Image Inpainting Models are Few-Shot Learners (Given the Right Data)

Anonymous Author(s)
Affiliation
Address
email

Abstract

How does one adapt a pre-trained visual model to novel downstream tasks *without task-specific finetuning or any model modification*? Taking inspiration from prompting in NLP systems, this paper investigates visual prompting: given input-output image example(s) of a new task at test time and a new input image, the goal is to automatically produce the correct output image, consistent with the proposed task. We show that posing this problem as a simple image inpainting task – literally just filling in a hole in a concatenated visual prompt image – turns out to be surprisingly effective, given that the inpainting algorithm has been trained on the right data. We train masked auto-encoding models on a new dataset that we curated - 88k unlabeled figures from academic papers sources on Arxiv. We apply visual prompting to these pretrained models and demonstrate results on various downstream tasks, including foreground segmentation, single object detection, colorization, edge detection, etc. All our code, models, and dataset will be made available.

1 Introduction

In the past few years, self-supervised learning has gained popularity in computer vision and natural language processing (NLP). The growing capacity of modern deep learning models made them prone to overfitting when trained on relatively small labeled datasets. Self-supervised learning provided a solution to this problem by generating "free labels" for any dataset, without the need for manual annotation, addressing the data hunger in these high-capacity deep learning models. However, features learned via self-supervision are not "ready for use" – they typically need to be adapted for a given downstream task by fine-tuning on some labeled dataset. Could this fine-tuning be avoided?

In NLP, prompting [5] has recently emerged as a way to employ a model for a new task without any additional training. A common way of task-prompting for a specific language understanding task at test time is to provide the trained model with input corresponding to example(s) of the target task together with the query. E.g., typing the following input prompt:

```
Je suis désolé              I'm sorry
J'adore la glace
```

will prompt the model [5] to perform the task of French-to-English translation, returning:

```
I love ice cream
```

Can this idea of test-time task prompting be generalized to the visual domain? That is, instead of the current situation in computer vision, where each trained model serves its predefined task (e.g.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

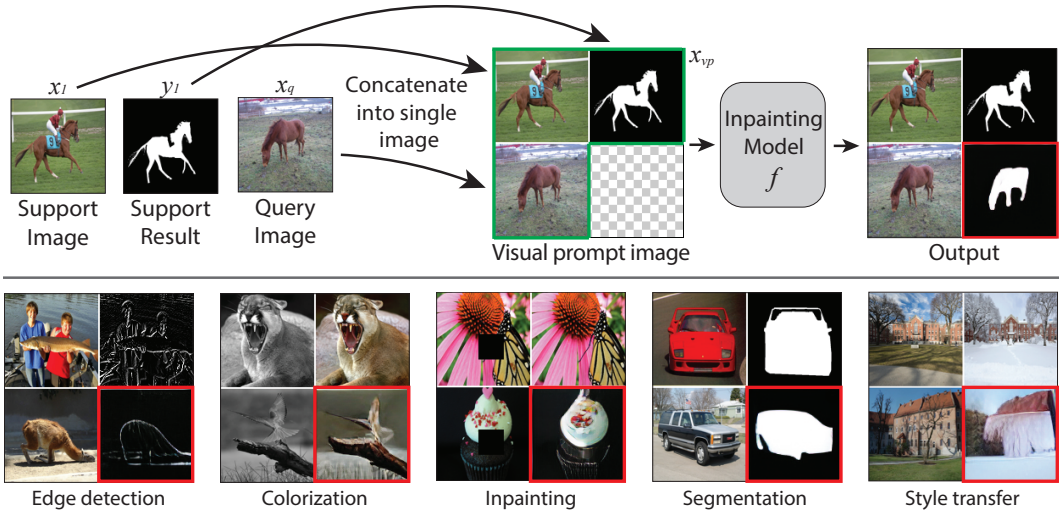

Figure 1: **Visual prompting by Image Inpainting.** *Top*: Prompting Image Inpainting Models. Given support example(s) (x_1, y_1) and image query x_q, we construct a grid-like single image called a *visual prompt* x_{vp}. The visual prompt is composed of the desired task example(s) and a new query image (all in green). The inpainting model goal is then to predict the masked region (red) such that it is consistent with the example(s). *Bottom*: an inpainting model can solve this way various computer vision tasks, given that it was trained on the right data. The model predictions are annotated in red.

segmentation, detection, classification), can we have a single general model that is able to perform a wide range of user-specified tasks *without any fine-tuning (i.e., weight modification)*?

In this paper we take a step toward this goal by demonstrating that large-capacity inpainting models, when trained on the right data, can be surprisingly effective tools for *visual prompting*. As shown in Figure 1, we define each task by constructing a new grid-like image that contains an input-output example(s) of the task and a novel query (green border). The input-output example describes the task, and the image query defines a new input. The model then produces the result by simply inpainting the rest of the image (red border). This setting is most similar to the classic Image Analogies [22] formulation, but is less constrained: instead of explicitly defining the A, A', and B images separately, we simply concatenate them into a single image with a hole (hence, visual prompting is not exactly an analogy since there is no implied left-to-right ordering). Our goal is also not dissimilar from the aims of meta-learning and few-shot learning methods, except that we make no distinction between tasks and example pairs. The only requirement of our formulation is that the tasks must be defined in terms of image-to-image translations, which is a very large subset of vision problems.

