ÉCOLE D'INGÉNIEURS

ADS

First step for automatic manga translation

May 2, 2022

Student:

Simon CAIGNART simon.caignart@gmail.com

Tutor:

Mauricio Iturralde

Keywords: Manga; Computer Vision; Deep Convolutional Neural Network; Object Detection

Abstract :

With the growth of digitized comics, and especially mangas, image understanding techniques are becoming important. In fact, the translation of these mangas could be made easier thanks to artificial intelligence. In this paper, I focus on the first step needed to achieve automatic manga translation, which is the text detection step, by using a state-of-the-art object detection model named YOLOR. I compared this model with other detection methods using the Manga109 dataset and confirmed that my model outperformed them based on the AP score. I also test my model on completely new images that I annotated manually to see if it could generalize well, and discovered that some improvement could be made on large texts and input type.

Contents

1 Introduction

Comic books are very popular in the world, and exist in many different forms, such as American comics, Franco-belgian comics called Bande dessinée, and Asian comics, called manga in Japan, which all have their own styles and unique characteristics. Hundreds of those comic books are printed everyday, and nowadays a lot of them are digitized to be read on the internet, however, a lot of them are not made available to the world due to the difficulty of translation. In fact, while some of them are easy to translate because of the popularity of their language (american comics), others suffer from the difficulty of their language, like mangas, resulting in high cost of translations, and therefore fewer translations. But, what if all mangas could be immediately translated into any language? In fact, recent improvements in artificial intelligence could help us in this task, especially branches of AI like computer vision or machine translation.

Mangas are composed of four main elements: pictures, words, balloons and panels. Pictures are used to represent objects, people and figures. Words and onomatopoeia indicate character's speech and thoughts. Balloons are used to holds the words and link them to the corresponding character, finally, panels are used to structure the narrative, joining together relevant pictures, words and balloons that form a scene and also mark the continuity of time and space by the transitions between them.

Using AI, we can imagine creating several models capable of performing the necessary tasks for complete automatic translation, namely, detect and retrieve the text of each balloon, translate it, delete the original texts, and finally rewrite the translated texts in the balloons. However, this is not as easy as it looks, in fact, in order to obtain great translations, we need to have more context than just the text. The first way to solve this is by taking into account the text from the previous balloons, in order to still be able to understand phrases where the subject is omitted for example. But we need more than the text itself. For instance, knowing who is talking to whom, the gender of the characters, and other visual information can greatly improve the translations.

As we can see, achieving complete automatic manga translation is a lot of work. In this paper, I will only focus on the first step of this task, which is the text detection.

2 State of the art

In this part, I will go through and summarize the works done in relation with my objective of automating translation of mangas. That is to say, works on the detection of text balloons in comics or mangas, texts themselves, frames, characters, or even faces. Indeed, this idea of automating the translation of manga is not new, in 2011, Arai and Tolle [1] were the first to attempt speech balloon detection. They limited themselves to simple page layouts, without any overlap of different elements, i.e, all speech balloons are contained in the frames. The method they used to extract the frames was a modified version of Connected-component labeling algorithm (CCL). Next, morphological operations were applied to the extracted frames, followed by another connected-component run, from which candidates were selected using some heuristics such as minimum white pixel occurrence, width to length ratio, or width and size relative to the frame dimensions. This method performed well in terms of correct extraction, but there were still many false detections, for example, large eyes or foreheads were detected as balloons.

The next year, in 2012, Ho et al [2] took a similar approach, but identified candidate regions with their HSV (Hue, Saturation, Value) values. In fact, white and light colour areas have an high value in terms of Value (V) and low value in terms of Saturation (S). After that, like Arai et al. [1], some heuristics were applied on the shape and size. Next, they used connected-component labeling followed by a morphological dilatation. The purpose of this dilatation was to link connected components together. After this step, small connected components were considered as noise, and the biggest one, as the text block. Again, this method only works on simple page layouts, without any overlap.

In 2013, Rigaud et al. [3] introduce the first method that can detect the full shape of a speech balloon, even if its contours are subjective. They took the approach of active contours created by Ren et al. [4]. By using energy terms like relative location of text, strong edges or smooth contours. This method still had a drawback, the location of the text needed to be known.

That's why in 2017, Rigaud et al. [5] proposed a new method in which the text was used as a confidence value on whether a candidate region is a speech balloon or not. The approach still used connected-component labeling technique to detect speech balloons and text within them. After that, some heuristics were used, like alignment and centering of text within a balloon, to compute a confidence value. This method still had drawbacks has it seemed to be working only for closed balloons.

Let's now take a look at the methods proposed using Deep convolutional neural networks (DCNN). Rigaud and al started to use convolutional neural networks features in recent work, in fact, they used deep learning to detect comic characters in [6], and in [7], they proposed a CNN model which aims at extracting characters, panels, balloons and the associations between characters and balloons. During the same year, Chu and Li [8] constructed a manga character face detector based on Convolutional neural networks (CNN).

