# Masked Autoencoders Are Scalable Vision Learners

paper

Kaiming He
Facebook AI Research (FAIR)

## Overview

1. Encoder-decoder architecture
2. Encoder only operates on **visible** patches
3. Lightweidht decoder reconstructs original image from **latent representation** and **mask tokens**.
4. Maksing a high proportion of input image, i.e. **75%** yields the best performance
5. **High perframance** while **high efficiency**

## Intuition

*What makes masked autoencoding different between vision and language?*

- CNN is not straighforward to integrate **mask tokens** or **positional embeddings**.
- Information density is **low** in vision: masking a very **high** porition of image.
- Decoder is different, vision reconstruct **lower semantic** pixel, text reconstruct **rich semantic** word.

## Method

### Architecture

Encoder-decoder

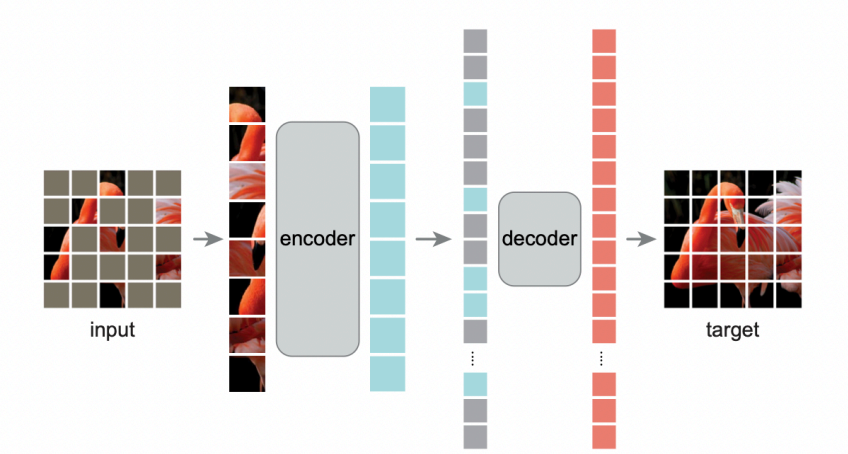


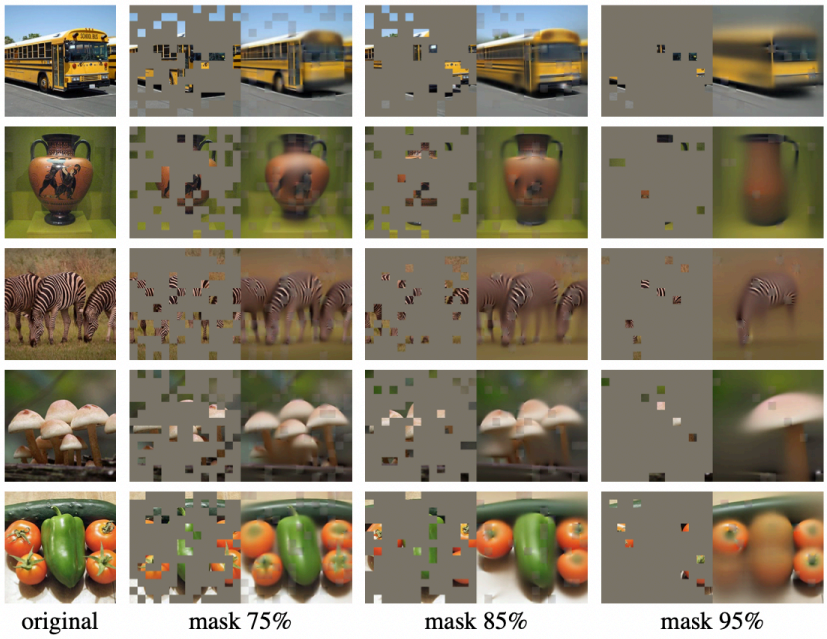

Figure 1. **Our MAE architecture**. During pre-training, a large random subset of image patches (*e.g.*, 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

## Masking

Following ViT, images are divided into non-overlapping patches. Sampling patches with **uniform distribution**.

Random sampling with a **high** ratio.

- Eliminates redundancy
- Prevents potential **center bias** (more maksed patches near the image center)
- **Efficient** encoder



### MAE encoder

**ViT** applied only on visible, unmaksed patches. **No mask tokens** are used!

We can train very **large** encoders with only **fraction of** compute and memory.

### MAE decoder

The input is

- Encoded visible pathes
- Mask tokens: shared learned vector
- add positional embeddings

Same as **ViT** structure

At last, use linear projection to reconstrcut original image $N \times L \times C \rightarrow N \times L \times p * p * 3$ reshape to $N \times 3 \times H \times W$

The MAE decoder is **only** used during pre-training, so the encoder has **<10%** computation per. token vs the encoder.