To obtain data that fits our framework, we utilize a domain that spans across a variety of computer vision tasks - figures and infographics from computer vision articles available on Arxiv. We build a large dataset of 88k figures, many of which contain grids of images and their corresponding task results (e.g. images and their segmentation masks/stylized versions/edges, etc.). We then train large-capacity inpainting models to predict randomly masked patches from figures given other patches from the same figure.

Our main contributions are as follows. First, we present a simple yet surprisingly powerful general approach for visual prompting. We show that various computer vision tasks can be treated as grid inpainting problems, given a few examples of task inputs and outputs and a query image. Second, we provide a new dataset that allows a model to learn such grid structures without any labeling, task descriptions, or any additional information about the grid structure. Finally, we show that, while using our new dataset for training is essential, adding more generic images from other sources (e.g. ImageNet) further improves the results.

2 Related Work

Natural Image Inpainting. Filling empty regions in an image has been widely explored for natural images. Earlier methods used data from the input image itself for inpainting [14, 4, 11, 3, 46], whereas later works utilized datasets of images as source data [19, 36, 54, 29, 30]. Recent methods have attempted to apply transformers to visual synthesis tasks [8, 56, 15, 55, 7]. Due to the exponentially large number of completion options for a single output patch, these approaches rely on a discrete latent codebook [49, 38, 15] which serves as a smaller yet expressive vocabulary. To tackle the multimodal nature of synthesis, different approaches have been proposed to model the distribution over possible completions [15, 55, 7]. For example, [15, 55] proposed to synthesize images line-by-line using an autoregressive model and [7] have proposed iterative parallel decoding. While the standard inpainting task typically aims to complete blank parts in natural images, our focus is on completing grid-like visual prompts, which require reasoning across multiple images to infer the completion.

Hole-filling as a Pretext task. Recent work has shown that self-supervised pretraining can generate powerful representations for transfer learning, even outperforming its supervised counterparts on challenging vision benchmarks [48, 18, 9, 21, 6, 10, 34, 16]. Pathak et al. proposed context encoding [36], where the goal is to predict an image region given its context. Based on the recent success of Vision Transformers (ViTs) [13], multiple works have proposed the Masked image Auto-Encoding (MAE) [1, 20, 52] as a self-supervised pretext task for Vision Transformers (ViTs). The goal of MAEs is to reconstruct the image given a small subset of input patches. After pretraining MAEs on unlabeled data, they produce representations that transfer well when fine-tuned on downstream tasks. Here, we use visual prompting to adapt these models to downstream tasks without any finetuning.

Few-Shot Learning. In this setting, the algorithm is trained on a labeled dataset of base classes, from which it should transfer to a set of novel classes given only a few training examples (like 10 or 30) [35, 26, 31, 51, 53, 45, 57, 2]. Unlike Few-Shot approaches, here we do not assume access to a large training set of base-classes, and our architecture is not task-specific. Our approach is Few-Shot only in the sense that we construct a visual prompt that contains one or two task examples.

Image Analogies. Hertzmann et. al. [22] proposed the framework of Image Analogies for texture synthesis, where the algorithm is given a pair of training images and a query image. The goal of the learner is to synthesize a new image conditioned on the query, following the relationship inferred from the training pair. Other works have used analogies in style transfer [47], and as a supervised image synthesis task [39]. Predicting the correct completion was previously modeled as a classification problem [41, 23], and other works have explored analogies in the context of learning different transformations between pairs of images [33, 44]. Unlike these approaches, we use inpainting MAE models that learn from data, without assuming any predefined analogies structure.

Prompting in NLP. With the recent success of large unsupervised language models [42, 12], Brown et al. [5] presented how a variety of NLP problems can be reformulated to a text completion problem given a predefined prompt, which can be used to solve different tasks without any finetuning. Prompting was shown to be a useful tool for solving various NLP tasks and benchmarks [37, 5]. More recently different approaches to prompting have emerged including Prompt Engineering [5, 32], Prompt Ensembling [24], and Prompt Prefix Tuning [28, 27]. Inspired by the success of prompting in NLP, we aim to study prompting in computer vision where prompting hasn't been widely explored.

3 Visual Prompting using Image Inpainting

We turn to describe how to perform visual prompting using Image Inpainting models. In Section 3.1, we describe Inpainting Models. We then proceed to discuss visual prompting and propose different ways to create visual prompts in Section 3.2. An illustration of the approach is included in Figure 1.

