Supplementary material
LayoutDM: Discrete Diffusion Model for Controllable Layout Generation

A. Implementation Details

A.1. Baseline

We explain more details on task-agnostic layout generation baselines using masking, especially when the original model is not designed for layout generation. We mostly describe unconditional generation cases, but partial layout fields can be easily injected by hard masking.

BART: BART is a denoising autoencoder and was originally designed for learning a sequence-to-sequence model for text generation. Text is usually represented as a 1D sequence of discrete tokens. Since we also handle the shuffled layout as a 1D sequence of discrete tokens during training, a BART-like model may be another solid baseline. To build a task-agnostic layout generation model, we apply random masking similar to the noise pattern of MaskGIT [1], instead of text-specific noises, such as span-level masking.

MaskGIT*: MaskGIT [1] is originally built for unconditional image generation. Following recent two-stage approaches for efficient image modeling, such as VQ-GAN [3], MaskGIT first generates a small number of discrete tokens and subsequently decodes those tokens into a continuous high-dimensional image by a pre-trained neural decoder. We consider the first generation part of MaskGIT to be another baseline. We use `[PAD]` to enable variable-length generation. For a masking schedule during decoding, *i.e*. fraction of the tokens masked in each iteration, we employ a cosine schedule as in MaskGIT.

VQDiffusion*: VQDiffusion [4] is a discrete diffusion-based model designed for text-to-image generation. To adapt VQDiffusion for conditional layout generation with minimal modification, we (i) remove the text conditioning branch in the reverse process, (ii) replace the image tokens with layout tokens, and (iii) add `[PAD]` token to enable variable-length generation. As described in the main manuscript, there are three major differences between VQDiffusion* and our proposed LayoutDM: modality-wise diffusion, decoupled positional encoding, and adaptive quantization.

We adjust the number of parameters for each model to have about 12M parameters for a fair comparison. We show the exact numbers in Tab. 1.

A.2. Relationship Guidance

Similarly to the main manuscript, let us denote the predicted coordinates of an i-th element $(\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i) \in [0, 1]^4$. We follow [9] to define the loss for penalizing size and location relationships between elements that do not match user specifications. For example, if we want to make the j-th element larger than the i-th element, the loss is defined by:

$$g_{lg}(i, j) = \max\left((1 + \gamma)\,\hat{w}_i\hat{h}_i - \hat{w}_j\hat{h}_j, 0\right), \quad (1)$$

where γ is a tolerance parameter, which is empirically set to 0.1. If we want to make the j-th element above the i-th element, the loss is defined by:

$$g_{ab}(i, j) = \max\left(\left(\hat{y}_j + \frac{\hat{h}_j}{2}\right) - \left(\hat{y}_i - \frac{\hat{h}_i}{2}\right), 0\right), \quad (2)$$

which compares the bottom of the j-th element and the top of the i-th element. Please refer to the code for losses for the rest of the relationships.

Although it is not experimentally demonstrated, we believe that it is also possible to incorporate area, aspect ratio, and reading order constraints used in Attribute-conditioned GAN [14].

- Area: given a target area of the element $a_i \in \mathbb{R}$, we use $|a_i - \hat{h}_i\hat{w}_i|$ as a loss.
- Aspect ratio: Given a target aspect ratio $r_i \in \mathbb{R}$, we use $|r_i - \frac{\hat{h}_i}{\hat{w}_i}|$ as a loss.
- Reading order: we follow [22] and define that the reading order solely depends on the distance between the left-top of the canvas and each element. We first compute the distance by $\hat{d}_i = \sqrt{(\hat{x}_i - \frac{\hat{w}_i}{2})^2 + (\hat{y}_i - \frac{\hat{h}_i}{2})^2}$. We can use $\max(\hat{d}_i - \hat{d}_j, 0)$ as a loss to make the i-th element come before the j-th element in the reading order.

A.3. Hyper-parameters

We search for the best hyper-parameters using a validation set. During sampling from $p_\theta(\boldsymbol{z}_{t-1}|\boldsymbol{z}_t)$ for all the tasks, we search for p used in nucleus (or top-p) sampling [7] out of $\{0.90, 0.95, 0.99, 1.0\}$. We train the models for 50 and 20 epochs in Rico and PubLayNet, respectively.

	Rico	PubLayNet
LayoutVAE [8] (C→S+P)	13.2	13.0
NDN-none [11] (C→S+P)	21.8	21.8
LayoutGAN++ [9] (C→S+P)	12.9	12.9
LayoutVAE [8] (C+S→P)	14.7	14.5
NDN-none [11] (C+S→P)	14.8	14.8
LayoutGAN++ [9] (C+S→P)	13.0	13.0
LayoutTrans [5]	12.7	12.7
LayoutTrans-fixed [5]	12.7	12.7
MaskGIT* [1]	12.7	12.7
BLT [10]	12.7	12.7
RUITE [17]	12.7	12.7
BART [12]	12.8	12.8
VQDiffusion* [4]	12.4	12.4
LayoutDM	12.4	12.4

Table 1. The number of parameters [M] used for each model.

We attempt a grid search for additional hyper-parameters in the refinement task. The ranges of possible values are the following: the distance margin m in $\{0.1, 0.2\}$ and the weighting term λ_π in $\{1.0, 2.0, 3.0, 4.0, 5.0\}$.

A.4. Evaluation

In unconditional generation, the model generates 1,000 samples from the random seed. In conditional generation, the test set of each dataset is used to make a partial input for conditional generation and the model generates one sample per each data in the test set.

B. Additional Results

B.1. Ablation Study

State space Continuous state space diffusion models have gained much attention compared to discrete state space models. Recently, Li *et al*. [15] propose DiffusionLM that adapts the continuous models to handle discrete text generation. DiffusionLM introduces an embedding and rounding step to bridge the continuous and discrete state spaces. We train DiffusionLM (with 12.6M parameters) and show the results in Tab. 2. We show the results of DiffusionLM with embedding dimensions $d = 16$ because it works best out of $\{16, 64, 128\}$ in Rico [2] dataset. Although We tried different samplers (DDPM [6] and DDIM [18]) and training timesteps, DiffusionLM is still far behind the discrete state space models in layout generation as shown in Tab. 2.

Refinement The logit adjustment proposed in the main manuscript has some choices for injecting positional prior. Without loss of generality, we describe a constraint that imposes the x-coordinate estimate of i-th element close to the noisy continuous observation $\hat{x}_i$. We denote a sliced vector of the prior term $\pi(\boldsymbol{z}_{t-1})$ that corresponds to the x-coordinate of i-th element as $\boldsymbol{\pi}_x^i \in \mathbb{R}^K$.

	State	#steps	Sampler	FID ↓
LayoutDM	dis.	100	-	**6.65**
VQDiffusion* [4]	dis.	100	-	<u>7.46</u>
DiffusionLM [15]	con.	100	DDIM	34.5
	con.	100	DDPM	24.8
	con.	1000	DDIM	33.8
	con.	1000	DDPM	22.8

Table 2. Ablation study results on the choice of state spaces: discrete (dis.) and continuous (con.), in the unconditional generation task of Rico [2] dataset. Top two results are highlighted in **bold** and <u>underline</u>, respectively.

	FID ↓	Max. ↑	Sim ↑
Default	**2.77**	**0.370**	**0.205**
Gaussian	5.82	<u>0.330</u>	<u>0.188</u>
Negation	<u>3.78</u>	0.276	0.169

Table 3. Ablation study results on the choice of logit adjustment methods in the refinement task. Top two results are highlighted in **bold** and <u>underline</u>, respectively.

- Gaussian: j-th token is more likely to be sampled when $\mathrm{loc}(j)$ is closer to $\hat{x}_i$. The prior is defined by:

$$\left[\boldsymbol{\pi}_x^i\right]_j = \begin{cases} \left(\mathrm{loc}\left(j\right) - \hat{x}_i\right)^2 & \text{if } |\mathrm{loc}(j) - \hat{x}_i| < m \text{ and } j \in X \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

  The ranges of possible values are similar to our method used in the main manuscript (Default).
- Negation: j-th token is never sampled when $\mathrm{loc}(j)$ is far away from $\hat{x}_i$. The prior is defined by:

$$\left[\boldsymbol{\pi}_x^i\right]_j = \begin{cases} 0 & \text{if } |\mathrm{loc}(j) - \hat{x}_i| < m \text{ and } j \in X \\ -\infty & \text{otherwise.} \end{cases} \tag{4}$$

  The ranges of possible values are the following: the distance margin m in $\{0.2, 0.4, 0.6, 0.7, 0.8, 0.9\}$.

