جامعة الملك عبدالله للعلوم والتقنية
King Abdullah University of Science and Technology

ColorMAE
Exploring data-independent masking strategies in Masked AutoEncoders

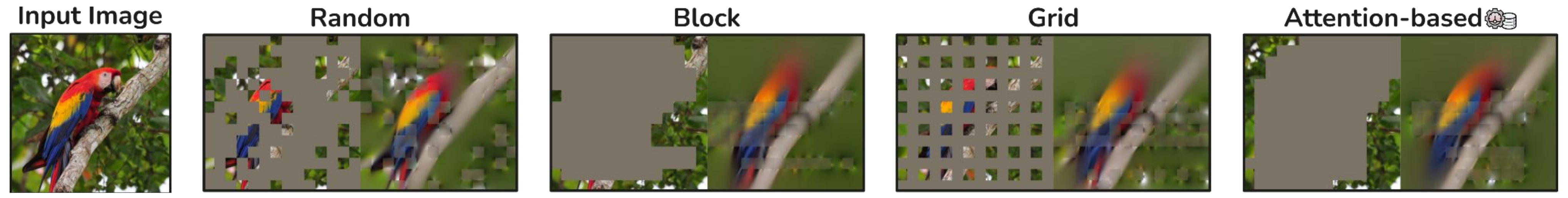
Carlos Hinojosa, Shuming Liu, Bernard Ghanem