3.1 Inpainting using MAE-VQGAN

Given an input image $x \in \mathbb{R}^{H \times W \times 3}$ and a binary mask $m \in \{0,1\}^{H \times W}$, the goal of an inpainting function f is to synthesize a new image $y \in \mathbb{R}^{H \times W \times 3}$, with the masked regions filled:

$$y = f(x, m)$$

To implement f with a neural network, it is necessary to consider design choices like the network architecture, how to train it, and whether it outputs a distribution over possible completions or pixels. We propose the MAE-VQGAN model, which combines ideas from MAE [20] and VQGAN [15]. Following the design of MAE, the model is based on ViT [50, 13] and it is trained via masked auto-encoding, by randomly masking image patches then applying f to reconstruct the image from the non-masked parts. MAE-VQGAN models the distribution $p_\theta(z_i|x, m)$, where $z_i \in V$ is a visual token from a VQGAN vocabulary V that corresponds to the ith ViT patch. For simplicity, we use a fixed ImageNet pretrained VQGAN codebook.[1] Unlike MAE which directly predicts pixels, MAE-VQGAN assigns probabilities to visual tokens via a softmax layer, which is better suited for capturing ambiguities. During training, we obtain ground truth visual tokens by mapping the image to visual tokens indices using the VQGAN encoder. The model is trained using cross entropy loss.

Let $\hat{z} = (\hat{z}_1, ..., \hat{z}_k)$ be the ordered set of predicted visual tokens. To obtain $\hat{z}_i$, we use argmax:

$$\hat{z}_i = \arg\max_{z_i} p_\theta(z_i|x, m)$$

Then, to decode the visual tokens to pixels, we apply VQGAN decoder to $\hat{z}$ to obtain y.

3.2 Prompting Inpainting Models

To prompt an inpainting model, we construct a *visual prompt*, a grid-like image composed of support examples, and a new query image. The model then has to inpaint the rest of the image such that it is consistent with the task defined in the support, applied over the image query (see Figure 1).

Let $S = \{(x_i, y_i)\}_{i=1}^{n}$ be the set of support examples where x_i is an image and y_i is a function of x_i (e.g y_i is a segmentation mask). We assume n is small (one or few examples). Then, given S and a new input query x_q, the goal is to predict the corresponding label y_q. To prompt the inpainting model discussed in Section 3.1, we need to define a function g that maps the support set S and query image x_q to a new image and a mask:

$$[x_{vp}, m] = g(S, x_q)$$

The image x_{vp} is the visual prompt and the mask m defines the masked region f has to predict. For a given task, there might exist multiple implementations of g that can be considered. The goal of the inpainting model is to reason about the visual prompt x_{vp}, and output a plausible completion without performing any additional training:

$$y_{vp} = f(x_{vp}, m)$$

To obtain y_q, we just take the part of y_{vp} corresponding to the mask m.

For the visual prompting to work, g should output a good visual prompt, composed of the supports S and query image x_q. Therefore, g has to determine where and how to embed the inputs in the visual prompt image, considering the nature of the completion task. All the functions g used in this work were hard-coded and manually engineered. In most cases, g stacks the support pairs and image query horizontally by creating an image grid of $(n+1) \times 2$ cells, where the ith support pair is placed in the ith row, and the image query is in the last row. The grid has a fixed size, and therefore before populating it the support pairs and query are first resized. Another consideration is how to draw every support pair (x_i, y_i). For example, if y_i is a segmentation mask, we can choose to use different colors to draw it. In Section 4.4, we describe different prompt design choices and their effect on the results.

Visual Prompt Ensembling. There could be multiple options to define g. The idea in prompt ensembling, inspired by NLP [24, 27], is to construct multiple different prompts, apply the inpainting model f on each prompt individually to obtain a set of predictions. The final prediction can be determined, for example, via majority voting, or weighted average. For simplicity, here we use a simple average.

[1] Using a publicly available checkpoint from https://github.com/CompVis/taming-transformers

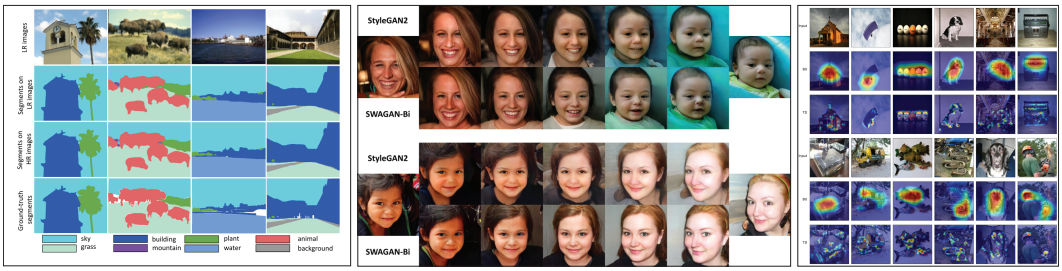

Figure 2: **Example images from our Computer Vision Figures dataset**. We curated a dataset of 88k unlabeled figures from Computer Vision academic papers.