### Reconstruction Target

- Predicting **pixel** values for each masked patch
- Loss: **mean squared error (MSE)**, **only on masked patches**

### Implementation

1. Generate a token for each input patch (**linear projection** and **positional embedding**)
2. **Randomly shuffle** the tokens and **remove** the **last** portion of the list
3. After encoding, **append** masked tokens and **unshuffle** the list
4. Decoder is applied to full list

# ImageNet Experiments

## Settings

1. Pretrain by MAE unsupvisedly
2. finetune on imagenet supevisdly (updating **all** the model)
    1. or, linear probing on imagenet supevisdly (updating **only** the classifier)

Pretraining:

| config | value |
| --- | --- |
| optimizer | AdamW |
| lr | 1.5e-4 |
| batch size | 4096 |
| warmup epochs | 40 |
| Training epochs | 800 |

End to end finetune:

| config | value |
| --- | --- |
| optimizer | AdamW |
| lr | 1e-3 |
| Batch size | 1024 |
| warmup epochs | 5 |
| training epochs | 100(B), 50(L/H) |

Supervised training from scratch

| config | value |
| --- | --- |
| oprimizer | AdamW |
| lr | 1e-4 |
| Batch size | 4096 |
| Warmup epochs | 20 |
| training epochs | 300(B), 200(L/H) |

linear classifier training: linear probing

| config | value |
| --- | --- |
| optimizer | LARS |
| lr | 0.1 |
| Warmup epoch | 10 |
| Training epoch | 90 |

## Baseline: ViT Large

| scratch | MAE finetune |
|---|---|
| 82.5 | **84.9** |

## Ablation

### Masking ratio

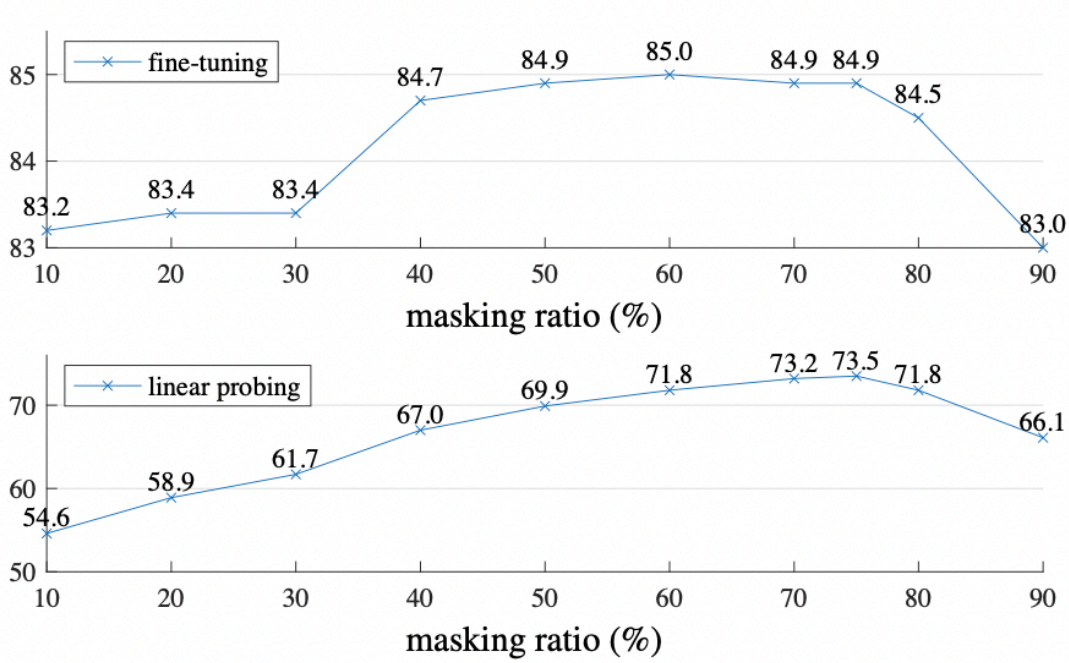



Figure 5. **Masking ratio**. A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

### Decoder

| blocks | ft | lin |
|---|---|---|
| 1 | 84.8 | 65.5 |
| 2 | **84.9** | 70.0 |
| 4 | **84.9** | 71.9 |
| 8 | **84.9** | **73.5** |
| 12 | 84.4 | 73.3 |

(a) **Decoder depth**. A deep decoder can improve linear probing accuracy.

| dim | ft | lin |
|---|---|---|
| 128 | **84.9** | 69.1 |
| 256 | 84.8 | 71.3 |
| 512 | **84.9** | **73.5** |
| 768 | 84.4 | 73.1 |
| 1024 | 84.3 | 73.1 |

(b) **Decoder width**. The decoder can be narrower than the encoder (1024-d).

