Supplementary Material for LINEEX: Data Extraction from Scientific Line Charts

Shivasankaran V P*, Muhammad Yusuf Hassan*, Mayank Singh
IIT Gandhinagar, Gujarat, India
vp.shivasan@iitgn.ac.in, md.hassan@iitgn.ac.in, singh.mayank@iitgn.ac.in

Abstract

This supplementary material provides additional details for the LINEEX *system proposed to extract data from scientific line charts. Specifically, here we provide details and results obtained with the background detection algorithm mentioned in the paper. We provide training specifications for the chart element detection and the keypoint extraction modules. Finally, we give details on the dataset used for the legend mapping module.*

1. Background Detection Results

We employ background detection as a postprocessing step to reduce erroneously detected keypoints. This helps the model filter out keypoints which are likely not on or close to a line, thus improving the precision score. The background detection algorithm can be found in the paper.

The results for the keypoint detection algorithm without background detection are enlisted in table 1. Our model suffers from low precision scores, especially in the Adobe dataset, without the background detection step. The main reason being our keypoint detection model has been trained to predict fixed M keypoints irrespective of the number of lines in the chart. Comparing the results presented in the paper, we can see that with the background detection step, the recall decreases by a small fraction while the precision increases. In the case of the Adobe dataset, there is a significant increase in precision and, thereby, in the F1 score.

2. Discussion on Metrics

We do not compare our results with the metric defined in Figureseer [7] due to the nature of their metric. The defined metric in Figureseer compares the predicted points on a predicted line with the ground truth points. This idea does not reflect the underlying information structure of line charts. For instance, let's consider the following case. A simple straight line AB with 2 ground truth points, one at A and another at B. Let N(> 2) predicted keypoints be uniformly spaced along AB. As per Figureseer's metric [7], only those predicted points which are close enough to their defined threshold are considered true positives, and the rest of the predicted points which lie between A and B on line AB are considered false positives. This way of evaluating is faulty as all points on line AB between the endpoints should have been considered true positives. On the contrary, the metric defined by ChartOCR [5] does linear interpolation to consider the above-demonstrated example.

3. Comparing Keypoint detection variants

We train keypoint detection models with $\mathrm{M} = 128, 256$ and $\alpha = 0.99$ the results can be found in Table 2. From here on $\mathrm{LINEEX}^{i}_{D+A}$ refers to the variant with $\mathrm{M} = i$. We also showcase the tradeoff between Recall and Precision in Fig 1 with respect to the number of keypoints used from the model.

4. Training Specifications

4.1. Keypoint Extraction

The keypoint detection model is adapted from PE-former [6]. We use the best performing variant of PE-former comprising of Cross-Covariance Image Transformer (XCiT) [3] as the encoder and DETR [4] based transformer decoder. The encoder was initialised with pretrained XCiT weights[1] and later finetuned on our dataset. The model was trained on 4 Tesla V100 GPUs with a batch size of 42 for 150 epochs.

4.2. Chart Element Detection

This module uses the popular transformer-based model, DETR [4], for predicting bounding boxes around chart components. We use the original DETR transformer architecture without making any changes to it, except for adapting

*Equal contribution

[1]https://dl.fbaipublicfiles.com/xcit/xcit_small_12_p16_384_dist.pth

		ExcelChart400K			Adobe Synthetic			Ours		
		Recall	Prec	F1	Recall	Prec	F1	Recall	Prec	F1
sim_{str}	ChartOCR	**0.85**	**0.98**	**0.90**	0.76	**0.72**	**0.71**	0.71	**0.90**	0.78
	LINEEX$_D$	0.80	0.67	0.68	0.94	0.37	0.50	0.84	0.74	0.75
	LINEEX$_{D+A}$	0.82	0.78	0.76	**0.95**	0.49	0.60	**0.86**	0.81	**0.81**
sim_{rel}	ChartOCR	0.85	**0.98**	**0.90**	0.78	**0.80**	**0.76**	0.74	**0.97**	0.83
	LINEEX$_D$	0.83	0.86	0.82	0.95	0.58	0.69	0.86	0.91	0.87
	LINEEX$_{D+A}$	**0.87**	0.89	0.84	**0.96**	0.65	0.74	**0.87**	0.93	**0.89**

Table 1: Keypoint extraction performance comparison between our proposed method and ChartOCR with no background detection step.