In 2018, Ogawa et al. [9] created annotations [10] for the Manga109 image dataset [11]. Manga109 is a large scale comic dataset containing 21,142 images. They assigned 527,685 annotations over the whole dataset with the help of 72 workers in six months. After that, they designed an object detector based on the SSD architecture [12], tailored for highly overlapped situations. In fact, previous methods had trouble identifying overlapped elements, like when face, body and speech balloon are overlapped. Therefore, they proposed a new CNN model called SSD300-fork, which perform well in these situations.

In 2019, Dubray et al. [13] developed a method to automatically detect and segment speech balloons in comics, using Deep convolutional neural network. Their model was trained on the Graphic Narrative Corpus [14], which is a digital corpus of graphic novels, memoirs, and non-fiction written in English. This model, which architecture is inspired by U-Net architecture [15], combined with a VGG-16 [16] based encoder, delivers impressive performance on comics. Unfortunately, the performance are not great on mangas, as the training set is mainly composed of classical comics.

In 2020, Juli´an Del Gobbo [17] released a thesis on Unconstrained Text Detection in manga. He also used the U-Net architecture for his work. His approach does not take into account the speech balloons, as he directly detect text everywhere on the page. His model, which make pixel-level text segmentation, performs better than other similar solutions. [18] [19]

Finally, in 2021, Hinami et al. [20] published a paper that share the same goal as mine, provide a method to automatically translate mangas. They used DCNN as well as Multimodal machine translation (MMT) techniques. MMT is the task of doing machine translation with multiple data sources. Their method is based on a context retrieval step, where they retrieve group texts into scenes (a scene is a manga panel), they estimate the reading order, and finally, they extract semantic information about the image, like the genders of the characters. After that, retrieved contexts are passed to their context aware translation model. Next, they clean the original Japanese texts with an image inpainting model. Lastly, they render the translated texts on the cleaned manga page.

Because I focus on the text detection step in this paper, my goal here is to beat the state of the art in this task. Hinami et al. [20] used the Faster R-CNN model [4] with ResNet101 [21] backbone. But these two technologies date from 2015. Since that, some newer ones came out, with better performances and accuracy. In fact, the state-of-the-art object detection methods can be categorized into two main types: One-stage vs. Two-stage object detectors. In the two-stage category, the state of the art is G-RCNN [22] (2021) , and in the one-stage category, the state of the art is YOLOR [23] (2021). Both these new models are great, but I will concentrate on only one of them in this paper. I will therefore use YOLOR to detect the texts of the manga pages, test its performance, and compare it to previous works.

3 The model: YOLOR

3.1 What is it?

YOLOR is a state-of-the-art machine learning algorithm for object detection, that is different from YOLOv1-YOLOv5 due to the difference in architecture, model infrastructure and authorship. YOLOR stands for “You Only Learn One Representation”, not to be confused with YOLO versions 1 through 5, where YOLO stands for “You Only Look Once”. This model is proposed as a “unified network to encode implicit knowledge and explicit knowledge together”. The findings of the YOLOR research paper [23] states that the results demonstrate benefit from using implicit knowledge. Its authors include Chien-Yao Wang, I-Hau, The, and Hong-Yuan Mark Liao. However, the previous versions of YOLO implementations have been created by Joseph Redmon and Ali Farhadi (YOLOv1 to YOLOv3), and Aleksey Bochkovskiy (YOLOv4). The authors of YOLOR represent the Taiwanese Institute of Information Science, Academia Sinica, and Elan Microelectronics Corporation of Taiwan.

3.2 How does it works?

Humans can learn and understand the physical world based on hearing, sight, and touch (explicit knowledge), but also on past experiences (implicit knowledge). Thus, humans can efficiently process brand new data by making use of abundant experience from prior learning that is gained through normal learning and stored in the brain.

Building on this idea, the YOLOR research paper describes an approach to combine explicit knowledge, defined as learning based on given data and input, with implicit knowledge. learned unconsciously. Therefore, the YOLOR concept is based on coding implicit and explicit knowledge together, similar to how the brain processes implicit and explicit knowledge combined. The unified network provided in YOLOR creates a unified representation to serve multiple tasks at the same time. The figure 1 illustrates these concepts.

There are three notable processes by which this architecture is functionally implemented: kernel space alignment, prediction refinement, and a convolutional neural networks (CNNs) with multitasking learning. As a result, when implicit knowledge is fed into a neural network that has been trained with explicit knowledge, the network benefits the performance of various tasks.