"ft" means **fine-tune**, "lin" means **linear probing**. **8block+512dim**, only 9% flops of ViT large

## Mask token

Skip mask token in encoder

| case | fine-tune | linear probing | Flops |
|---|---|---|---|
| Encoder with mask | 84.2 | 59.6 | 3.3x |
| Encoder without mask | **84.9** | **73.5** | 1x |

## Reconstrcution target

1. Pixels without normalization
2. Pixels with normalization
3. MAE predict tokens (from DALLE pretrained dVAE)

| case | fine-tune | linear probing |
|---|---|---|
| pixel with norm | 84.9 | 73.5 |
| pixel without norm | **85.4** | **73.9** |
| dVAE | 85.3 | 71.6 |

## Data augmentation

Perform decently with **few** augmentation

| case | finetune | linear probing |
|---|---|---|
| none | 84.0 | 65.7 |
| crop, fixed size | 84.7 | 73.1 |
| crop, rand size | **84.9** | **73.5** |
| crop, color jit | 84.3 | 71.9 |

## Mask sampling strategy


Figure 6. **Mask sampling strategies** determine the pretext task difficulty, influencing reconstruction quality and representations (Table 1f). Here each output is from an MAE trained with the specified masking strategy. Left: random sampling (our default). Middle: block-wise sampling [2] that removes large random blocks. Right: grid-wise sampling that keeps one of every four patches. Images are from the validation set.

| case | ratio | finetune | linear probing |
|---|---|---|---|
| radom | 75 | **84.9** | **73.5** |
| block | 50 | 83.9 | 72.3 |
| block | 75 | 82.8 | 63.9 |
| grid | 75 | 84.0 | 66.0 |

## Training schedule

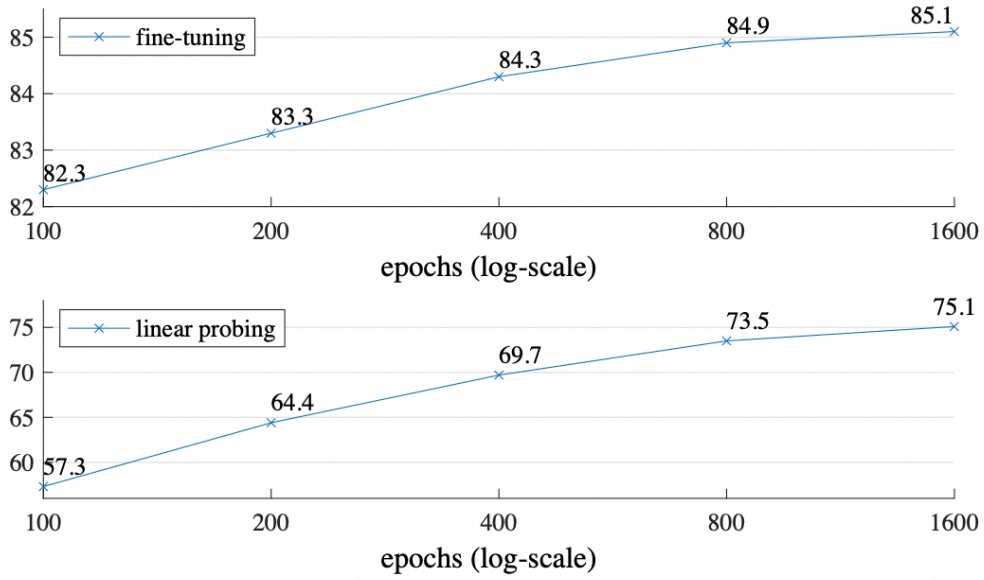



Figure 7. **Training schedules**. A longer training schedule gives a noticeable improvement. Here each point is a full training schedule. The model is ViT-L with the default setting in Table 1.

# Comparison with previous results

## Comparison with unsupervised

| method | pre-train data | ViT-B | ViT-L | ViT-H | ViT-$H_{448}$ |
|---|---|---|---|---|---|
| scratch, our impl. | - | 82.3 | 82.6 | 83.1 | - |
| DINO [5] | IN1K | 82.8 | - | - | - |
| MoCo v3 [9] | IN1K | 83.2 | 84.1 | - | - |
| BEiT [2] | IN1K+DALLE | 83.2 | 85.2 | - | - |
| MAE | IN1K | 83.6 | 85.9 | 86.9 | **87.8** |

Table 3. **Comparisons with previous results on ImageNet-1K**. The pre-training data is the ImageNet-1K training set (except the tokenizer in BEiT was pre-trained on 250M DALLE data [50]). All self-supervised methods are evaluated by end-to-end fine-tuning. The ViT models are B/16, L/16, H/14 [16]. The best for each column is underlined. All results are on an image size of 224, except for ViT-H with an extra result on 448. Here our MAE reconstructs normalized pixels and is pre-trained for 1600 epochs.