We show the quantitative evaluation results in Tab. 3. We can see that Default outperforms other possible choices by a large margin.

B.2. Speed-Quality Trade-off

We show more speed-quality trade-off curves in Fig. 1. We perform generation with a batch size of 64 and report the average runtime to generate a single layout for all the models. Lightly colored regions around the line plots, such as

the one in BLT for C→S+P in Rico represent the standard deviation of three trials for each model, though the deviations are too small to see in most cases.

B.3. More Results

We show more results compared with task-specific baselines in C→S+P (Fig. 2), C+S→P (Fig. 4), unconditional generation (Fig. 6), the refinement task (Fig. 8) for PubLayNet. Typical failure cases are frequent overlap between elements (often in BLT), unnecessarily broad blank space (often in LayoutTrans.), and lack of diversity. We show more results in C→S+P (Fig. 3), C+S→P (Fig. 5), unconditional generation (Fig. 7), the refinement task (Fig. 9) for Rico. Rico is more difficult to generate since the number of categories is large and elements are less aligned compared to PubLayNet.

B.4. Diversity-Fidelity Trade-off

We introduce density and coverage metrics by [16] to analyze the results from a different viewpoint. Density measures fidelity; *i.e.*, how closely generated samples resemble real ones. Coverage measures diversity; *i.e.*, whether generated samples cover the full variability of the real samples. We plot the diversity and fidelity of iterative refinement-based models in Fig. 10 as we increase the number of timesteps for the iterative prediction. Discrete diffusion-based models usually have higher coverage scores and lower density scores. We conjecture that the coverage difference comes from the inference decoding strategy. BLT [10] and MaskGIT* [1] fix high-confident predictions and re-initialize lower-confident fields by `[MASK]` for the next step that leads to higher fidelity. In contrast, discrete diffusion-based models *randomly* corrupt the predictions and result in higher diversity.

B.5. Alignment and Overlap

We additionally show the metrics reported in many previous works: Alignment and Overlap. Note that these metrics only capture the fidelity of generated layouts. There are a few variants for both Alignment [9, 11, 13, 14] and Overlap [9, 13, 14]. We employ the definition in [9]. We scale the values of Alignment by $100\times$ for visibility. For reference, we show Alignment and Overlap computed in a validation set as *Real data*. The lowest score in Alignment or Overlap does not always mean the best performance for a model, but a model closest to *Real data* is the best model. We show the result in Fig. 11. In the fixed-length generation *i.e.* C→S+P and C+S→P, LayoutDM performs almost comparably to VQDiffusion* [4] and BART [12], and better than the other models. In the variable-length generation *i.e.* the completion task and unconditional generation, autoregressive models, such as BART [12] and LayoutTrans. [5], are moderately better than LayoutDM. This result is reasonable since these models predict the fields one by one. Diffusion-based models, such as LayoutDM and VQDiffusion*, are better than BLT and MaskGIT*. We believe this is because diffusion models avoid the error accumulation in iterative prediction according to [4].

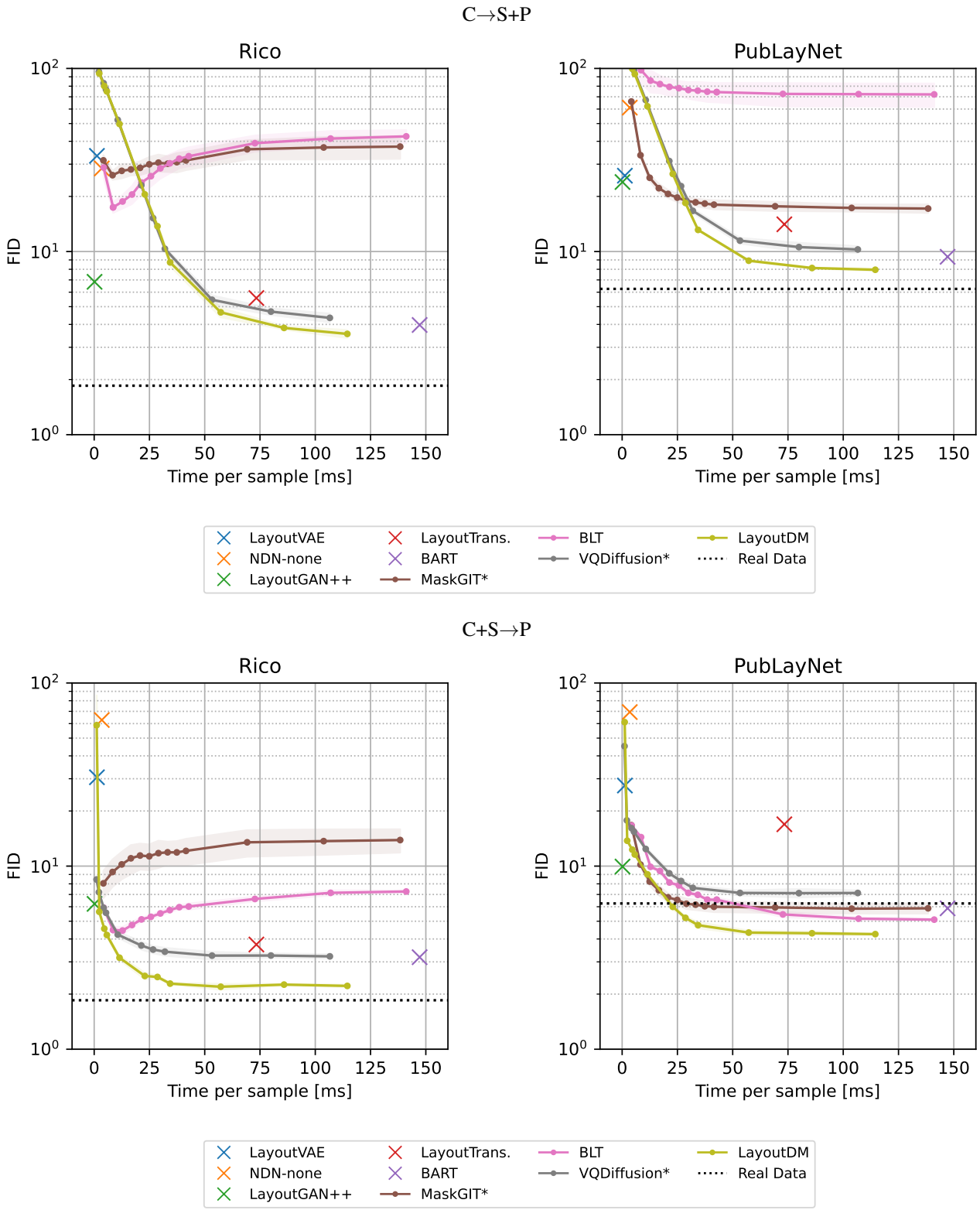
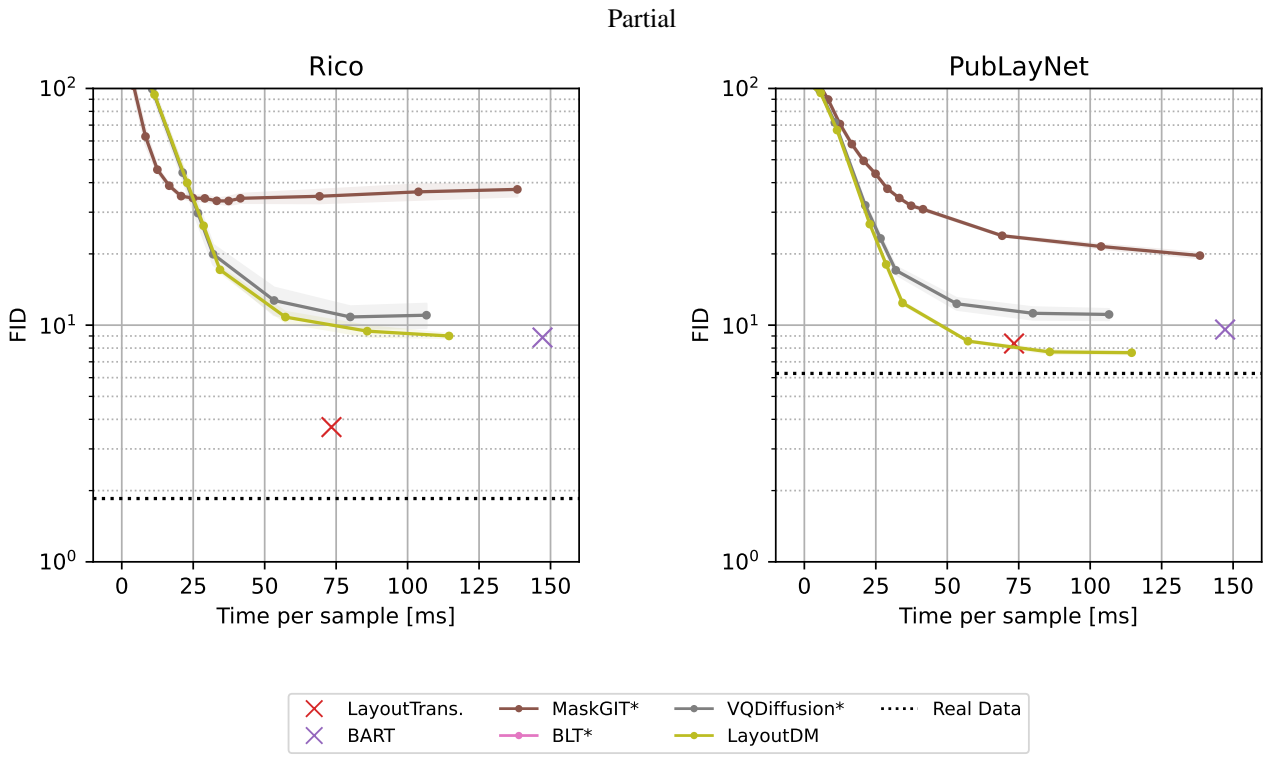