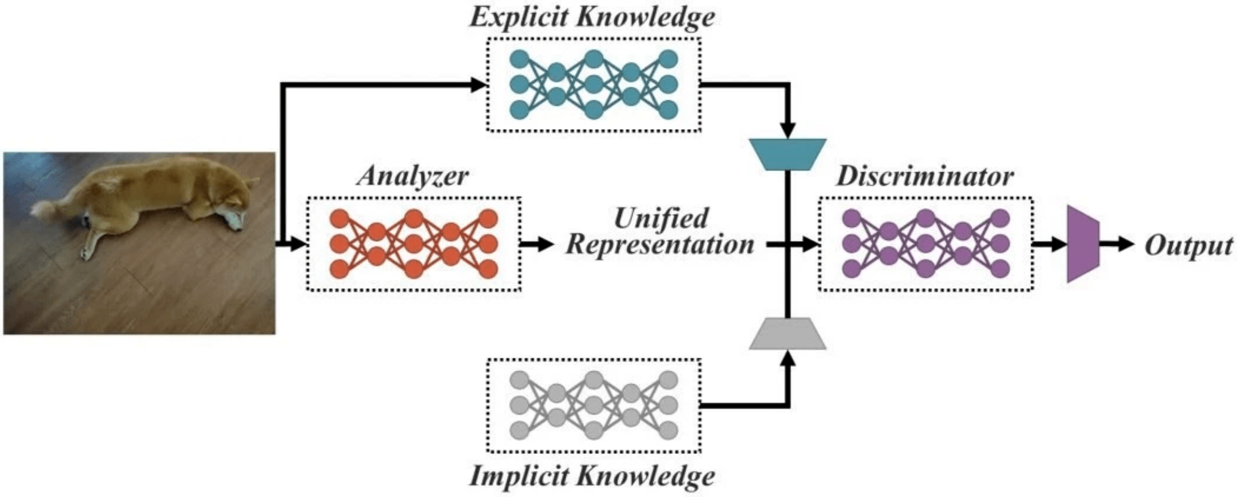


Figure 1: YOLOR concept with implicit and explicit knowledge-based multi-task learning [23]

3.3 What's new in YOLOR?

Humans can answer different questions with a single input. Given one piece of data, people can analyze data from different angles. For example, a photo of something can elicit different responses about the described action, location, etc. YOLOR aims to provide this capability to machine learning models, so that they can perform multiple tasks with a single input.

Convolutional Neural Networks (CNNs) often serve a specific purpose, while they can be trained to solve multiple problems at once, which is exactly the goal of YOLOR. While CNNs learn how to parse input to get outputs, YOLOR tries to have CNNs both (1) learn how to get outputs and also (2) what all the different outputs could be. Rather than just one output, it can have many.

3.4 YOLOR performance and precision

The new YOLOR algorithm aims to accomplish tasks using a fraction of the additional costs predicted for comparison algorithms. Thus, YOLOR is a unified network that can process explicit and implicit knowledge together and generate a finely tuned general representation through this methodology.

Combined with modern methods, YOLOR achieved object detection accuracy comparable to scaledYOLOv4, while inference speed increased by 88%. This makes YOLOR one of the fastest object detection algorithms in modern computer vision. On the MS COCO dataset, the mean accuracy of YOLOR was 3.8% higher than that of PPYOLOv2, at the same inference speed. See figure 2 for a visual representation of the performance of YOLOR vs others.