Advantage

1. Can **scale up** easily, up to $448 \times 448$
2. comparing with BEiT, MAE is more **accurate** while being **simpler and faster**.
3. total pre-training time is **less** than the other methods when trained on the same hardware.

### Comparison with supervised

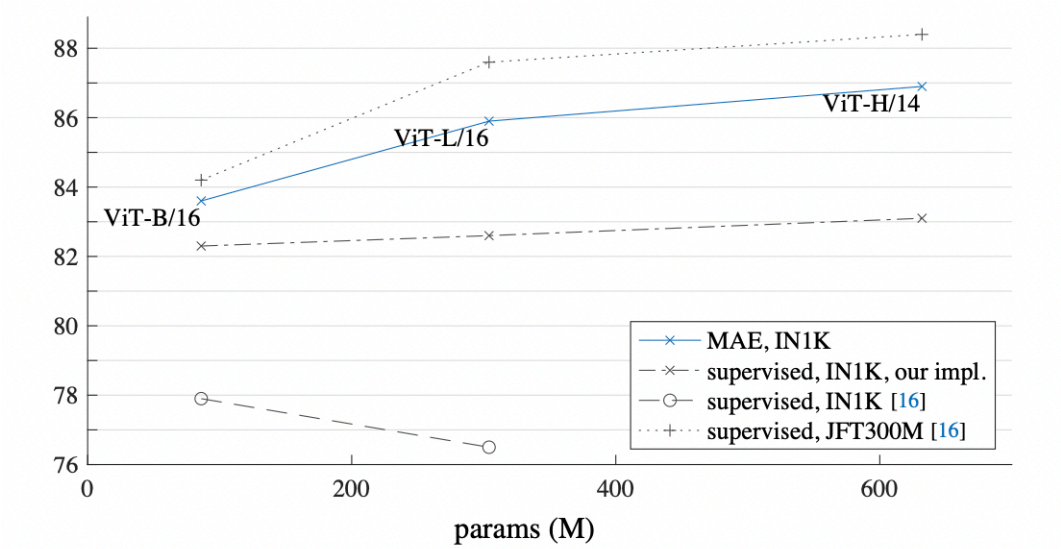



Figure 8. **MAE pre-training *vs*. supervised pre-training**, evaluated by fine-tuning in ImageNet-1K (224 size). We compare with the original ViT results [16] trained in IN1K or JFT300M.

## Partial finetuing

Linear probing misses **strong but non linear features**

Partial finetuning: finetune the **last several layers** while freezing others

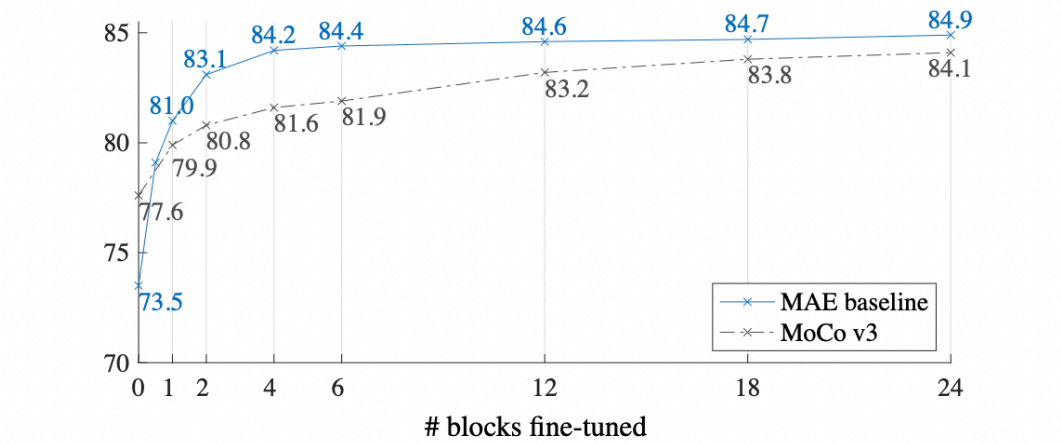



Figure 9. **Partial fine-tuning** results of ViT-L w.r.t. the number of fine-tuned Transformer blocks under the default settings from Table 1. Tuning 0 blocks is linear probing; 24 is full fine-tuning. Our MAE representations are less linearly separable, but are consistently better than MoCo v3 if one or more blocks are tuned.

