recognition? toward scene text recognition with fewer labels. In: CVPR (2021) 8
3. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. TACL (2017) 6
4. Gillies, S., et al.: Shapely: manipulation and analysis of geometric objects (2007–), https://github.com/Toblerity/Shapely (Date last accessed 02-14-2022) 2
5. Hu, R., Singh, A., Darrell, T., Rohrbach, M.: Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In: CVPR (2020) 6
6. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015) 8
7. Liao, M., Pang, G., Huang, J., Hassner, T., Bai, X.: Mask textspotter v3: Segmentation proposal network for robust scene text spotting. In: ECCV (2020) 6
8. Liu, Y., Jin, L., Zhang, S., Luo, C., Zhang, S.: Curved scene text detection via transverse and longitudinal sequence connection. Pattern Recognition (2019) 8
9. Liu, Y., Shen, C., Jin, L., He, T., Chen, P., Liu, C., Chen, H.: Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting. TPAMI (2021) 6, 7
10. Matsui, Y., Ito, K., Aramaki, Y., Fujimoto, A., Ogawa, T., Yamasaki, T., Aizawa, K.: Sketch-based manga retrieval using manga109 dataset. MTAP (2017) 1, 2
11. Niu, B., Wen, W., Ren, W., Zhang, X., Yang, L., Wang, S., Zhang, K., Cao, X., Shen, H.: Single image super-resolution via a holistic attention network. In: ECCV (2020) 1

12. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NeurIPS (2015) 6
13. Smith, L.N., Topin, N.: Super-convergence: very fast training of neural networks using large learning rates. AI/ML for MDO (2019) 8
14. Zhang, Y., Li, K., Li, K., Wang, L., Zhong, B., Fu, Y.: Image super-resolution using very deep residual channel attention networks. In: ECCV (2018) 1