Figure 1. Speed-quality trade-off of different models.



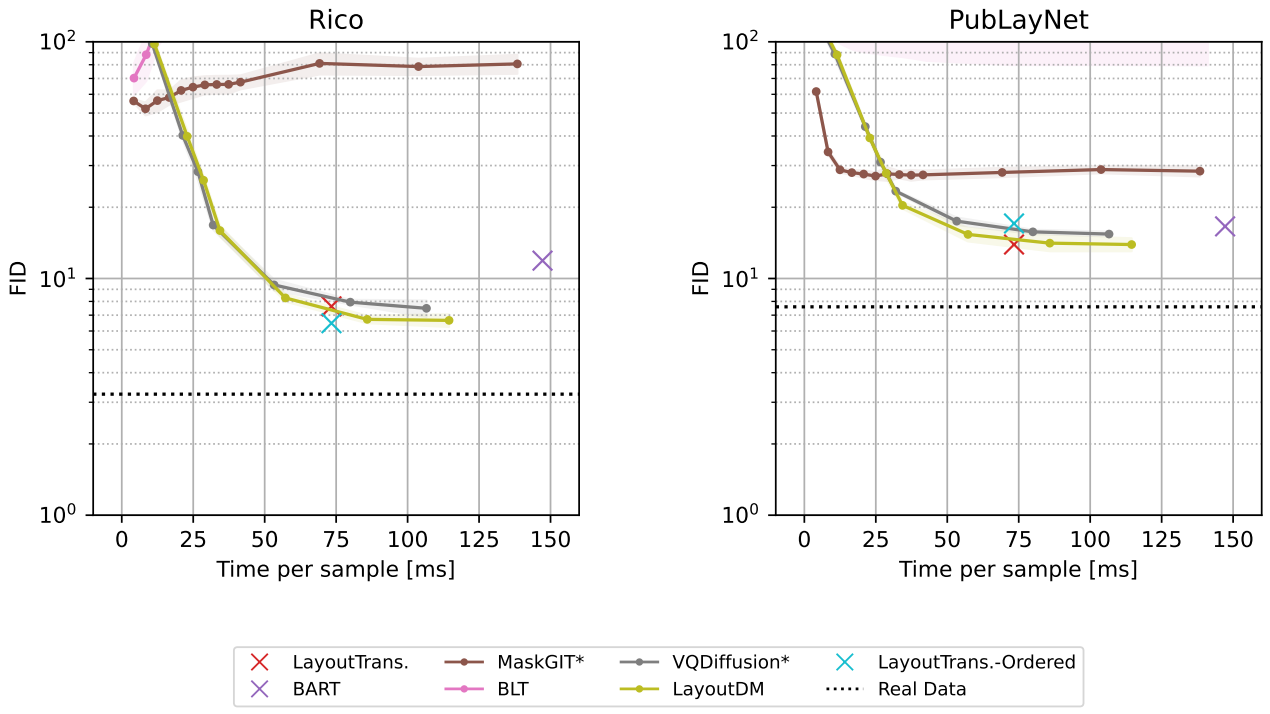


Figure 1. (cont.) Speed-quality trade-off of different models.