## Transfer learning

### COCO

| | | $AP^{box}$ | | $AP^{mask}$ | |
|---|---|---|---|---|---|
| method | pre-train data | ViT-B | ViT-L | ViT-B | ViT-L |
| supervised | IN1K w/ labels | 47.9 | 49.3 | 42.9 | 43.9 |
| MoCo v3 | IN1K | 47.9 | 49.3 | 42.7 | 44.0 |
| BEiT | IN1K+DALLE | 49.8 | **53.3** | 44.4 | 47.1 |
| MAE | IN1K | **50.3** | **53.3** | **44.9** | **47.2** |

Table 4. **COCO object detection and segmentation** using a ViT Mask R-CNN baseline. All entries are based on our implementation. Self-supervised entries use IN1K data *without* labels. Mask AP follows a similar trend as box AP.

### Others

| method | pre-train data | ViT-B | ViT-L |
|---|---|---|---|
| supervised | IN1K w/ labels | 47.4 | 49.9 |
| MoCo v3 | IN1K | 47.3 | 49.1 |
| BEiT | IN1K+DALLE | 47.1 | 53.3 |
| MAE | IN1K | **48.1** | **53.6** |

Table 5. **ADE20K semantic segmentation** (mIoU) using UperNet. BEiT results are reproduced using the official code. Other entries are based on our implementation. Self-supervised entries use IN1K data *without* labels.

| dataset | ViT-B | ViT-L | ViT-H | ViT-$H_{448}$ | prev best |
|---|---|---|---|---|---|
| iNat 2017 | 70.5 | 75.7 | 79.3 | **83.4** | 75.4 [55] |
| iNat 2018 | 75.4 | 80.1 | 83.0 | **86.8** | 81.2 [54] |
| iNat 2019 | 80.5 | 83.4 | 85.7 | **88.3** | 84.1 [54] |
| Places205 | 63.9 | 65.8 | 65.9 | **66.8** | 66.0 [19]† |
| Places365 | 57.9 | 59.4 | 59.8 | **60.3** | 58.0 [40]‡ |

Table 6. **Transfer learning accuracy on classification datasets**, using MAE pre-trained on IN1K and then fine-tuned. We provide system-level comparisons with the previous best results.
†: pre-trained on 1 billion images. ‡: pre-trained on 3.5 billion images.

| | IN1K | | | COCO | | ADE20K | |
|---|---|---|---|---|---|---|---|
| | ViT-B | ViT-L | ViT-H | ViT-B | ViT-L | ViT-B | ViT-L |
| pixel (w/o norm) | 83.3 | 85.1 | 86.2 | 49.5 | 52.8 | 48.0 | 51.8 |
| pixel (w/ norm) | 83.6 | 85.9 | 86.9 | 50.3 | 53.3 | 48.1 | 53.6 |
| dVAE token | 83.6 | 85.7 | 86.9 | 50.3 | 53.2 | 48.1 | 53.4 |
| △ | 0.0 | -0.2 | 0.0 | 0.0 | -0.1 | 0.0 | -0.2 |

Table 7. **Pixels *vs.* tokens** as the MAE reconstruction target. △ is the difference between using dVAE tokens and using normalized pixels. The difference is statistically insignificant.

# SimMIM: a Simple Framework for Masked Image Modeling

paper

MSRA

## Difference and similarity with MAE

- Masking strategy: use $32 \times 32$, but MAE use $16 \times 16$.
- Encoder input: use masked patches but MAE only use unmasked ones.
    - therefore SimMIM can use Swin

- Deocder design: use a linear layer, but MAE use transformer.
- loss: only masked part same as MAE

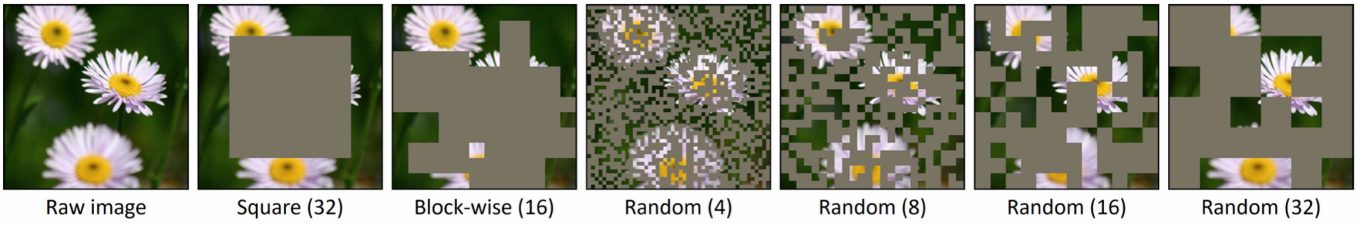



Figure 2. Illustration of masking area generated by different masking strategies using a same mask ratio of 0.6: square masking [38], block-wise masking [1] apply on 16-sized patches, and our simple random masking strategy on different patch sizes (e.g., 4, 8, 16 and 32).