3.3 The Computer Vision Figures Dataset

The images produced by g are by construction not natural. Specifically, these images have a grid-like figure structure that stitches together images coming from different distributions, like natural images and segmentation masks. Therefore, a model trained on a standard dataset (e.g., ImageNet [40]) might struggle to process these grid-like images. To mitigate the domain gap, we collect a new dataset.

The Computer Vision Figures (Figures) dataset consists of $88,645$ images that more closely resemble the structure of our visual prompts. The dataset was collected from Arxiv, the open-access web archive for scholarly articles from a variety of academic fields. Arxiv sources are publicly available to download starting from 2010. We downloaded all paper sources from 2010 to 2022 and selected the Computer-Vision partition "cs.CV" sources, as they contain images that more closely resemble a grid structure, as shown in Figure 2. To remove unrelated source images like graphs or charts, we manually tagged 2000 images and trained a binary image classifier to assign a high score to source images in a figure-like structure with at least one natural image. We then used the classifier over the entire data to keep only the most informative source images, coming from $23,302$ different papers. We randomly partitioned 90% of the data to train and left the rest for validation. We include additional examples and provide more information about the dataset in the Supplementary Material.

4 Experiments and Results

To study visual prompting, we pretrain different models (see Section 4.1) on ImageNet and on the Figures dataset, then evaluate the models using different prompts on simple downstream computer vision tasks (see Section 4.2). Using a synthetic dataset, we assess how the choice of model and data affect the success of visual prompting in Section 4.3, and explore different prompting design choices in Section 4.4. We include additional experiments and visualizations in the Supplementary Material.

4.1 Models and Baselines

To study the effect of model choice on prompting results, we experiment using different models, including MAE-VQGAN (see Section 3.1) and several other inpainting models briefly described below.

VQGAN [15] is an autoregressive transformer model used for inpainting and image generation. Visual tokens are predicted sequentially, line-by-line, and the model is trained using cross-entropy loss. The VQGAN model codebook is used to encode visual tokens, and it is trained beforehand using perceptual loss [25] and GAN loss [17]. We train it on ImageNet and our Figures dataset, following hyperparams in [15], and use a pretrained codebook with a vocabulary of size $|V| = 1024$.

BEiT [1] is a masked auto-encoding model. The model maps each input 16×16 patch to a visual token from a d-VAE [38] vocabulary of size 8192. To encode each visual token, the image is first resized to 112×112 and then mapped to 196 tokens. We use the publicly available BEiT large model, pretrained on ImageNet-21k. We also pretrain a large BEiT model on Figures for 1000 epochs.

Table 1: **Visual prompting results on computer vision tasks.** For Foreground Segmentation and Single Object Detection, we report the *mIOU* score. For Colorization, we report the *MSE*.

Model	Foreground Segmentation ↑				Single Object Detection ↑				Colorization ↓	
	Split 0	Split 1	Split 2	Split 3	Split 1	Split 2	Split 3	Split 4	MSE	LPIPS
Copy	12.92	17.90	13.52	15.29	12.14	13.50	13.03	12.38	2.63	0.75
BEiT (IN-21k)	0.38	0.93	0.90	0.95	0.24	0.32	0.19	0.10	1.25	0.73
VQGAN (IN-1k)	6.96	10.55	9.59	9.43	5.19	4.99	5.09	5.10	2.44	0.66
MAE (IN-1k)	1.92	6.76	3.85	4.57	1.37	1.98	1.62	1.62	1.13	0.87
MAE-VQGAN (IN-1k)	2.22	7.07	5.48	6.28	3.34	3.21	2.80	2.80	3.31	0.75
BEiT (Figures)	5.38	3.94	3.20	3.29	0.17	0.02	0.14	0.16	0.60	0.70
VQGAN (Figures)	12.56	17.51	14.27	15.06	2.27	2.37	2.48	1.99	1.50	0.56
MAE (Figures)	17.42	25.70	18.64	16.53	5.49	4.98	5.24	5.84	**0.43**	0.55
MAE-VQGAN (Figures)	**27.83**	**30.44**	**26.15**	**24.25**	**24.19**	**25.20**	**25.36**	**25.23**	0.67	**0.40**

MAE [20]. Similar to BEiT, MAEs attempt to reconstruct a masked input image. Unlike in BEiT, the model directly regresses pixels and it is trained with $l2$ loss. During pretraining, only non-masked tokens are fed into the encoder, which results in a faster training time. We use a publicly released checkpoint pretrained on ImageNet, and pretrain another model for 1000 epochs on our dataset.

Copy Support. This simple baseline simply replicates the first support example label as the output.

Implementation Details. All the models we describe are large transformer-based models [50, 13], with patch size 16×16, embedding dim 1024, 24 layers, and 16 heads. For training, we used a machine with 8 Quadro RTX 6000 GPUs, with a batch size of 48. The input image size is 224×224.

4.2 Downstream Computer Vision Tasks

We evaluate the inpainting models described above on computer vision tasks, as described below.