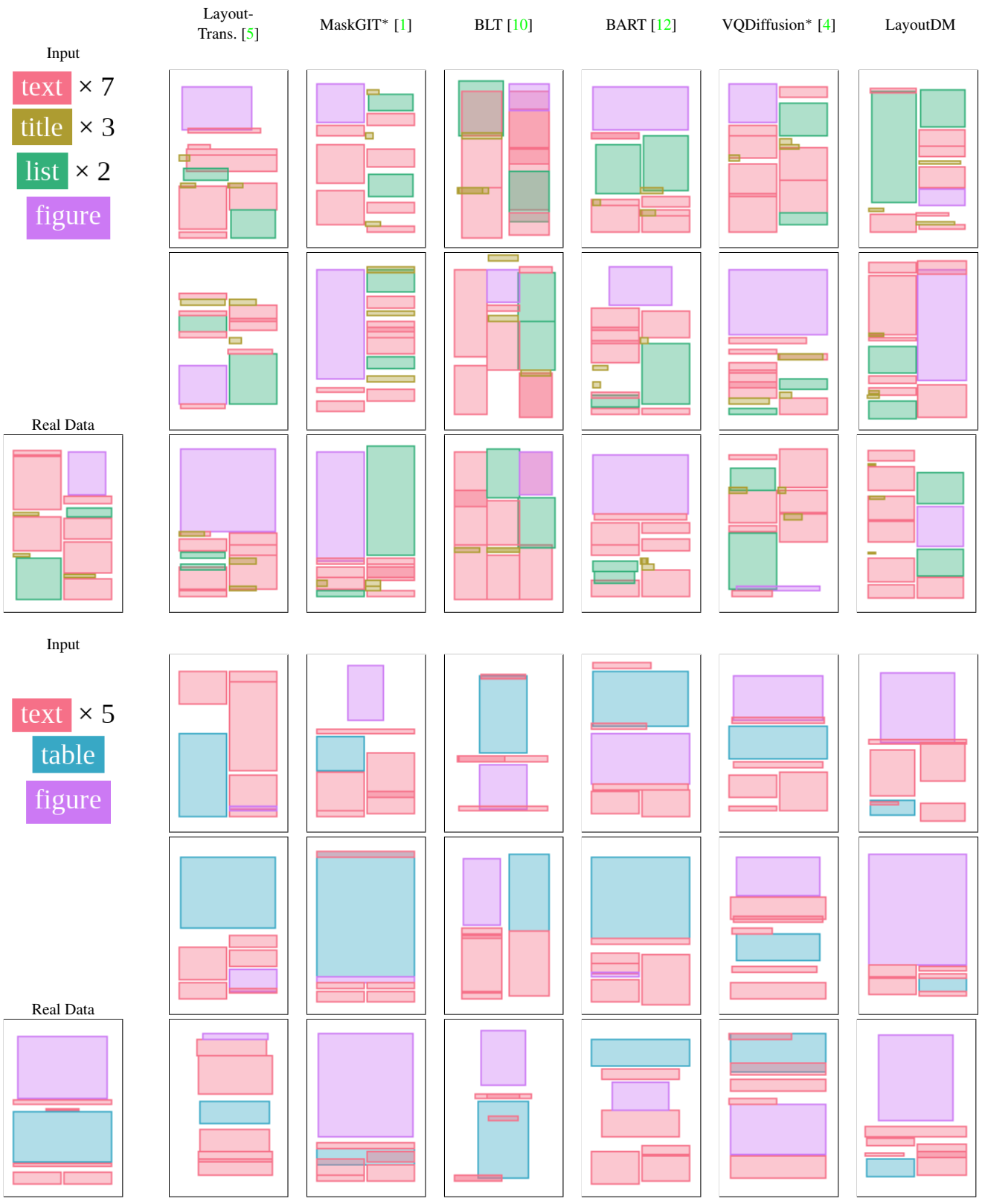


Figure 2. Comparison of conditional generation in C→S+P for PubLayNet. We obtain three samples from each model to demonstrate the diversity.

Figure 3. Comparison of conditional generation in C→S+P for Rico. We obtain three samples from each model to demonstrate the diversity.

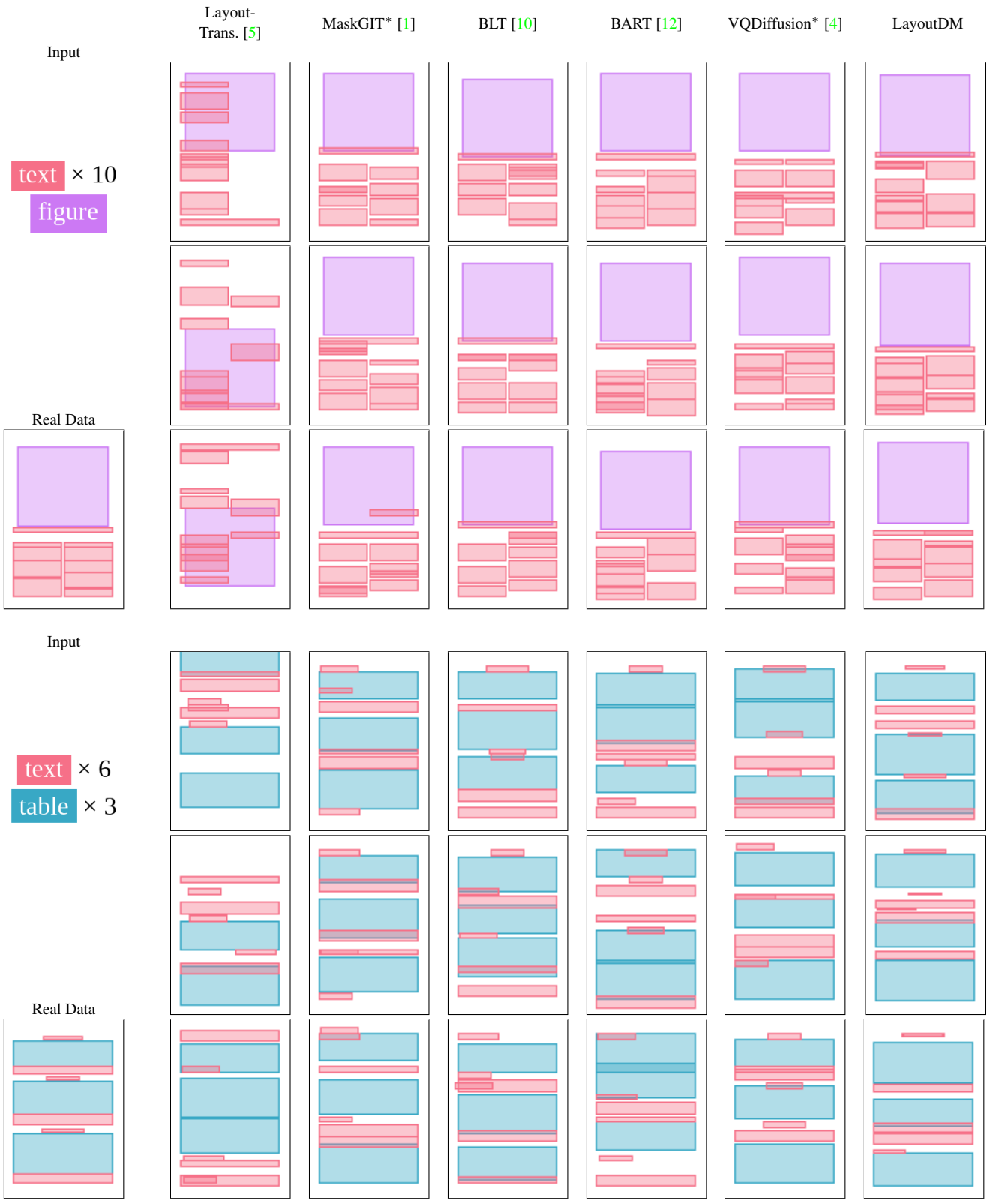


Figure 4. Comparison of conditional generation in C+S→P for PubLayNet. We obtain three samples from each model to demonstrate the diversity. Note that the size condition of each element is not shown for limited space. Please refer to Real Data for the size.