# Performace

ViT-B 224 imagenet1k

| method | fine tune | linear probing |
|---|---|---|
| simMIM | **83.8** | 56.7 |
| MAE | 83.6 | **73.5** |

Same as MAE conclusion

| case | fine-tune | linear probing | Flops |
|---|---|---|---|
| Encoder with mask | 84.2 | 59.6 | 3.3x |
| Encoder without mask | **84.9** | **73.5** | 1x |

# Extension

# ConvMAE: Masked Convolution Meets Masked Autoencoders

paper

Shanghai AI lab, MMlab, SenseTime

# Overview

*Can MAE be used in ViT variants like (PVT, ConvViT) which use multicale and local inductive bias?*

- MAE in multiscale
- MAE in convolution

# Methods

# ConvMAE

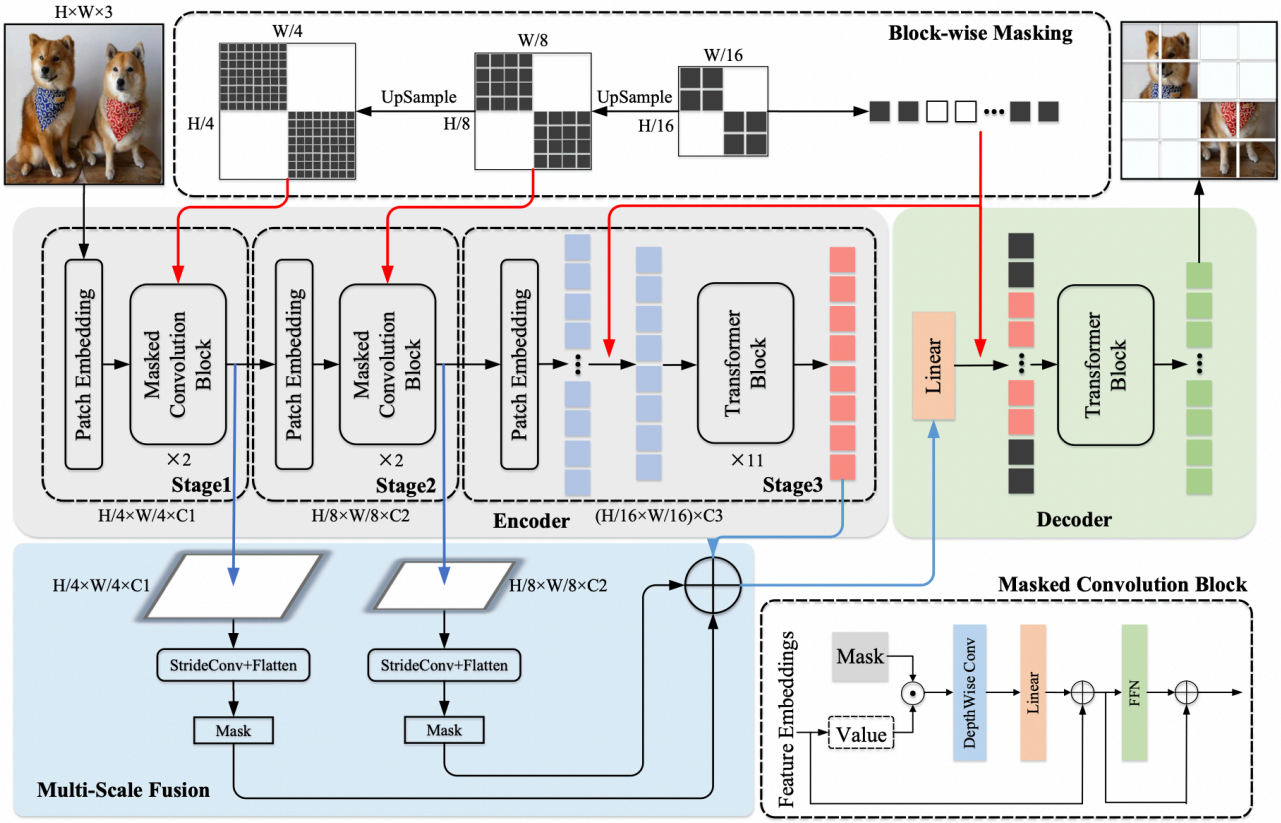



Figure 1: The pipeline of our proposed ConvMAE which consists of a hybrid convolution-transformer encoder, block-wise masking strategy with masked convolution and multi-scale decoder.