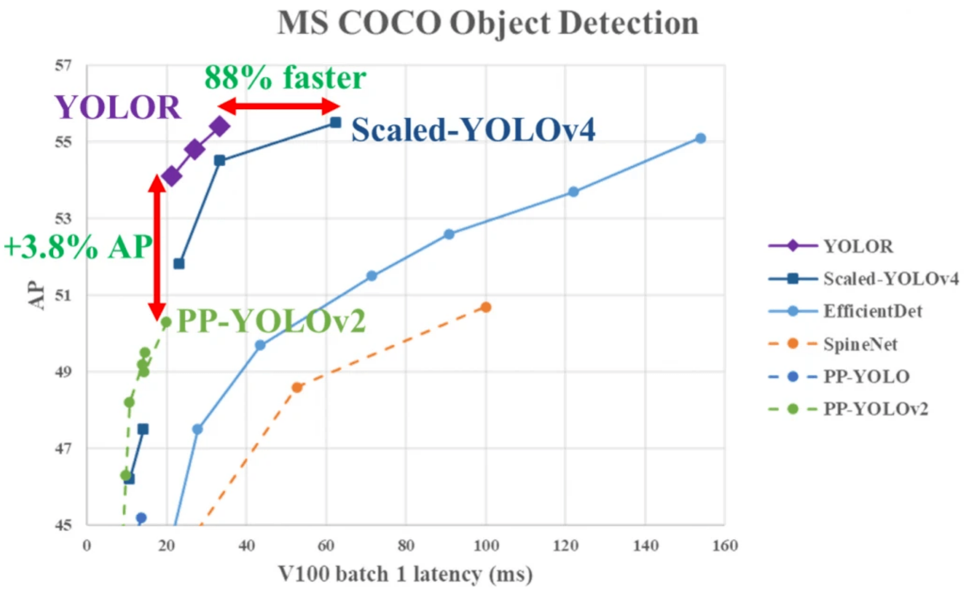
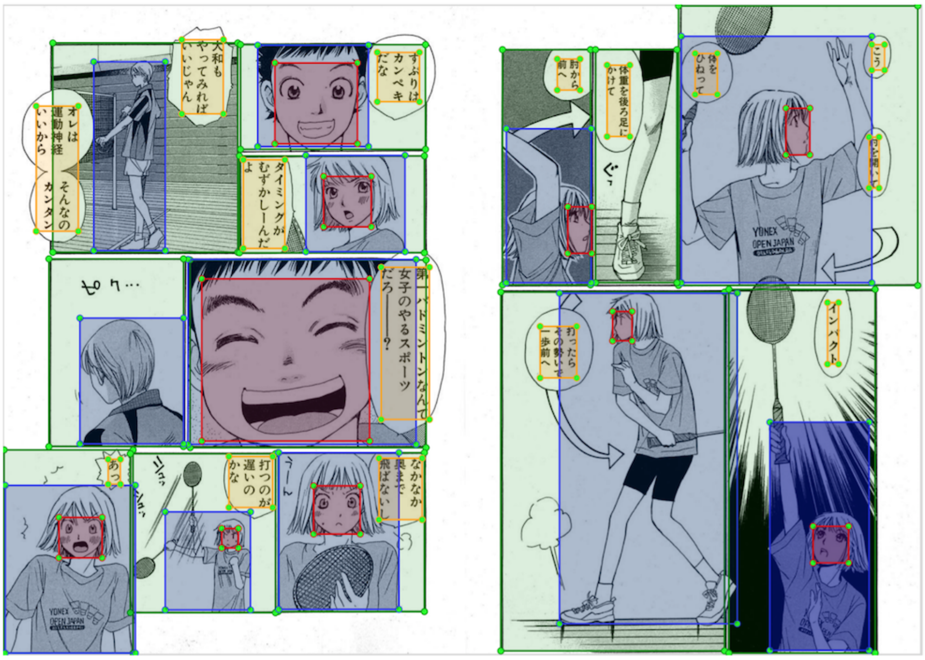


Figure 2: Performance of YOLOR vs. YOLO v4 and others [23]

4 Dataset: Manga109

To train my machine learning model, I used the Manga109 dataset [10]. This dataset has been compiled by the Aizawa Yamasaki Matsui Laboratory, Department of Information and Communication Engineering, the Graduate School of Information Science and Technology, the University of Tokyo. It is intended for use in academic research. Manga109 is composed of 109 manga volumes drawn by professional manga artists in Japan. These manga were commercially made available to the public between the 1970s and 2010s, and encompass a wide range of target readerships and genres. All the pages of the dataset were annotated with bounding boxes and other informations, see figure 3 for an annotated page example.



YamatoNoHane ©Saki Kaori

Figure 3: Example of an annotated page

Each volume of a manga corresponds to a single xml file. The xml files are UTF-8 encoded, structured in the following style:

- book : Title of the work
  - characters: List of characters appear in this work
    - character
    - character
    - ...
  - pages: List of pages in double-page spread manner
    - page
    - page
    - ...

1. A character gives the correspondence between each character's name and its ID.

2. A page gives overview (page number, size of the image) and objects information in the page. There are four types of objects. For each object, there is an object ID, a bounding box (xmin, xmax, ymin, ymax), and additional information specific to the object. The 4 types of objects are the following:
    - face : Face of a character
        - Character ID
    - body : Body of a character
        - Character ID
    - text : Typed text and some handwritten text
        - Text content
    - frame : Frame
        - No additional information

4.1 Converting the dataset

Because the dataset annotations don't follow a specific format, I had to create a script to convert it to my needs. I used python and converted it to the COCO format. COCO [24] is a format for specifying large-scale object detection, segmentation, and captioning datasets. It uses a single json file to describe all the annotations and is simple, that's why I choosed to convert the original annotations to this format.

After successfully converting the original dataset to COCO format, I converted it again to YOLO V5 PyTorch format, because YOLOR uses this format.

4.2 Preprocessing and augmentations

To increase training performances, inference time, and avoid overfitting, I applied preprocessing and augmentations to the dataset.

4.2.1 Preprocessing

The first preprocessing that I applied was to discard EXIF rotations and standardize pixel ordering. When an image is captured, it contains metadata that dictates the orientation by which it should be displayed relative to how the pixels are arranged on disk. This directive (stored in the EXIF orientation field) speeds up encoding the image at capture-time so cameras can efficiently sample data from their sensors without unwanted artifacts. This means that most cameras store images pixels exactly the same whether the camera is oriented in landscape or portrait mode. They just flip a bit to signal to the viewer whether to display the pixels as-is or to rotate them by 90 or 180 degrees when displaying the image. Unfortunately, this can cause issues if the application displaying the images is unaware of the metadata and naively displays the image without respecting its EXIF orientation. Therefore, I made sure to fix this.