Visual Prompt. Given one support pair and a query image, we structure the prompt in the same fashion for all tasks. We construct a grid of 2×2 sub-images, where the support pair is embedded in the first row, and the query image appears in the bottom left cell. See the example in Figure 1.

Foreground Segmentation. The goal is to binary-segment the query image to Foreground and Background. The support example is an image and corresponding binary segmentation mask. The query is a new image, and the goal is to complete a corresponding segmentation mask. We use the Pascal-5i [43] dataset, which is comprised of 4 different image splits where every split contains between 346 and 725 images and associated segmentation masks. For each class, the data contains a few support image-mask pairs, together with held-out image queries. For every image query, we choose one random support pair. To evaluate, every pixel in the completed image is first mapped to the nearest Foreground or Background color. Finally, we report the mean IOU (mIOU) metric.

Single Object Detection. Similarly to Foreground Segmentation, the goal here is to binary-segment the object that appears in the query image. However, this task is more challenging than Foreground Segmentation because the support mask is obtained from a bounding box which is more coarse than a segmentation mask. We use the Pascal VOC 2012 dataset using images and their associated detection boxes. For simplicity, we use Pascal annotations to include only images with a single object and filter out trivial images that have an object covering more than 50% of the image. We randomly select a support and image query of the same object class and repeat the process with 4 different random seeds. For evaluation, we follow a similar process as in Foreground Segmentation to obtain a binary segmentation mask. Then we keep the connected component with the largest area using morphological operations and draw a bounding box around it. Finally, we report the mIOU results.

Colorization. The goal is to map a gray-scale image to a color image. The support part contains a gray-scaled image and the corresponding color image, as shown in Figures 1 and 3. We randomly sampled 1000 image pairs for support and image query from ImageNet [40] validation set and converted them to gray-scale to obtain gray-scale and color version for each image. We report the MSE loss.

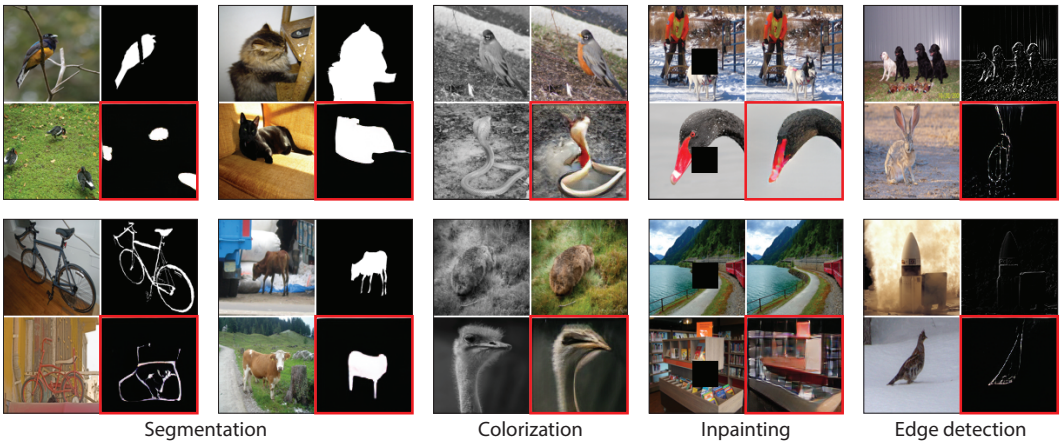

Figure 3: **Visual prompting prediction examples.** Each visual prompt was fed to an MAE-VQGAN model trained on the Figures dataset. For each visual prompt, the result is marked in red.

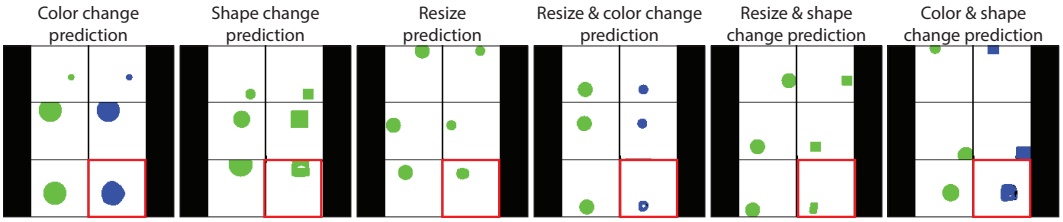

Figure 4: **Synthetic data study results.** MAE-VQGAN predictions are annotated with a red square.

Results. We include quantitative results in Table 1, and qualitative completion results in Figure 3. Training on the Figures dataset improves the results for most models in all the downstream tasks. MAE-VQGAN outperforms the other models by a large margin for detection and segmentation and generates much sharper images than the MAE. We find that VQGAN struggles to output accurate results, likely due to sequential decoding. The BEiT model is outperformed by MAE, most likely because its training process is less sample efficient. For more results, see the Supplementary Material.