- 3 Stages: $\frac{1}{4}$ and $\frac{1}{8}$: convolution, $\frac{1}{16}$: transformer

### Block-wise Masking with Masked Convlutions

First generate the random mask 75% of **stage-3** input tokens -> upsample to stage-1 and stage-2

Use Masked Convlution Block

## Multiscale Decoder

$$E_d = Linear(StrideConv(E_1, 4) + StrideConv(E_2, 2) + E_3)$$

## ConvMAE for downstream tasks

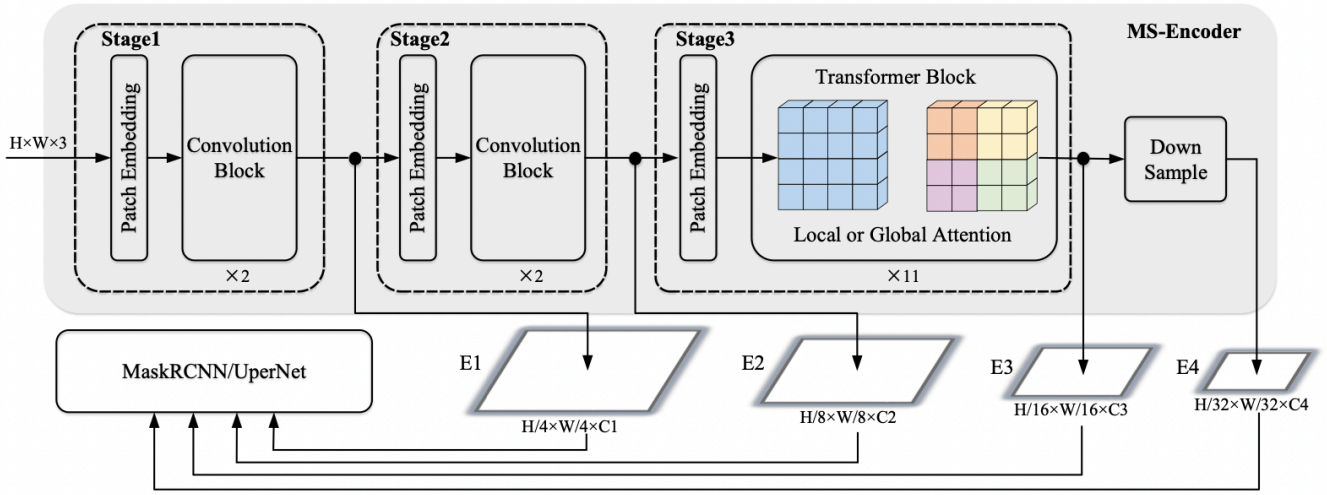



Figure 2: Overview of finetuning ConvMAE for object detection and semantic segmentation. The intermediate features of different stages serve as multi-scale inputs for an FPN [38] module.

- $E_4$ is obtained from $E_3$ with 2x2 **max pooling**
- Replace all but **1st, 4th, 7th, 11th** Global Attention with swin block

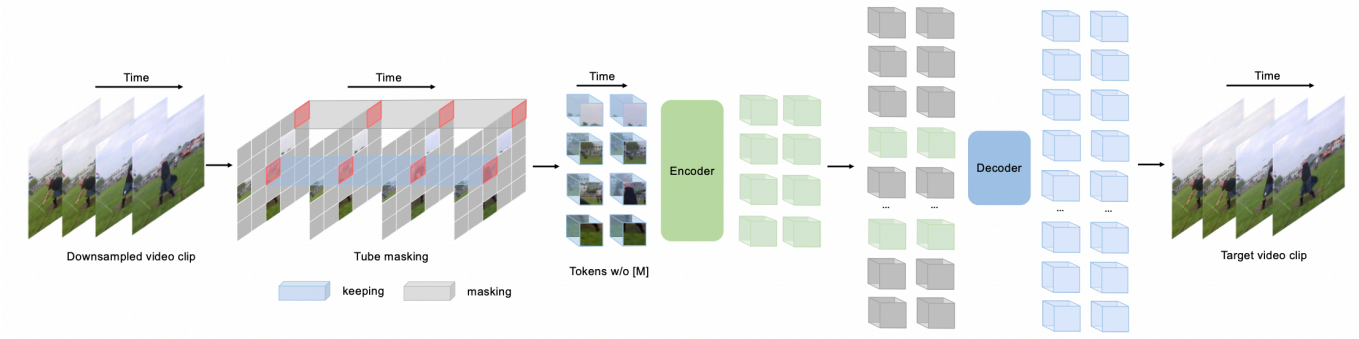
# VideoMAE: Masked Autoencoders are Data-Efficient Learners for Self-Supervised Video Pre-training

paper

## Overview

- High proportion masking ratio in video (90% to 95%)