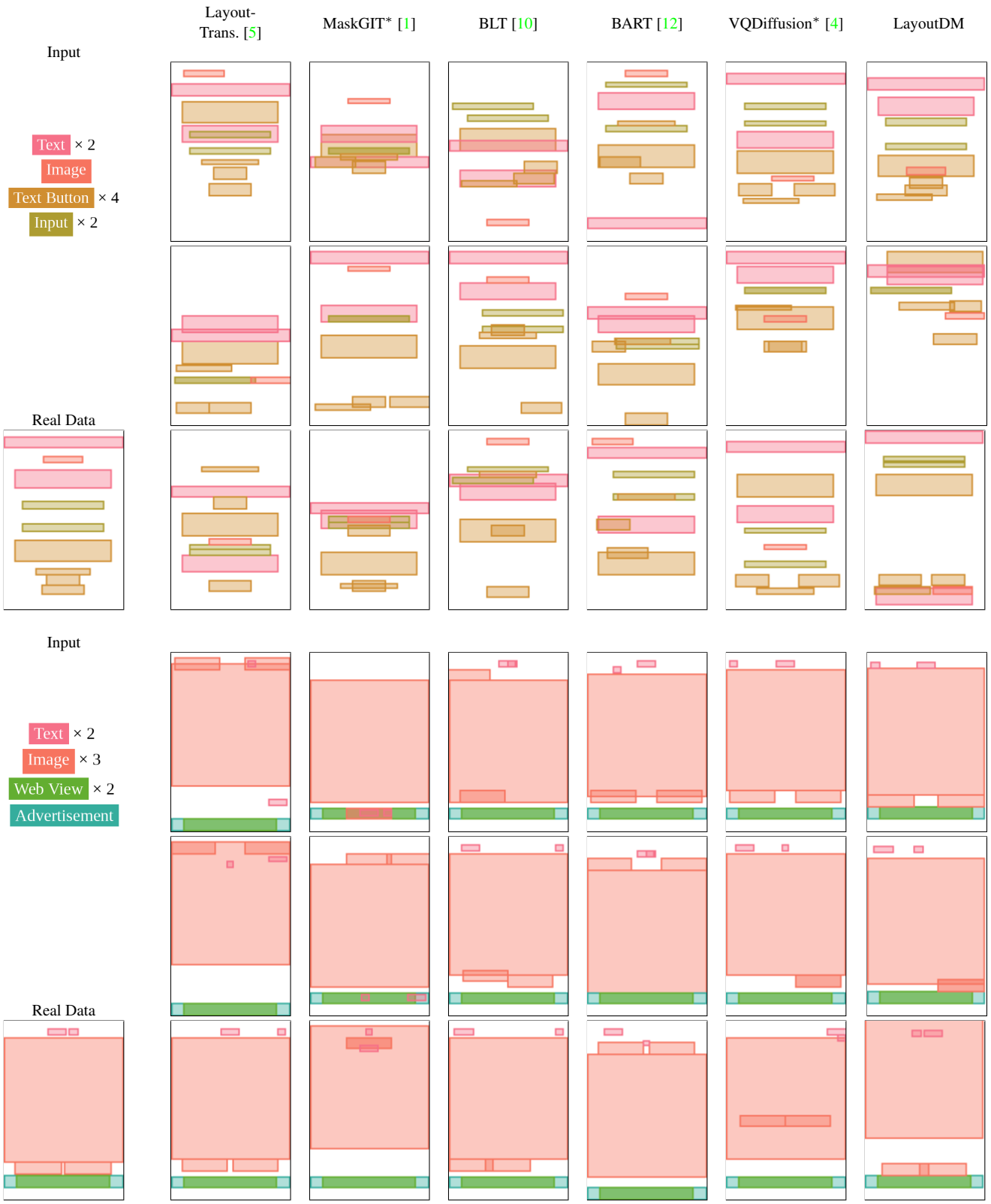


Figure 5. Comparison of conditional generation in C+S→P for Rico. We obtain three samples from each model to demonstrate the diversity. Note that the size condition of each element is not shown for limited space. Please refer to Real Data for the size.

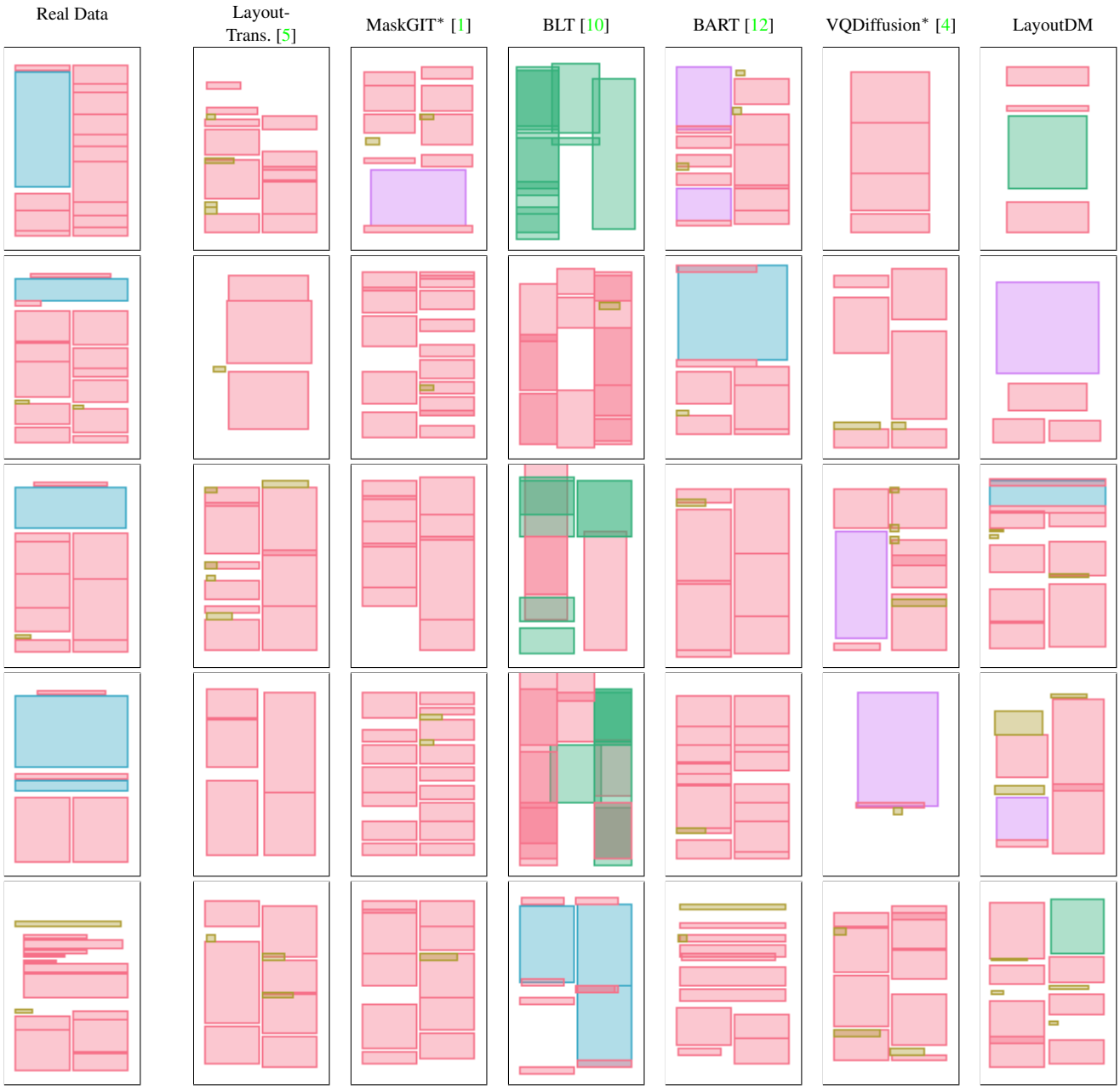


Figure 6. Comparison of unconditional generation for PubLayNet. We obtain five samples from each model to demonstrate the diversity.