The second preprocessing that I applied was to resize the images to 416x416 pixels, in order to reduce file sizes and speed up training. Because I maintained raw image aspect ratio, I filled the padding pixels with black pixels.

4.2.2 Augmentations

I then applied augmentations to the images. The first one was brightness, adding variability to image brightness helps the model to be more resilient to lighting changes. Between -10% and +10% was applied to each images.

The second is rotation, adding variability to rotations helps the model to be more resilient to scan orientation. Between -1° and +1° was applied.

4.3 Separation of data

The dataset was separated into train, validation, and test splits to prevent the model from overfitting and to accurately evaluate it. The dataset contains 10 000 images, and was split as follows:

- Training set: 7k images
- Validation set: 2k images
- Testing set: 1k images

5 Training of the model

5.1 Google Colab

The model was trained on Google Colab. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs. It was an excellent choice in my case, as I didn't have access to a GPU to train my model.

5.2 Training performances

For detection, a common way to determine if one object proposal was right is Intersection over Union (IoU, IU). This takes the set A of proposed object pixels and the set of true object pixels B and calculates:

$$IoU(A,B) = \frac{A \cap B}{A \cup B}$$

Commonly, IoU $>$ 0.5 means that it was a hit, otherwise it was a fail. After that, we can calculate the precision, recall and maP.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

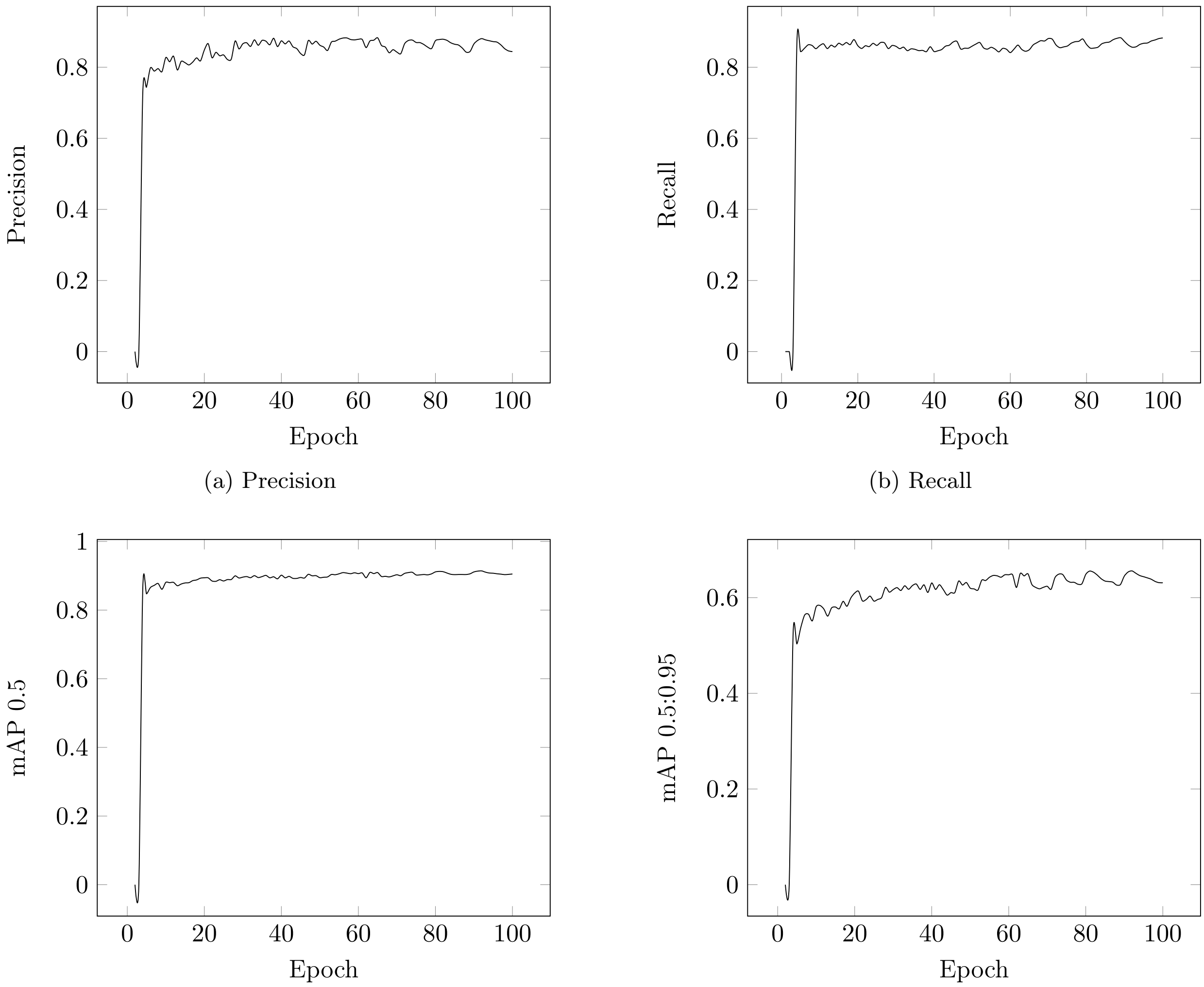
$$mAP = Mean\ of\ the\ APs$$

- True Positive (TP(c)): a proposal was made for class c and there actually was an object of class c
- False Positive (FP(c)): a proposal was made for class c, but there is no object of class c
- Average Precision (AP(c)) for class c: Area under the precision-recall curve of class c