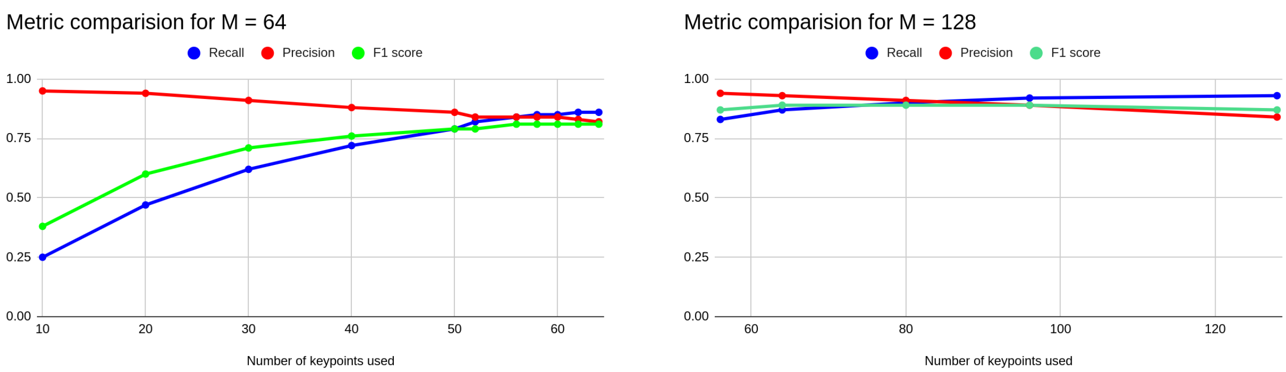


Figure 1: Influence of the number of keypoints used in the metrics

		ExcelChart400K			Adobe Synthetic			Ours		
		Recall	Prec	F1	Recall	Prec	F1	Recall	Prec	F1
sim_{str}	LINEEX$^{64}_{D+A}$	0.84	**0.80**	**0.78**	0.94	**0.67**	**0.74**	0.86	**0.84**	**0.83**
	LINEEX$^{128}_{D+A}$	0.88	0.60	0.66	0.93	0.57	0.65	**0.92**	0.66	0.73
	LINEEX$^{256}_{D+A}$	**0.89**	0.46	0.54	**0.96**	0.29	0.40	0.82	0.61	0.65
	RESNETLINEEX$_{D+A}$	0.25	0.20	0.19	0.22	0.32	0.23	0.79	0.73	0.73
sim_{rel}	LINEEX$^{64}_{D+A}$	0.85	**0.90**	**0.85**	0.93	**0.81**	**0.84**	0.87	**0.94**	**0.89**
	LINEEX$^{128}_{D+A}$	0.89	0.79	0.81	0.94	0.77	0.82	**0.93**	0.84	0.87
	LINEEX$^{256}_{D+A}$	**0.90**	0.60	0.68	**0.96**	0.47	0.58	0.83	0.77	0.77
	RESNETLINEEX$_{D+A}$	0.29	0.22	0.21	0.30	0.44	0.32	0.82	0.87	0.83

Table 2: Keypoint extraction metrics for M = 128.

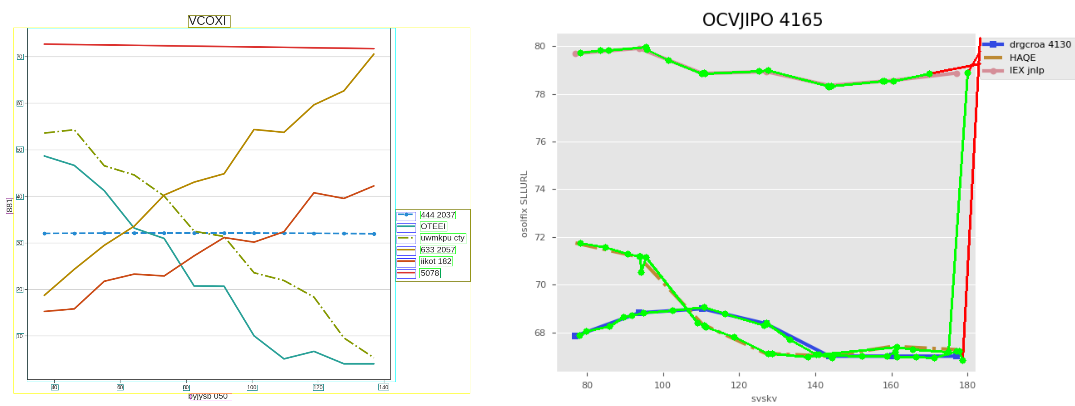
its last layer for the ten classes, as mentioned in the paper. The model was trained on 4 Tesla V100 GPUs with a batch size of 56. We use the official DETR code [1] and hyper-parameters as provided by the authors. The model was initialized with pretrained DETR weights [2] and finetuned on our dataset.

5. The Legend Mapping Dataset

The legend mapping dataset was constructed to contain sets of legend markers and line patches aggregated from multiple synthetically generated charts. For each distinct marker style, patches were sampled from lines across multiple charts and stored as a set. Patches of legend markers were also stored in a separate set for the same marker style. The legend mapping module was trained by providing the model with positive and negative samples for matching a line patch to a marker patch of the same style. The deep-ranking [8] algorithm was used, followed by an MLP to match patch similarity, as described in the paper.

6. Additional Qualitative Results

Here, we provide representative outputs of the element detection and legend mapping modules for a better intuitive understanding for the reader. Please refer to Figure 2.



(a) Visualizing output of the element detection module.

(b) Visualizing output of the legend mapping module

Figure 2: Visualizing outputs of the element detection and legend mapping modules. (a) Boxes represent the chart elements detected by LINEEX (b) Green dots and green lines represent the detected points and grouped lines, respectively. Red lines represent a mapping from a detected line to its corresponding legend marker.

References

[1] Detr model. https://github.com/facebookresearch/detr. [Online; accessed 20-Jun-2022].

[2] Detr weights. https://dl.fbaipublicfiles.com/detr/detr-r50-e632da11.pth. [Online; accessed 20-Jun-2022].

[3] Alaaeldin Ali, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. Xcit: Cross-covariance image transformers. *Advances in neural information processing systems*, 34, 2021.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229, 2020.

[5] Junyu Luo, Zekun Li, Jinpeng Wang, and Chin-Yew Lin. Chartocr: Data extraction from charts images via a deep hybrid framework. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*. The Computer Vision Foundation, January 2021.

[6] Paschalis Panteleris and Antonis Argyros. Pe-former: Pose estimation transformer. *arXiv preprint arXiv:2112.04981*, 2021.

[7] Noah Siegel, Zachary Horvitz, Roie Levin, Santosh Divvala, and Ali Farhadi. Figureseer: Parsing result-figures in research papers. *ECCV*, pages 664–680, 2016.

[8] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1386–1393, 2014.