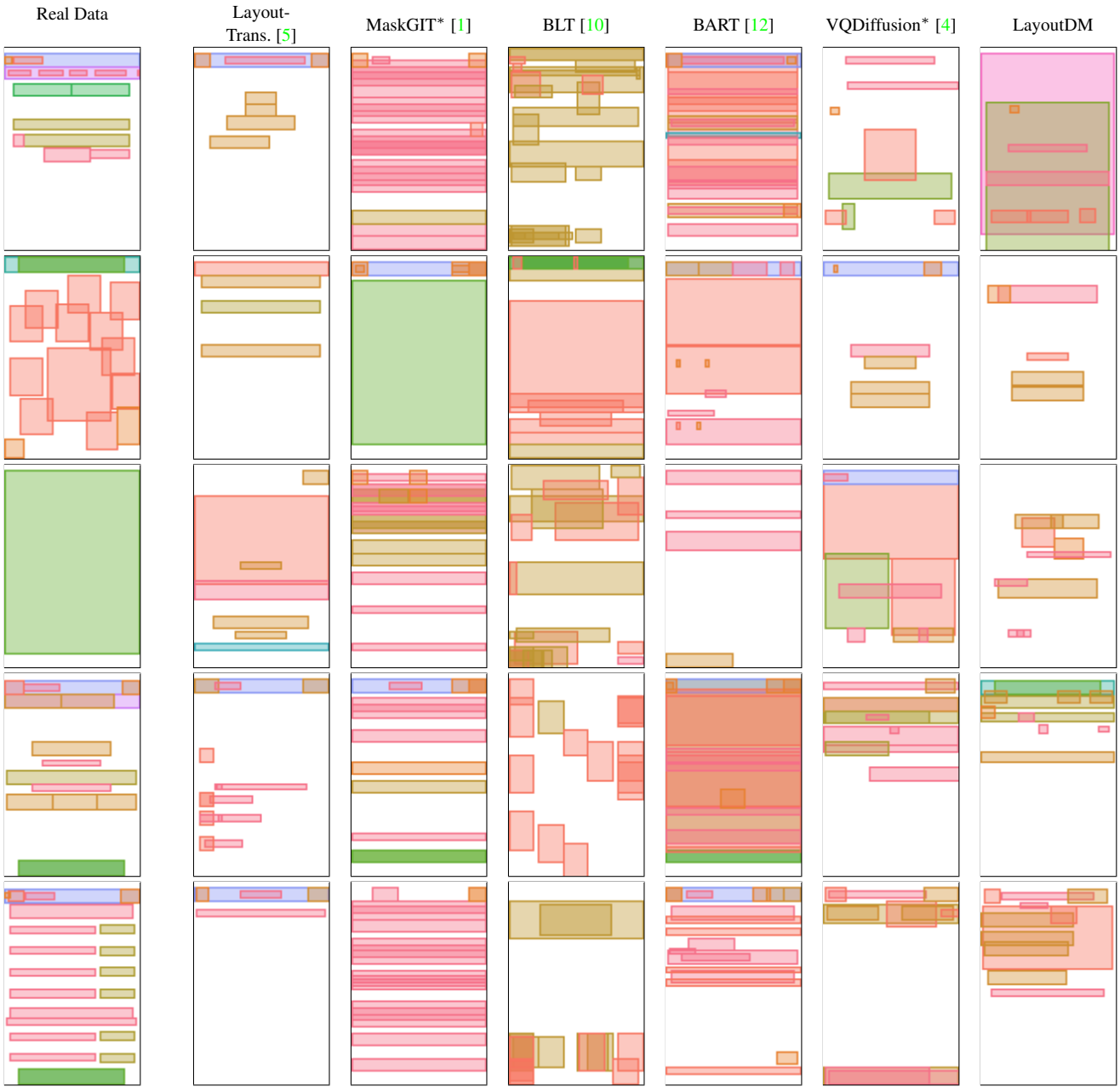


Figure 7. Comparison of unconditional generation for Rico. We obtain five samples from each model to demonstrate the diversity.

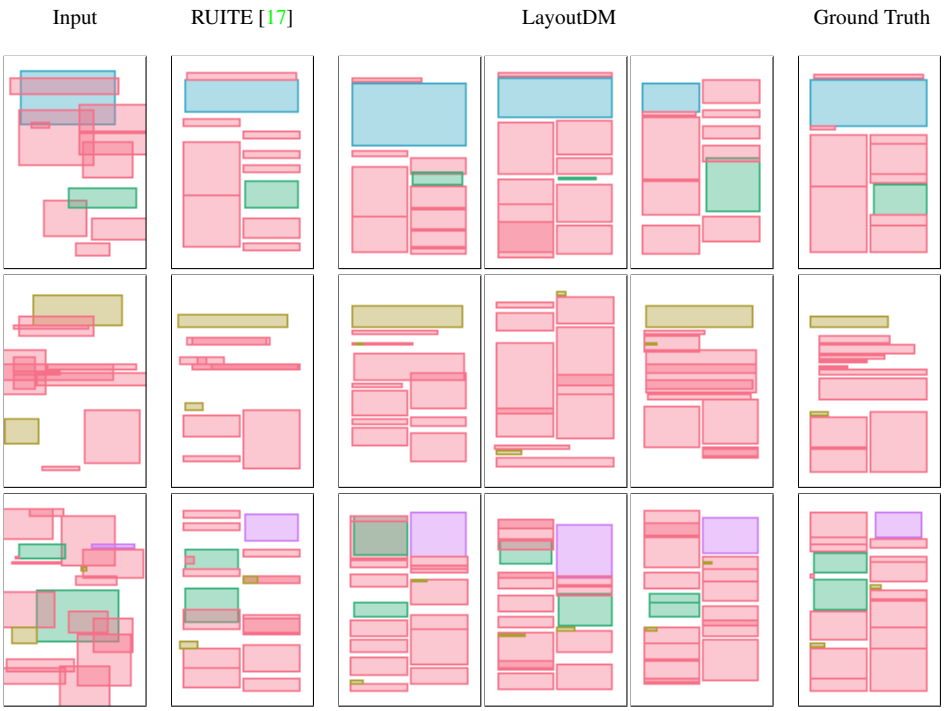


Figure 8. Comparison of the refinement task for PubLayNet. We obtain three samples from LayouytDM to demonstrate the diversity.

Figure 9. Comparison of the refinement task for Rico. We obtain three samples from LayouytDM to demonstrate the diversity.

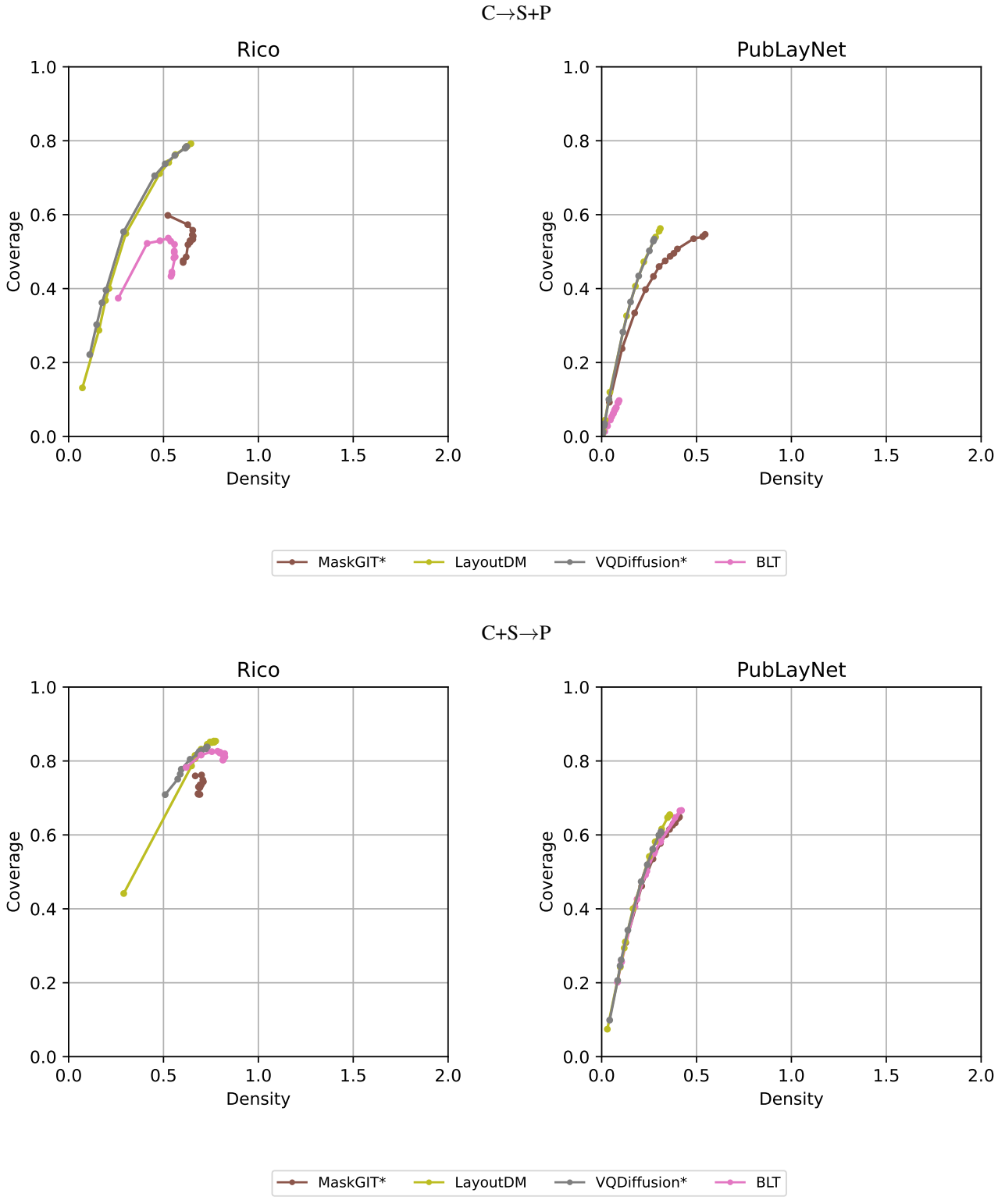


Figure 10. Density-coverage trade-off of different models.