4.3 Synthetic Data Study

To assess the compositional prediction capabilities of inpainting models, we created a set of 3 simple synthetic tasks and 3 of their combinations, and evaluated each model on 100 examples per task.

Visual Prompt. Given two support pairs and a query image, we structure the prompt in the same fashion for all tasks. We construct a grid of 3×2 sub-images, where the support pairs are embedded in the first two rows, and the query image in the bottom left cell. We include examples in Figure 4.

Change prediction. For each support pair (x_i, y_i), x_i is an image of a colored shape, and y_i is the same image with an introduced change. The change can be either in color, shape, size or a combination of two changes. We describe the tasks in more detail in the Supplementary Material.

Evaluation. We map each predicted pixel to its nearest neighbor color from a predefined set of options: black, white, blue, or green. We then measure and report the *color-aware* mIOU, by considering pixel predictions that appear in the ground-truth shape color as foreground and treat the rest as background.

Results. The results are presented in Table 2, for MAE-VQGAN prediction examples see Figure 4. Without training on the Figures dataset, inpainting models fail to generalize to these previously unseen tasks. The performance of all models increases when they are trained on the Figures dataset. Yet, the same models struggle with combinations of tasks due to the increasing complexity. The VQGAN model utilizes sequential decoding and therefore lacks context, which leads to poor performance. The

Table 2: **Synthetic data study results.** We report the color-aware mIOU on the six tasks.

	Color	Shape	Size	Color & Shape	Color & Size	Shape & Size
Copy	5.53	6.71	1.17	6.74	1.17	1.86
VQGAN (IN-1k)	0.91	6.51	6.24	2.40	0.70	6.53
BEiT (IN-22k)	15.99	9.08	1.26	7.23	2.84	2.66
MAE (IN-1k)	0.00	2.07	1.20	0.00	0.00	1.56
MAE-VQGAN (IN-1k)	0.13	2.94	3.71	0.00	0.01	3.60
VQGAN (Figures)	6.96	19.11	16.21	7.40	2.24	18.41
BEiT (Figures)	40.92	31.43	7.12	**33.10**	**21.21**	12.98
MAE (Figures)	**70.23**	43.99	34.72	19.30	18.99	**46.02**
MAE-VQGAN (Figures)	40.40	**46.53**	**42.04**	20.41	18.27	40.33

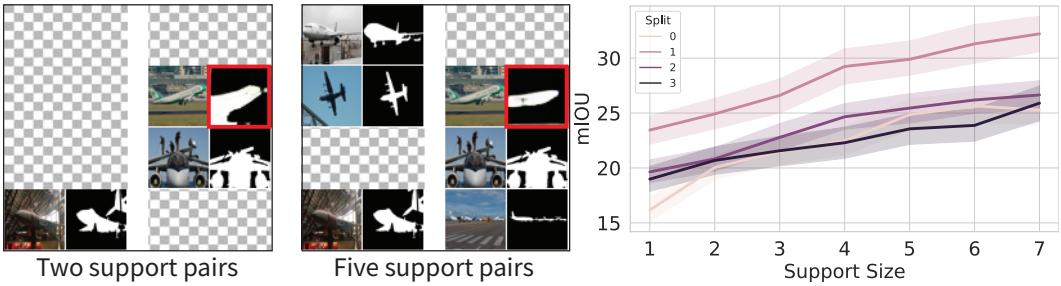

Figure 5: **More supports, better results.** *Left*: we construct visual prompts with increasing number of support examples, for a fixed query image (inpaintings annotated in red). *Right*: We observe that more examples improve the overall mIOU results on the four Pascal-5i splits.

MAE model outperforms MAE-VQGAN on color, and BEiT performs poorly in size. These models rely on pretrained codebooks (VQGAN and dVAE) that are likely not geared towards these tasks.

4.4 Analysis

More supports, better results. We study how increasing the number of supports in the visual prompt affects the results. Intuitively, we expect that including more support examples should reduce ambiguities and lead to better results. We use an MAE-VQGAN pretrained on the Figures dataset, and use data from PASCAL-5i. We construct a large grid that can populate up to 8 supports and an image query. We randomly choose different numbers of supports and randomize the placements. The results in Figure 5 confirm that using more support examples leads to better segmentation results.

Prompt Engineering. We explore the effect of constructing different visual prompts for Foreground Segmentation and their corresponding MAE-VQGAN results (see Figure 6.a-c). The model generates plausible completions when changing the prompt layout (e.g. horizontal order vs. vertical order) and when changing the mask colors (black and white vs. purple yellow). The mIOU results in Table 3 indicate that the model performs better with a vertical layout and when the segmentation mask colors are black and white. Interestingly, by analyzing the average attention heads of a masked patch token, we observe that the attention changes following the change in the prompts layout (see Figures 6.d-e).

Table 3: **Prompt Engineering.** Foreground Segmentation mIOU results on Pascal-5i when using different prompt layouts and color.

	Horizontal	Vertical
Black/White	27.17	31.57
Purple/Yellow	23.44	28.47

Table 4: **Prompt Ensembling.** We report here *color-aware mIOU*. In every line, the result is based on an ensemble of all previous prompts.