carlos.hinojosa@kaust.edu.sa

Project Page

EUROPEAN CONFERENCE ON COMPUTER VISION
MILANO 2024

Background and Motivation

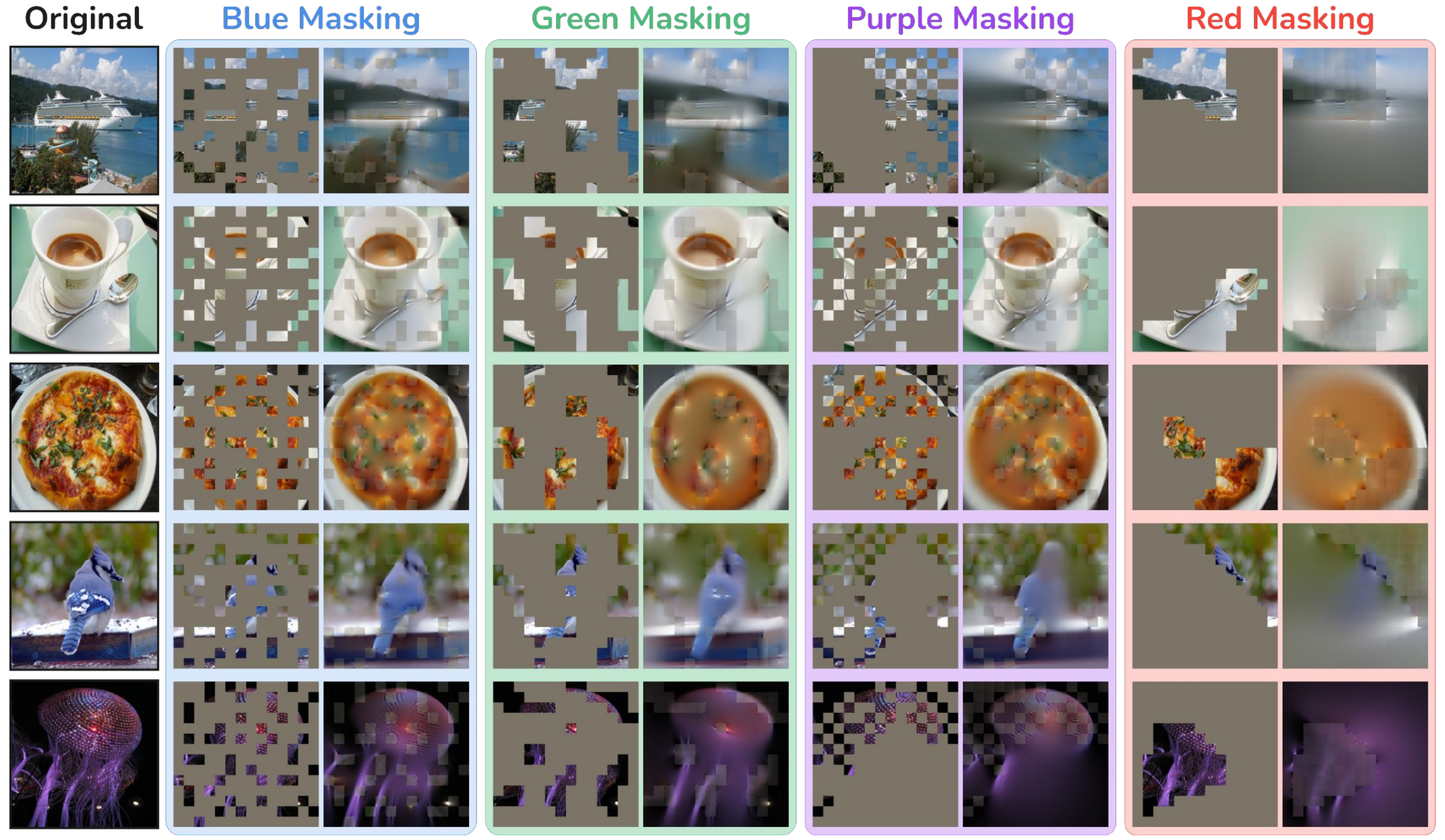


Current **Data-dependent** Masking approaches:
- ✅ Allow to extract better feature representations
- ❌ Additional computation costs

Current **Data-independent** Masking approaches:
- ❌ Lower visual representations
- ✅ No additional computation costs

Question: *Can we enhance MAE performance beyond random masking without relying on input data or incurring additional computational costs?*

Our Masking Strategies



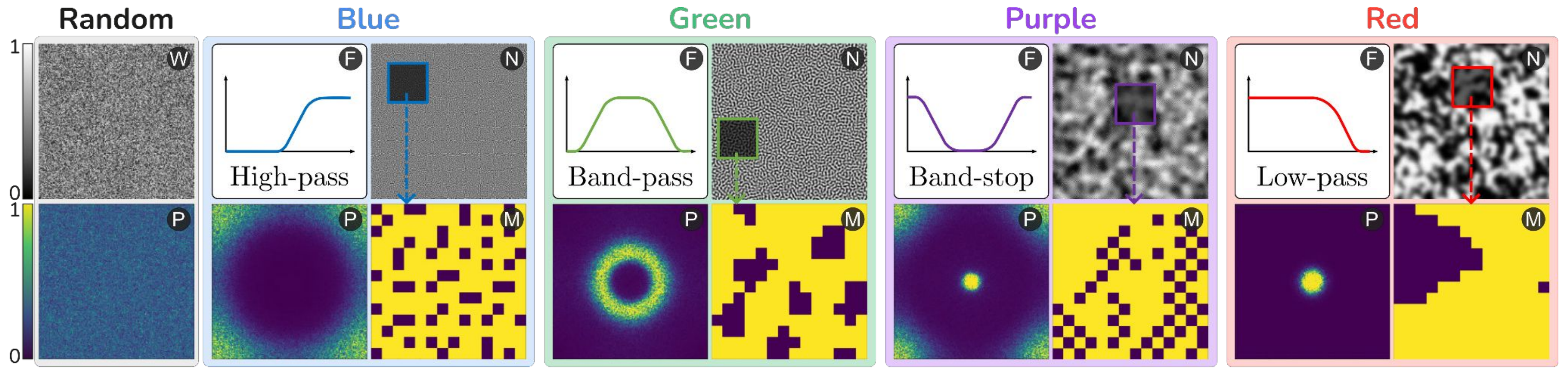
Contributions

I. We propose a simple yet effective masking strategy to generate different data independent masks by sampling and filtering random noise. Our method does not incorporate additional learnable parameters into the MAE model, preserving computational efficiency during pre-training.

II. We investigate four distinct mask types created by applying low-pass, highpass, band-pass, and band-stop filters to random noise. We offer detailed analysis and comparisons of these masks across three downstream tasks: image classification, semantic segmentation, and object detection.

III. Through extensive experiments, we demonstrate that the "Green masking" (ColorMAE-G) significantly enhances MAE performance compared to random masking.

Proposed Data-independent Masking Generation



Ⓦ Random noise Ⓕ Filter Ⓝ Filtered noise Ⓟ Periodogram Ⓜ Mask → Select highest values ■ Patch masked ■ Patch visible

In image processing, the concept of *color noise* refers to different types of noise, each characterized by a unique frequency distribution, such as predominance in the low-frequency band. Inspired by this concept, we introduce a simple yet effective data-independent method, termed ColorMAE, which generates binary mask patterns by filtering random noise. We explore four types of filters to yield mask patterns with different spatial and semantic priors. To align with traditional terminology in image processing, we categorize the produced patterns as Red, Blue, Green, and Purple noise.