## Methods




### Characteristics of Video Data

- Temporal redundancy
- Temporal correlation

## Temporal downsampling

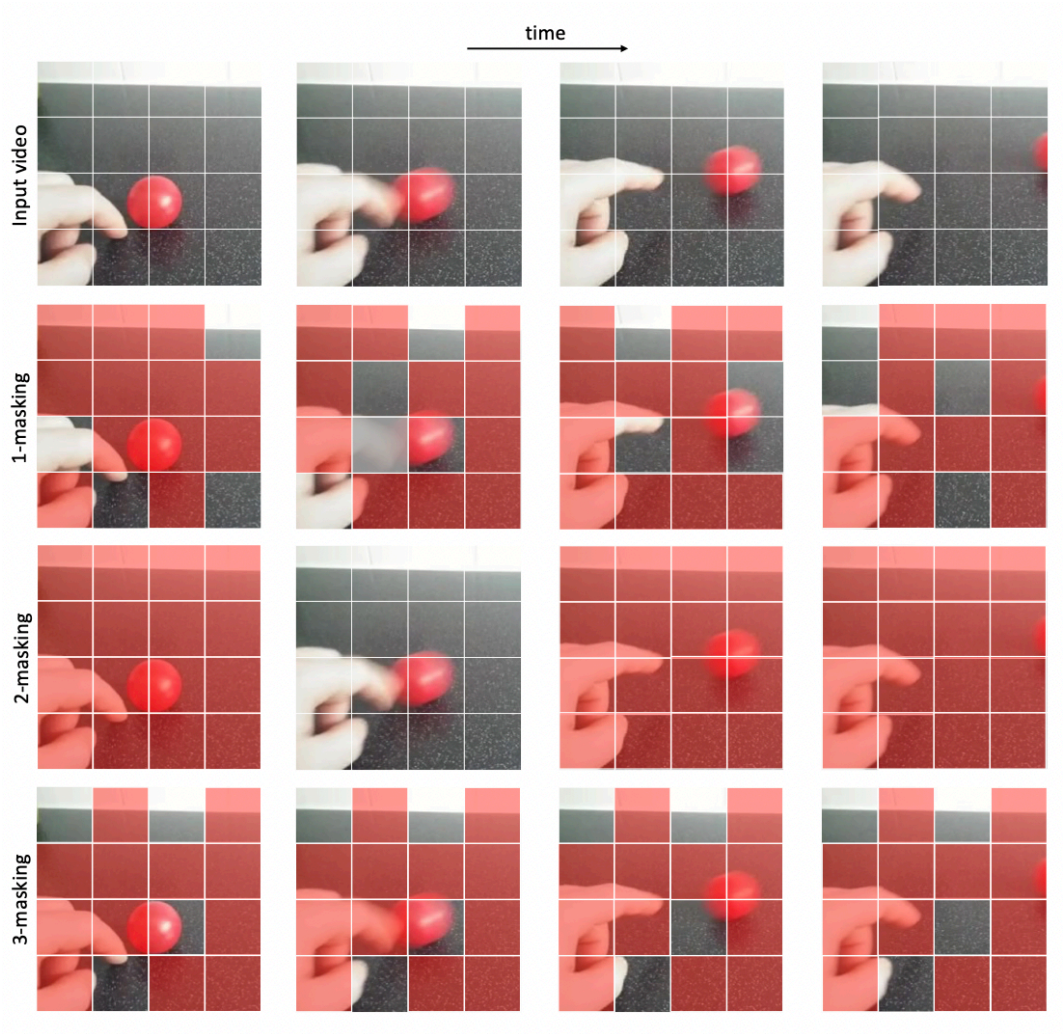
Based on **redundancy**, it sample frames with time stride $\tau = 4 \text{ or } 2$.

## Cube embedding

Take each cube of size $2 \times 16 \times 16$ as one token embedding, total $\frac{T}{2} \times \frac{H}{16} \times \frac{W}{16}$.

## Tube masking

masking map is the same for all frames




## Extremely high mask ratio

VideoMAE is in favor of **extremely high** masking ratios (90%-95%)