Prompt Layout	Color	Shape	Size
Horizontal	39.97	46.54	42.01
+ Vertical	41.31	54.71	46.18
+ Vertical w/ Rows Swap	**44.14**	**60.42**	**49.42**

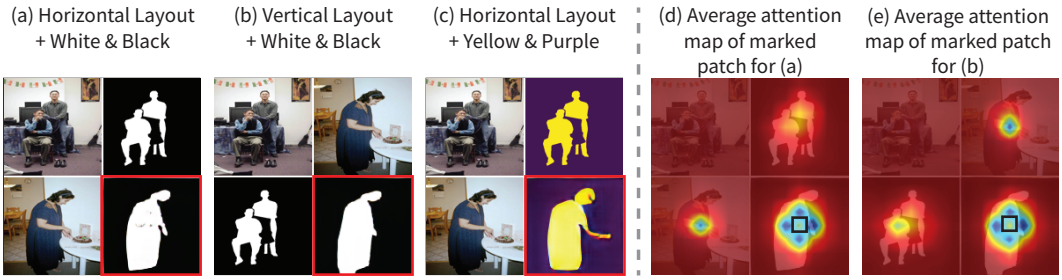

Figure 6: **Various prompt designs.** *Left*: Three different segmentation prompts for the same query, with the same support set. *Right*: The average attention maps or the selected patch (with black bounding box) for two of the prompts on the left. The highest attention values (in the non-masked patches) appear on the corresponding area in the query image for both cases.

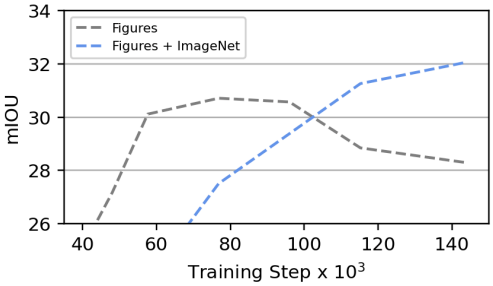

Figure 7: **Training MAE-VQGAN on more data improves visual prompting results**. Foreground Segmentation results on Pascal-5i, when trained over the Figures dataset and on the combined Figuresand ImageNet dataset.

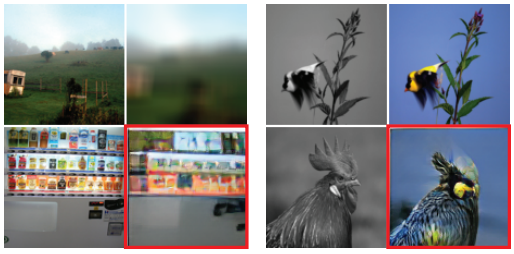

Figure 8: **Limitations.** *Left*: The MAE-VQGAN relies on a pretrained VQGAN vocabulary, which can not represent out-of-distribution blurry images. *Right*: with a single support example there is an inherit ambiguity - the task can be either colorization or style-transfer.

Prompt Ensembling. Inspired by Prompt Ensembling in NLP [27], given the same support pair and image query, we construct multiple different visual prompts (e.g, horizontal and vertical layouts, see Figure 6 a-b). We then average the completion results. The results in Table 3 on the Synthetic Study tasks demonstrate that utilizing multiple prompts can lead to improved, and more stable performance.

Dataset effect. We evaluate the effect of pretraining on a larger and more diverse dataset. We compare training on ImageNet only, Figures only, and a combination of the two. We report the mIOU results on Pascal 5i for Foreground Segmentation in Figure 7. The MAE-VQGAN trained on ImageNet achieves a consistently low 5 points mIOU. The model trained on the combined dataset performs best, which demonstrates that MAE-VQGAN can benefit from additional amounts of unlabeled images.

Limitations. For visual prompting to work, the inpainting models still require training on the Computer Vision Figures. However, our initial experiments suggest that combining it with more natural images leads to improved performance. Another limitation with the approach we propose for visual prompting is ambiguities in the task definition (See Figure 8). Ambiguities can potentially be avoided by providing multiple support examples. Lastly, to synthesize pixels we rely on the VQGAN decoder, which was pretrained on ImageNet. This leads to naturally looking images, and thus the model fails to synthesize out-of-distribution completions, as in the "image blurring task" in Figure 8.

5 Conclusion

This paper proposes an approach to visually prompt inpainting models. We show that inpainting models trained using a Masked Auto-Encoding objective on an unlabeled Computer Vision Figures dataset can be prompted to solve computer vision tasks without fine-tuning. Therefore, in Computer Vision, just like in NLP, Masked Auto-Encoders can behave as Few-Shot learners.

References

[1] Hangbo Bao, Li Dong, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[2] Amir Bar, Xin Wang, Vadim Kantorov, Colorado J Reed, Roei Herzig, Gal Chechik, Anna Rohrbach, Trevor Darrell, and Amir Globerson. Detreg: Unsupervised pretraining with region priors for object detection. *arXiv preprint arXiv:2106.04550*, 2021.