Red Noise.

Let $W(x, y)$ represent a random noise image. We apply a blurring operation using a Gaussian kernel G_σ with standard deviation σ.

$$N_r = G_\sigma * W$$

Blue Noise.

To generate blue noise patterns, it is required to apply a high-pass filter over W. A practical approach involves subtracting a low-pass filtered version of W from the original noise.

$$N_b = W - G_\sigma * W$$

Green Noise.

This noise is the mid-frequency component of white noise. It can be generated by applying a band-pass filter over W, eliminating both high and low frequencies.

$$N_g = G_{\sigma_1} * W - G_{\sigma_2} * W$$

Purple Noise.

Purple noise only has high and low frequencies. To generate it, we use a band-stop filter on the noise W by first employing a band-pass filter to obtain green noise, then subtracting it from the original W.

$$N_p = W - (G_{\sigma_1} * W - G_{\sigma_2} * W)$$

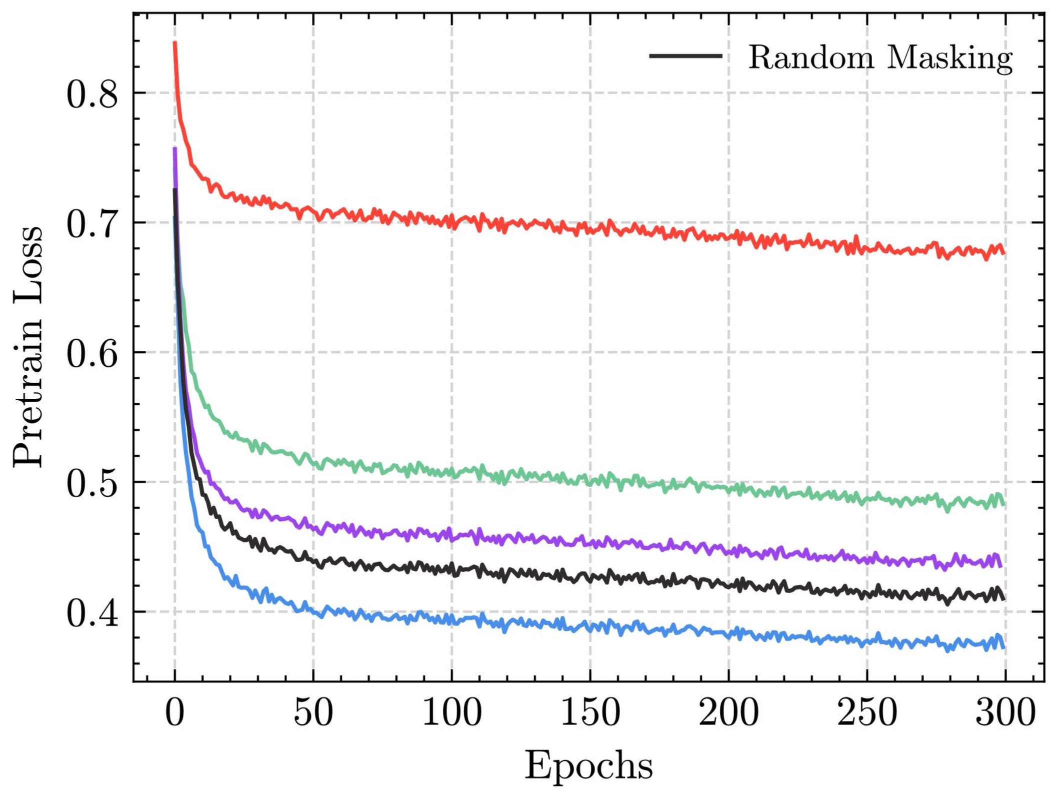
Masking Generation

We pre-compute color noises offline and store them in GPU memory before MAE pre-training. During pre-training, we apply random spatial transformations (crop, horizontal flip, vertical flip) to the loaded noise tensor, generating a P-sized noise window for each image in batch B. We then select the highest values based on the desired mask ratio. The pseudocode on the right shows our masking approach in PyTorch style.

```python
import torch

def mask_generation(N,P,B,T,mask_ratio):
    # N: noise tensor ( e.g. blue, green, purple or red noise).
    # P: total number of patches. B: batch size.
    # T: random crop, horizontal, and vertical flip PyTorch transforms.
    # mask_ratio: the mask ratio of total patches (e.g. 0.75).
    # apply random transforms (T) to get a √P × √P noise windows
    windows = T(N)[:B] # Assuming B < N.shape[0]
    len_keep = int(P * (1 - mask_ratio))
    windows = windows.view(B, -1)
    # keep stronger values from the noise
    ids_shuffle = torch.argsort(windows, dim=1, descending=True)
    ids_restore = torch.argsort(ids_shuffle, dim=1)
    ids_keep = ids_shuffle[:, :len_keep]
    # generate the binary mask: 0 is keep, 1 is remove
    mask = torch.ones([B, P])
    mask[:, :len_keep] = 0
    # unshuffle to get the binary mask
    mask = torch.gather(mask, dim=1, index=ids_restore)
    return mask, ids_restore, ids_keep
```

Pretraining and Complexity



Masking Strategy	Parameters (M)	Flops (G)	Memory (GB)	Pre-training Time per Epoch (Min)
Random	111.91	16.87	27.44	5.21