Therefore, during training, several performance scores were calculated for each epoch in order to evaluate the effectiveness of the model:

- Precision
- Recall
- mAP 0.5 (mAP for 0.5 IoU threshold)
- mAP 0.5:0.95 (average mAP over different IoU thresholds, from 0.5 to 0.95)

See figure 4 to visualize the evolution of these scores over the training period.



(a) Precision

(b) Recall

(c) mAP 0.5

(d) mAP 0.5:0.95

Figure 4: Evolution of the training scores over 100 epochs

6 Experiments

6.1 Comparison with previous works results

After 100 epochs of training, I compared my model performances with previous CNN-based object detection methods using the Manga109-annotations dataset. Thanks to this large scale annotated dataset, I easily trained, evaluated, and compared machine learning-based methods.

Table 1 shows average precision (AP) scores for each method. I followed PASCAL VOC [25] metrics and used $IoU \geq 0.5$ as the threshold, which is standard criteria in object detection for naturalistic images.

As you can see, YOLOR outperformed other methods. The AP improved by 7.8% against SSD300-Fork.

Method	AP for text
Modified Faster-R-CNN [26]	62.0
YOLOv2 [27]	64.6
SSD300 [12]	82.0
SSD300-Fork [9]	84.1
YOLOR	**91.9**

Table 1: The comparison with previous works results for bounding box text detection using Manga109-annotations.

The figure 5 is an inference output of the model, you can see that it correctly detected the texts of the double page.

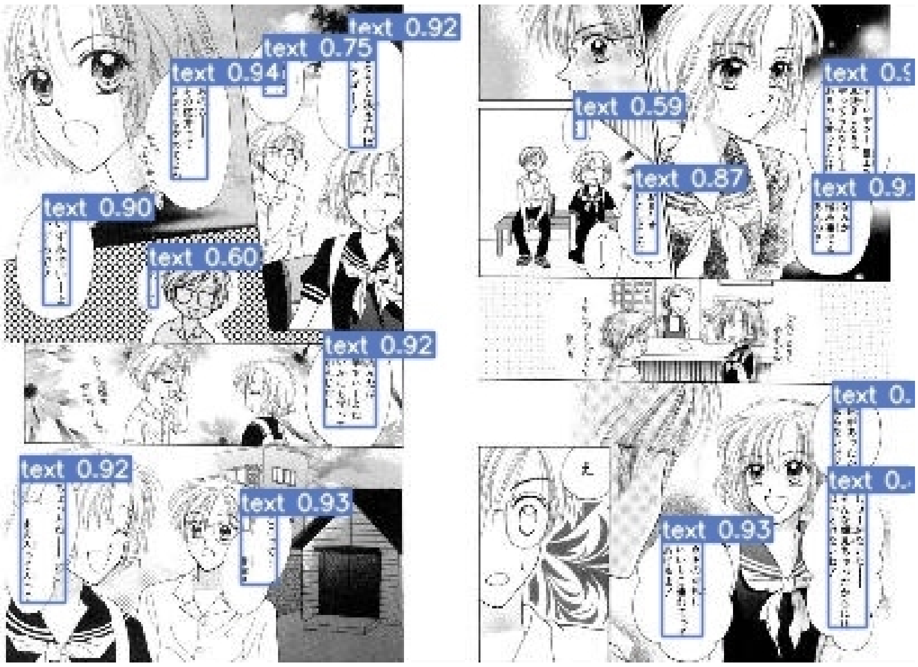


Figure 5: Great detection of the texts on a Manga109 sample.

6.2 Test on new images

I also tested my model with images totally new to it. To do this, I took 100 pages of mangas that were not in the Manga109 dataset, like One Piece, Naruto, Attack on titan, etc. I then annotated all these pages manually. The purpose of this test was to see if the performance of my model remained the same on completely new drawing styles, and therefore if it could generalize well. Table 2 shows the AP of my model on this new dataset.

Method	AP for text
YOLOR	**87.0**

Table 2: Score obtained by YOLOR on my custom dataset.

This score is slightly lower than on the Manga109 dataset, this can be explained by two factors. First of all, the images I annotated are not double pages but single page, moreover, my annotations are not pixel accurate. But in the end, this score is still higher than the score of SSD300-Fork on Manga109, and is pretty good. You can check out figure 6 for a good detection sample.

