

Hyperparameter	Value
<i>MAE Pretraining</i>	
optimizer	AdamW [38]
base learning rate	1e-4
weight decay	0.05
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.95$
batch size	4096
learning rate schedule	cosine decay
total batches or iterations	249600
warmup iterations	$1/8 \times$ total iterations
augmentation	RandomResizedCrop
#GPU	64 V100 (32 gb)
Wall-clock time	~ 36 hours
<i>Encoder ViT Architecture</i>	
#layers	12
#MHSA heads	12
hidden dim	768
class token	yes
positional encoding	sin cos
<i>Decoder ViT Architecture</i>	
#layers	8
#MHSA heads	16
hidden dim	512
class token used	yes
positional encoding	sin cos

Table 3: Training and architectural hyperparameters for MAE pretraining.

444 A MAE Hyperparameters

445 We list key hyperparameters for the MAE training loop in Table 3. Note that these parameters
446 were employed directly from original MAE paper [8] and are actually shared by relevant robotics
447 baselines [13, 18]. Consistent with the terminology in [8], the employed learning rate is the base
448 learning rate scaled by (total batch size / 256). For a head-on comparison with prior work [8,
449 13], we train the ViT for iterations equivalent of 800 epochs over ImageNet dataset. This rigorous
450 benchmarking took # GPUs \times wall clock time \times # data ablations = $64 \times 1.5 \times 12 = 1152$ GPU days.

451 B BC Hyperparameters

452 The following section describes the hyperparameters used in our behavior cloning loop. As dis-
453 cussed in Sec. 3, the BC policy begins by taking in the image and passing it through the pre-trained
454 encoder to get a representation, $E(i_t)$. That representation is then concatenated to the joint in-
455 formation to get a policy input, $x_t = [E(i_t), j_t]$. The policy input is fed through a 2-layer mlp
456 network, with a batchnorm preceding the first layer, ReLU activations [3], and hidden dimensions
457 of [512, 512]. Additionally, we add dropout [19] to the two mlp layers w/ probability $p = 0.2$ after
458 the ReLU activations. The result of the top layer is then passed to 2 linear layers, that predict the
459 mean (μ), mixing parameters (ϕ), and standard deviation (σ) of a Gaussian Mixture Model (GMM)
460 distribution w/ m modes:

$$p(x) = \sum_{i=1}^m \phi_i N(x | \mu_i, \sigma_i)$$

461 The choice of GMM was based on prior work [36, 37] that showed it could dramatically improve
462 performance. After some tuning, we used $m = 5$ on the RoboSuite tasks (note their benchmark [36]
463 used $m = 5$) and the real world tasks, since it worked best. However, for Franka Kitchen and
464 MetaWorld, we found no significant difference. As a result, we used $m = 1$ (i.e. standard Gaussian
465 distribution) for those tasks to maximize comparability with prior benchmarks [13, 40].

466 The policy was optimized for 50000 iterations using the ADAM optimizer [38], with a learning rate
467 of 0.0001 and a L2 weight decay of 0.0001. In addition, we applied data augmentation (random
468 crops and random blur) to the input image i_t , before passing it E . This was based on recommenda-
469 tions for best practices from Hansen et. al. [42]. The full code for this setup is open-sourced on our
470 website: <https://sites.google.com/view/robotics-datasets-analysis>.

471 C Task Hyperparameters

472 This section describes the hyperparameters made while setting up both sim and real world tasks. All
473 code (for robot/sim environments and BC training) is open sourced.

474 **Simulation** The simulation tasks were taken from standard benchmarks (MetaWord [39], Franka
475 Kitchen [40], RoboSuite [41]) in the robotics field. The training demonstrations were collected by
476 previous work (CortexBench [13], Relay Policy Learning [40], RoboMimic [36] respectively), and
477 were directly used in our tasks. We fine-tune on $n = 25$ demos for MetaWorld/Franka Kitchen, and
478 $n = 200$ demos on RoboSuite (again to stay consistent with older papers). Task success is measured
479 by the environments themselves, and we get numbers by estimating success rates empirically using
480 50 test trajectories. Note that we only evaluate the policy at the end of training (unlike some prior
481 work that evaluated multiple times over the course of training). This was done to ensure the sim
482 evaluation setup matched the real world (i.e. we can't evaluate real policies multiple times during
483 training).

484 **Real World** As discussed in Sec. 3, our real world tasks were built using a Franka Panda robot,
485 and we collected 50 demonstrations for each task using a VR tele-op setup. We heavily encourage
486 the reader to get a feel for the training data and tasks by viewing the supplemental video on our
487 website: <https://sites.google.com/view/robotics-datasets-analysis>.

488 The following section expands on our real world task descriptions from Sec. 3, and provides some
489 additional details:

- 490 • **Block stacking** requires the robot to pick up the red block and place it on the green block.
491 This is the simplest task, since the robot only has to adapt to new object configurations
492 during test time, but it still requires the robot to precisely localize and grasp the (small) red
493 block.
494 We evaluated agents on this task using 25 test positions for the red/green block. These test
495 positions were kept fixed for all policies to ensure maximum reproducibility.
- 496 • **Pouring** requires the robot to lift the cup and pour almonds in the target bowl. During
497 test time the cup and target bowls are both novel objects (unseen during training), and are
498 placed in random locations. Thus, this task forces the robot to generalize to new visual
499 inputs.
500 We evaluated 3 separate cup/target bowl pairs in 5 positions each (so 15 trials total). Note
501 that none of these objects or positions were seen during test time. Again, the object and
502 position combinations were kept fixed across every model tested.
- 503 • **Toasting** is the final task, and it requires the robot to pick up the object, place it in the
504 toaster, and then shut the toaster. During test time, we use a novel object and randomize
505 both the object's initial pose and the toaster's initial orientation. This is the most difficult
506 task, since it requires the robot to execute a multi-stage manipulation strategy, while also
507 generalizing to new visual scenarios.
508 We evaluated 2 target objects pairs and randomized the toaster orientation into 5 separate
509 poses (so 10 trials total). Note that none of these objects or toaster orientations were seen
510 during test time. As before, all the test conditions were shared across all policies.