Blue	111.91	16.87	28.21	5.18
Green	111.91	16.87	28.21	5.18
Purple	111.91	16.87	28.21	5.18
Red	111.91	16.87	28.21	5.18

Pretext Task: Green masking masks out smaller random segments, making the pretext task difficult enough to learn better representations.
Complexity: *Data-adaptive* masking approaches increase the computation costs and number of parameters. Our proposed *data-independent* masking is efficient and does not add extra model parameters or computational overhead.

Experimental Results

We evaluate transfer learning performance using our pre-trained ColorMAE models on ImageNet-1K Classification, COCO Object Detection and Instance Segmentation, and ADE20k Semantic Segmentation.

Ablation Studies: Downstream tasks performance after fine-tuning

	Classification (Top-1 accuracy)					Semantic Segmentation (mIoU)					Object Detection (AP^{bbox})				
Pretrain Epochs	Random	Blue	Green	Purple	Red	Random	Blue	Green	Purple	Red	Random	Blue	Green	Purple	Red
100	81.69	**81.82**	**81.82**	80.82	78.83	42.20	40.33	**42.24**	38.22	35.31	45.90	**46.00**	45.90	44.10	40.80
300	82.82	82.56	**82.98**	82.39	81.35	44.51	43.42	**45.80**	43.85	42.08	48.50	48.10	**48.70**	47.20	45.10
800	83.17	83.02	**83.57**	82.92	82.41	46.46	44.81	**49.18**	45.96	44.78	49.15	49.10	**49.50**	48.50	46.90
1600	83.43	83.26	**83.77**	83.20	82.73	47.46	46.35	**49.26**	47.23	46.08	49.60	49.50	**50.10**	49.10	47.20

Comparison with state-of-the-art methods pre-trained on ImageNet-1K

Method	Pretrain Epoch	Pre-trained Data	ADE20K mIoU	ImageNet Top-1 Acc.	COCO AP^{bbox}	AP^{bbox}_{50}	AP^{bbox}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}
Non-MIM										
MoCo v3 ‡	600	IN1K	47.2	83.0	45.5	67.1	49.4	40.5	63.7	43.4
DINO ‡	1600	IN1K	47.2	83.3	46.8	68.6	50.9	41.5	65.3	44.5
DropPos	800	IN1K	47.8	84.2	47.7	68.3	52.8	42.6	65.3	46.2
MIM with data-adaptive masking										
AttMask	100	IN1K	45.3	-	48.8	-	-	42.0	-	-
UM-MAE	200	IN1K	42.6	82.9	45.9	64.5	50.2	-	-	-
SemMAE§	800	IN1K	44.9	83.4	45.6	66.2	55.2	40.9	63.3	44.4
HPM	800	IN1K	48.5	84.2	50.1	-	-	44.6	-	-
MIM with data-independent masking										
BEiT	800	IN1K+DALLE	45.6	83.2	40.8	59.4	44.1	36.0	56.8	38.2
MAE†	800	IN1K	46.5	83.2	49.2	69.7	53.9	43.4	66.6	46.9
MixedAE	800	IN1K	48.7	83.5	50.3	69.1	54.8	43.5	66.2	47.4
ColorMAE-G	800	IN1K	49.2	83.6	49.5	70.0	54.2	43.7	67.1	47.1
MAE	1600	IN1K	48.1	83.6	**50.6**	69.4	**55.0**	43.8	66.6	47.5
ColorMAE-G	1600	IN1K	**49.3**	**83.8**	50.1	**70.7**	54.7	**44.4**	**67.8**	**48.0**