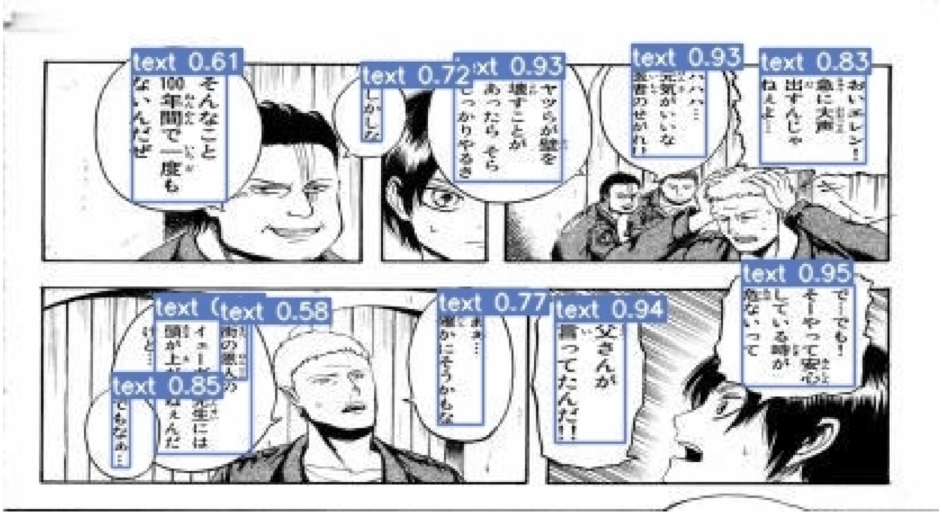

text 0.83
おいエレン！ 急に大声 出すんじゃ ねぇよ…
text 0.93
ハハハ… 元気がいいな 医者のせがれ！
xt 0.93
ヤツらが壁を 壊すことが あったら そら しっかりやるさ
text 0.72
text 0.61
そんなこと 100年間で一度も ないんだぜ
text 0.95
でーでも！ そーやって安心 している時が 危ないって
text 0.94
父さんが 言ってたんだ！！
text 0.77
まぁ… 確かにそうかもな
text 0.58
街の連中の
text (
イェーガー先生には 頭が上がらねぇんだ けど…
text 0.85
でもなぁ…

Figure 6: Great detection of the texts in the image

After analysis, I noticed that my model struggles with large texts. As you can see in figure 7, despite a correct detection, its confidence remains low, with 65%. In figure 8, the large text is not detected at all.

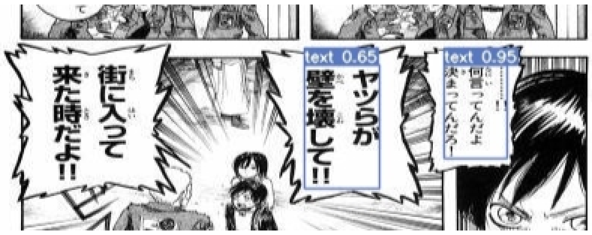


Figure 7: Large texts not detected or with poor confidence

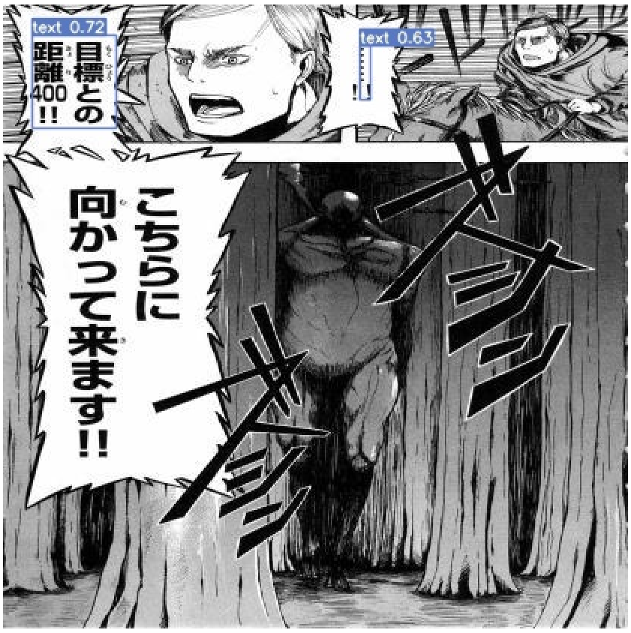


Figure 8: Large text not detected at all

7 Conclusion

In this paper, I proposed to use YOLOR for the text detection step of automatic manga translation. This model's concept is based on coding implicit and explicit knowledge together, similar to how the brain processes implicit and explicit knowledge.

I compared this model to previous CNN-based works like SSD300-fork using Manga109-annotations, and confirmed YOLOR achieved the best performance. It outperformed SD300-fork, by 7.8 % based on the AP score for text detection.

I also tested my model on new images that I manually annotated to see if it generalized well and found that the model could be improved for better text detection when the input image is a single page, as well as also enhance the detection of large texts.