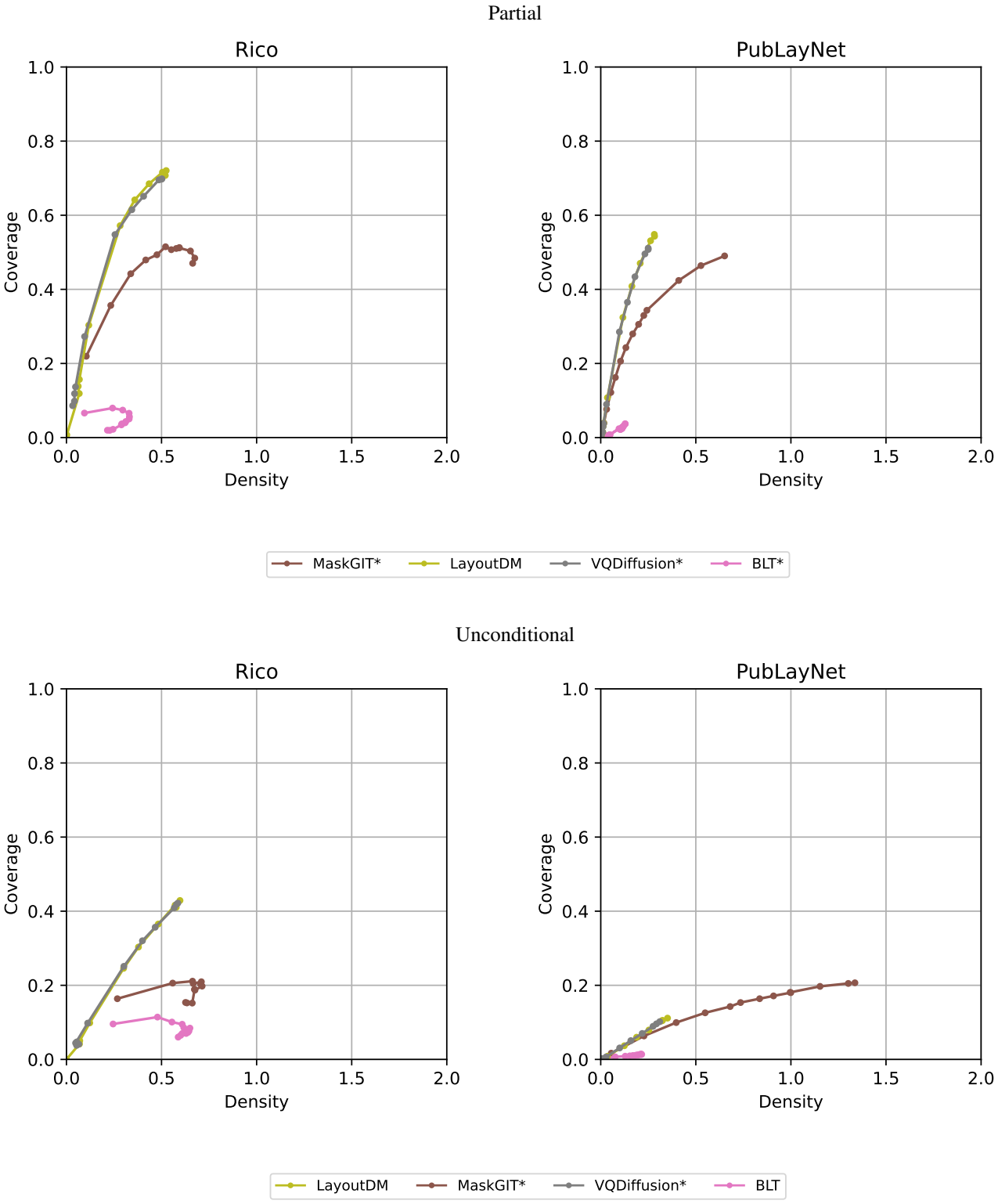


Figure 10. (cont.) Density-coverage trade-off of different models.

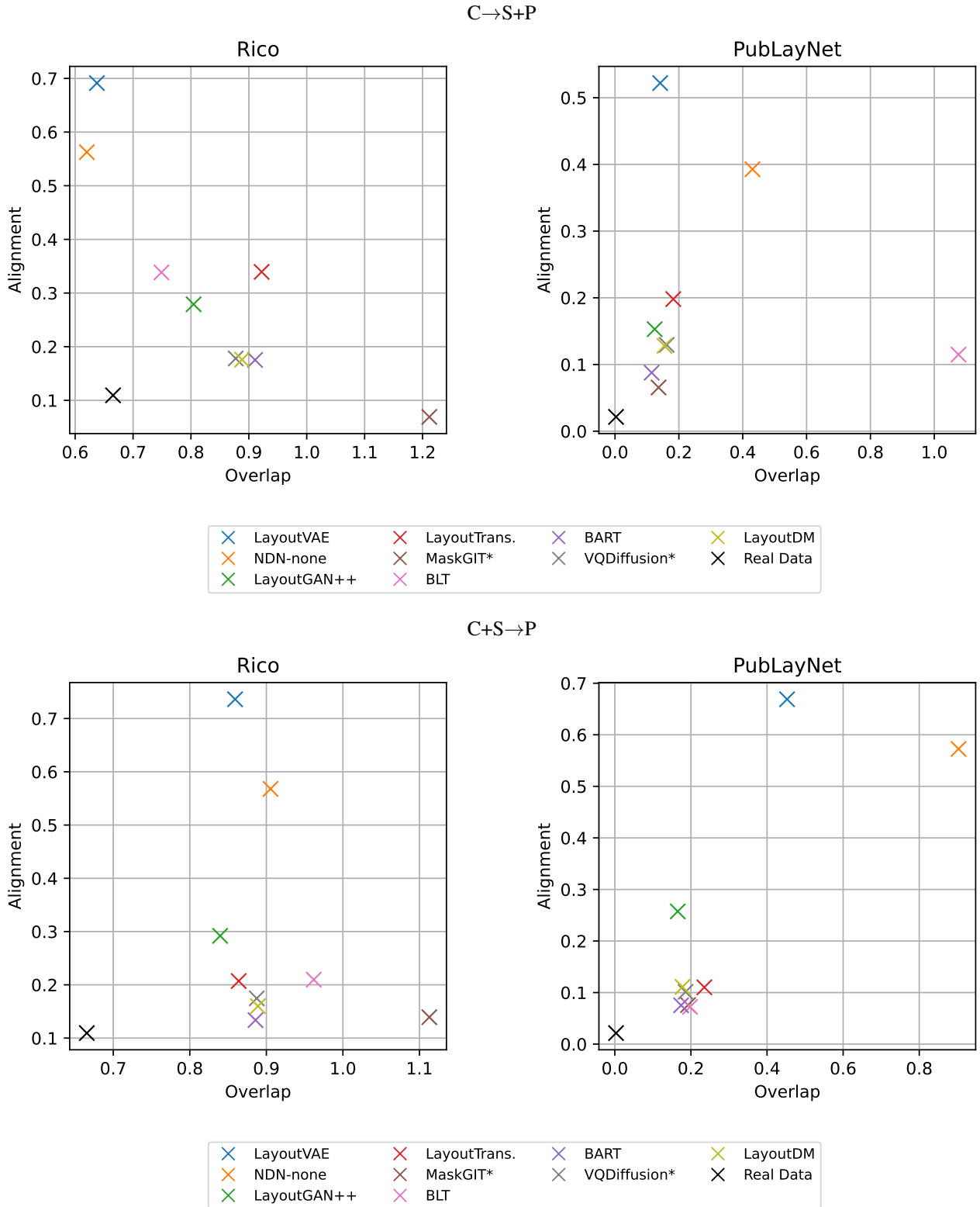
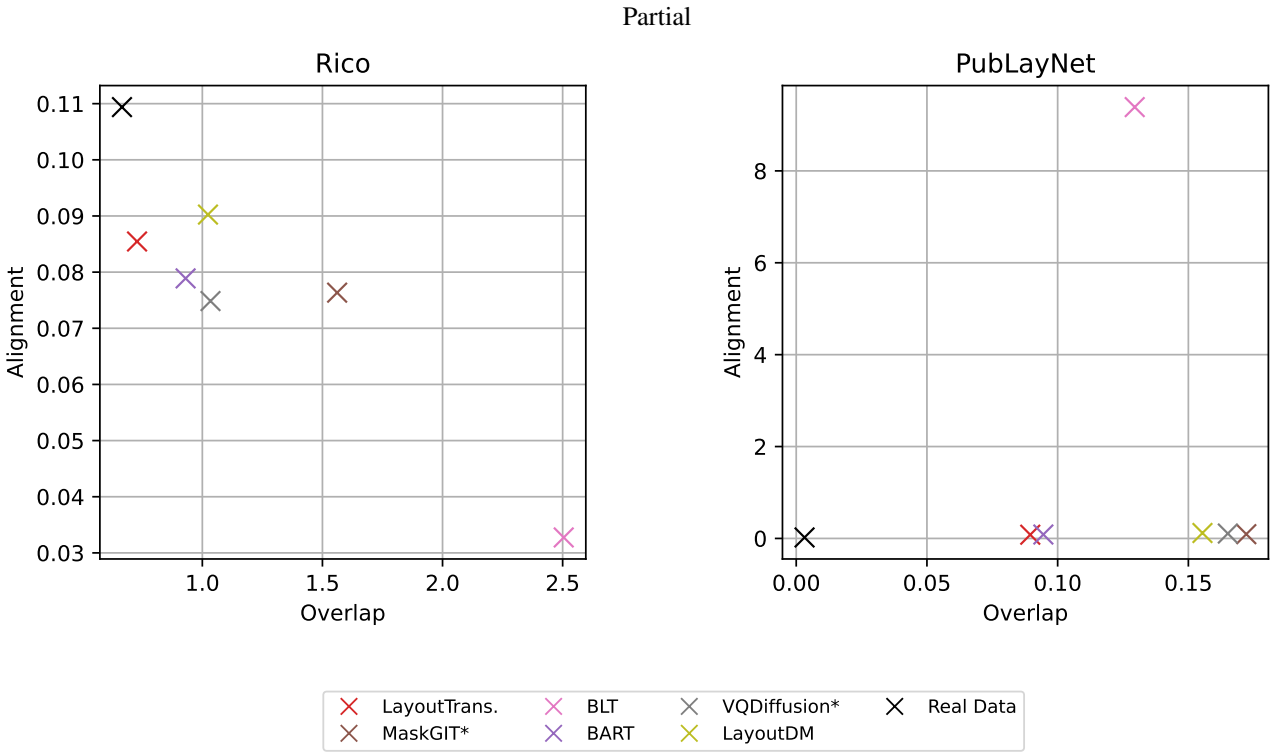