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[4] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 417–424, 2000.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[6] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.

[7] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. Maskgit: Masked generative image transformer. *arXiv preprint arXiv:2202.04200*, 2022.

[8] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pages 1691–1703. PMLR, 2020.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[10] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[11] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212, 2004.

[12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[14] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, pages 1033–1038, Corfu, Greece, September 1999.

[15] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12873–12883, 2021.

[16] Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. Learning representations by predicting bags of visual words. In *CVPR*, 2020.

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[18] Priya Goyal, Dhruv Mahajan, Abhinav Gupta, and Ishan Misra. Scaling and benchmarking self-supervised visual representation learning. In *ICCV*, 2019.

[19] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[20] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. *CoRR*, abs/2111.06377, 2021.

[21] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[22] Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 327–340, 2001.

[23] Felix Hill, Adam Santoro, David GT Barrett, Ari S Morcos, and Timothy Lillicrap. Learning to make analogies by contrasting abstract relational structure. *arXiv preprint arXiv:1902.00120*, 2019.

[24] Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438, 2020.

[25] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[26] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8420–8429, 2019.

[27] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *CoRR*, abs/2104.08691, 2021.

[28] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[29] Guilin Liu, Fitsum A Reda, Kevin J Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European conference on computer vision (ECCV)*, pages 85–100, 2018.

[30] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[31] Yongfei Liu, Xiangyi Zhang, Songyang Zhang, and Xuming He. Part-aware prototype network for few-shot semantic segmentation. In *European Conference on Computer Vision*, pages 142–158. Springer, 2020.

[32] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *arXiv preprint arXiv:2104.08786*, 2021.

[33] Roland Memisevic and Geoffrey Hinton. Unsupervised learning of image transformations. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[34] Ishan Misra and Laurens van der Maaten. Self-supervised learning of pretext-invariant representations. In *CVPR*, 2020.

[35] Khoi Nguyen and Sinisa Todorovic. Feature weighting and boosting for few-shot segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 622–631, 2019.

[36] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[37] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[38] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[39] Scott E Reed, Yi Zhang, Yuting Zhang, and Honglak Lee. Deep visual analogy-making. *Advances in neural information processing systems*, 28, 2015.

[40] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[41] Fereshteh Sadeghi, C Lawrence Zitnick, and Ali Farhadi. Visalogy: Answering visual analogy questions. *Advances in Neural Information Processing Systems*, 28, 2015.

[42] Justyna Sarzynska-Wawer, Aleksander Wawer, Aleksandra Pawlak, Julia Szymanowska, Izabela Stefaniak, Michal Jarkiewicz, and Lukasz Okruszek. Detecting formal thought disorder by deep contextualized word representations. *Psychiatry Research*, 304:114135, 2021.

[43] Amirreza Shaban, Shray Bansal, Zhen Liu, Irfan Essa, and Byron Boots. One-shot learning for semantic segmentation. *CoRR*, abs/1709.03410, 2017.

[44] Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European conference on computer vision*, pages 140–153. Springer, 2010.

[45] Zhuotao Tian, Hengshuang Zhao, Michelle Shu, Zhicheng Yang, Ruiyu Li, and Jiaya Jia. Prior guided feature enrichment network for few-shot segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[46] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep image prior. *arXiv:1711.10925*, 2017.

[47] Paul Upchurch, Noah Snavely, and Kavita Bala. From a to z: supervised transfer of style and content using deep neural network generators. *arXiv preprint arXiv:1603.02003*, 2016.

[48] Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018.

[49] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[51] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. *arXiv preprint arXiv:2003.06957*, 2020.

[52] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv preprint arXiv:2111.09886*, 2021.

[53] Boyu Yang, Chang Liu, Bohao Li, Jianbin Jiao, and Qixiang Ye. Prototype mixture models for few-shot semantic segmentation. In *European Conference on Computer Vision*, pages 763–778. Springer, 2020.

[54] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6721–6729, 2017.

[55] Jiahui Yu, Xin Li, Jing Yu Koh, Han Zhang, Ruoming Pang, James Qin, Alexander Ku, Yuanzhong Xu, Jason Baldridge, and Yonghui Wu. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

[56] Yingchen Yu, Fangneng Zhan, Rongliang Wu, Jianxiong Pan, Kaiwen Cui, Shijian Lu, Feiying Ma, Xuansong Xie, and Chunyan Miao. Diverse image inpainting with bidirectional and autoregressive transformers. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 69–78, 2021.

[57] Gengwei Zhang, Guoliang Kang, Yi Yang, and Yunchao Wei. Few-shot segmentation via cycle-consistent transformer. *Advances in Neural Information Processing Systems*, 34, 2021.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] The code and data will be released upon the acceptance of the paper.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] We will redistribute the data for research purposes.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [Yes] No offensive content was observed in the dataset.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]