8 Future work

Many different adaptations, tests, and experiments have been left for the future due to lack of time. Future work concerns modifying the dataset to improve the detection of large texts, as well as including single page images to the dataset so we can give to the model only a single page of manga, not a double page. I would also like to train the model for more epochs to improve its performances.

References

[1] H. Tolle and K. Arai, "Method for real time text extraction of digital manga comic," *International Journal of Image Processing*, 02 2011.

[2] A. K. N. Ho, J.-C. Burie, and J.-M. Ogier, "Panel and speech balloon extraction from comic books," in *2012 10th IAPR International Workshop on Document Analysis Systems*, 2012, pp. 424–428.

[3] C. Rigaud, J.-C. Burie, J.-M. Ogier, D. Karatzas, and J. Van De Weijer, "An active contour model for speech balloon detection in comics," in *2013 12th International Conference on Document Analysis and Recognition*, 2013, pp. 1240–1244.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, pp. 91–99, 2015.

[5] C. Rigaud, J.-C. Burie, and J.-M. Ogier, "Text-independent speech balloon segmentation for comics and manga," 01 2017, pp. 133–147.

[6] N.-V. Nguyen, C. Rigaud, and J.-C. Burie, "Comic characters detection using deep learning," in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 03, 2017, pp. 41–46.

[7] ——, *Multi-task Model for Comic Book Image Analysis: 25th International Conference, MMM 2019, Thessaloniki, Greece, January 8–11, 2019, Proceedings, Part II*, 01 2019, pp. 637–649.

[8] W.-T. Chu and W.-W. Li, "Manga facenet: Face detection in manga based on deep neural network," 06 2017, pp. 412–415.

[9] T. Ogawa, A. Otsubo, R. Narita, Y. Matsui, T. Yamasaki, and K. Aizawa, "Object detection for comics using manga109 annotations," 03 2018.

[10] K. Aizawa, A. Fujimoto, A. Otsubo, T. Ogawa, Y. Matsui, K. Tsubota, and H. Ikuta, "Building a manga dataset "manga109" with annotations for multimedia applications," *IEEE MultiMedia*, vol. 27, no. 2, pp. 8–18, 2020.

[11] Y. Matsui, K. Ito, Y. Aramaki, T. Yamasaki, and K. Aizawa, "Sketch-based manga retrieval using manga109 dataset," *Multimedia Tools and Applications*, vol. 76, 10 2017.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg, "Ssd: Single shot multibox detector," vol. 9905, 10 2016, pp. 21–37.

[13] D. Dubray and J. Laubrock, "Deep CNN-based Speech Balloon Detection and Segmentation for Comic Books," *arXiv e-prints*, p. arXiv:1902.08137, Feb 2019.

[14] A. Dunst, R. Hartel, and J. Laubrock, "The graphic narrative corpus (gnc): Design, annotation, and analysis for the digital humanities," 11 2017, pp. 15–20.

[15] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," vol. 9351, 10 2015, pp. 234–241.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv 1409.1556*, 09 2014.

[17] J. Gobbo and R. Matuk Herrera, "Unconstrained text detection in manga: a new dataset and baseline," 09 2020.

[18] U.-R. Ko and H.-G. Cho, *SickZil-Machine: A Deep Learning Based Script Text Isolation System for Comics Translation*, 08 2020, pp. 413–425.

[19] yu45020, “Text segmentation and image inpainting,” https://github.com/yu45020/Text_Segmentation_Image_Inpainting, 2019.

[20] R. Hinami, S. Ishiwatari, K. Yasuda, and Y. Matsui, “Towards fully automated manga translation,” 12 2020.

[21] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *CoRR*, vol. abs/1512.03385, 2015. [Online]. Available: http://arxiv.org/abs/1512.03385

[22] A. Pramanik, S. Pal, J. Maiti, and P. Mitra, “Granulated rcnn and multi-class deep sort for multi-object detection and tracking,” *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. PP, pp. 1–11, 01 2021.

[23] C.-Y. Wang, I.-H. Yeh, and H.-y. Liao, “You only learn one representation: Unified network for multiple tasks,” 05 2021.

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. Zitnick, “Microsoft coco: Common objects in context,” 05 2014.

[25] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. M. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, pp. 98–136, 2014.

[26] Y. Aramaki, Y. Matsui, T. Yamasaki, and K. Aizawa, “Text detection in manga by combining connected-component-based and region-based classifications,” 09 2016, pp. 2901–2905.

[27] J. Redmon and A. Farhadi, “Yolo9000: Better, faster, stronger,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525.

Glossary

Convolutional neural network In deep learning, a convolutional neural network is a class of artificial neural network, most commonly applied to analyze visual imagery.. 2

Deep convolutional neural network A deep convolutional neural network consists of many neural network layers.. 2

Multimodal machine translation Multimodal machine translation is the task of doing machine translation with multiple data sources - for example, translating "a bird is flying over water" + an image of a bird over water to German text.. 3