Figure 11. Alignment and overlap of different models.



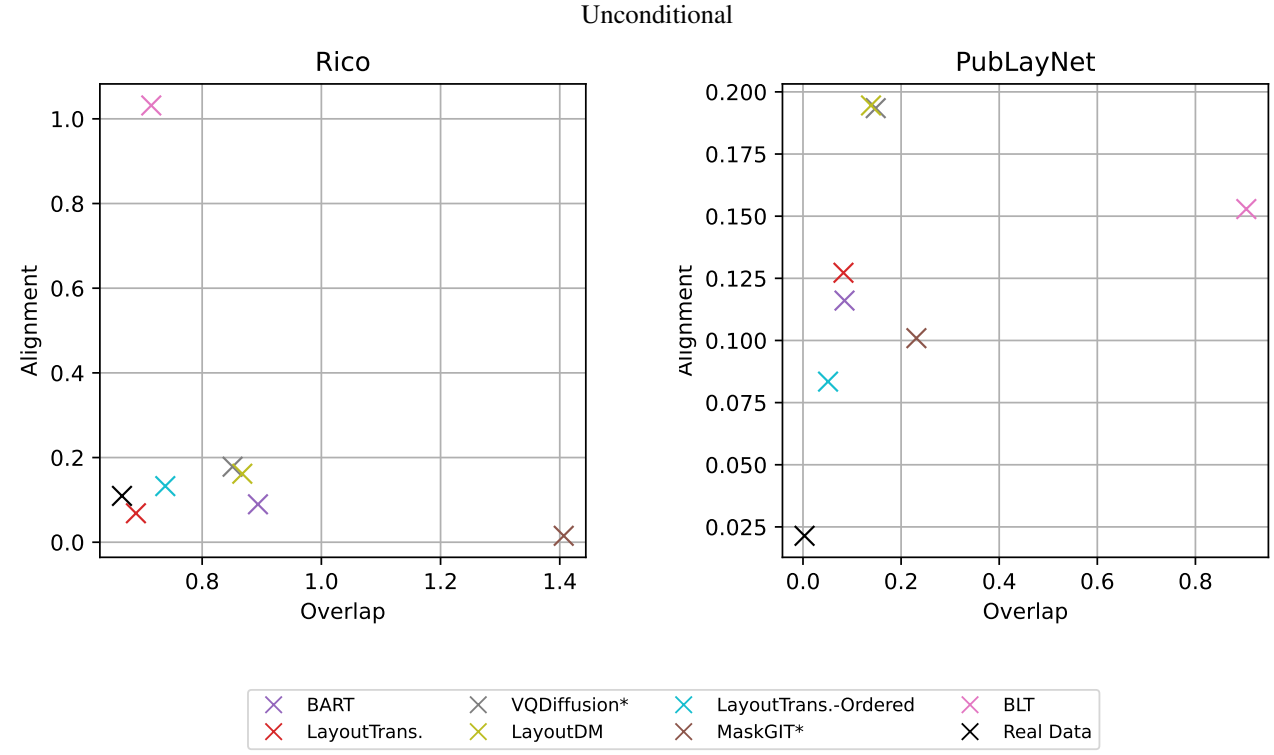


Figure 11. (cont.) Alignment and overlap of different models.

References

[1] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *CVPR*, 2022. 1, 2, 3, 6, 7, 8, 9, 10, 11

[2] Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *UIST*, 2017. 2

[3] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 1

[4] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *CVPR*, 2022. 1, 2, 3, 6, 7, 8, 9, 10, 11

[5] Kamal Gupta, Alessandro Achille, Justin Lazarow, Larry Davis, Vijay Mahadevan, and Abhinav Shrivastava. Layout-Transformer: Layout generation and completion with self-attention. In *ICCV*, 2021. 2, 3, 6, 7, 8, 9, 10, 11

[6] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2020. 2

[7] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *ICLR*, 2019. 1

[8] Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. LayoutVAE: Stochastic scene layout generation from a label set. In *CVPR*, 2019. 2

[9] Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. Constrained graphic layout generation via latent optimization. In *ACM MM*, 2021. 1, 2, 3

[10] Xiang Kong, Lu Jiang, Huiwen Chang, Han Zhang, Yuan Hao, Haifeng Gong, and Irfan Essa. BLT: Bidirectional layout transformer for controllable layout generation. In *ECCV*, 2022. 2, 3, 6, 7, 8, 9, 10, 11

[11] Hsin-Ying Lee, Weilong Yang, Lu Jiang, Madison Le, Irfan Essa, Haifeng Gong, and Ming-Hsuan Yang. Neural design network: Graphic layout generation with constraints. In *ECCV*, 2020. 2, 3

[12] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 2020. 2, 3, 6, 7, 8, 9, 10, 11

[13] Jianan Li, Jimei Yang, Aaron Hertzmann, Jianming Zhang, and Tingfa Xu. LayoutGAN: Generating graphic layouts with wireframe discriminators. In *ICLR*, 2019. 3

[14] Jianan Li, Jimei Yang, Jianming Zhang, Chang Liu, Christina Wang, and Tingfa Xu. Attribute-conditioned layout gan for automatic graphic design. *IEEE TVCG*, 27(10), 2020. 1, 3

[15] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. Diffusion-LM improves controllable text generation. In *NeurIPS*, 2022. 2

[16] Muhammad Ferjad Naeem, Seong Joon Oh, Youngjung Uh, Yunjey Choi, and Jaejun Yoo. Reliable fidelity and diversity metrics for generative models. In *ICML*, 2020. 3

[17] Soliha Rahman, Vinoth Pandian Sermuga Pandian, and Matthias Jarke. RUITE: Refining ui layout aesthetics using transformer encoder. In *26th International Conference on Intelligent User Interfaces-Companion*, 2021. 2, 12

[18] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021. 2