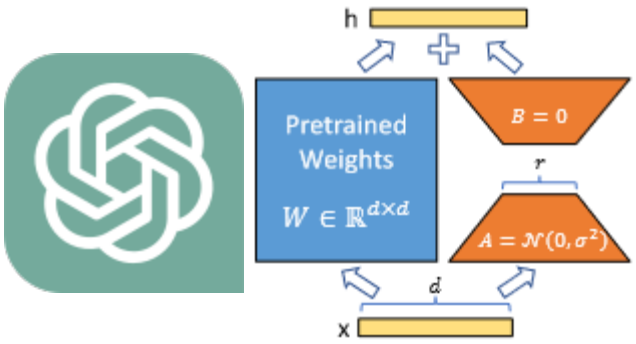
LoRA Explained by ChatGPT



Setting	Options	Description	Drawbacks
base_model	String path to base model file	This option specifies the path to the base model file, which is used as a starting point for training the new model. The model will be fine-tuned on the new data provided.	It is important to select a base model that is relevant to the task and data, otherwise the fine-tuning process may not result in improved performance.
img_folder	String path to folder containing training images	This option specifies the path to the folder containing the training images used for training the model.	The quality and quantity of the training images can greatly affect the performance of the model. It is important to have enough diverse, high-quality images for the model to learn from.
output_folder	String path to folder for saving the output model	This option specifies the path to the folder where the output model will be saved after training.	The output folder should have enough storage space to save the model files. It is important to regularly backup the model files to prevent loss of work in case of data loss.
change_output_name	Boolean	Specifies whether to change the output name of the trained model. If set to True, the output name will be changed, otherwise it will remain the same.	N/A
save_json_folder	String	The path to the folder where the json files of the trained model will be saved.	N/A
load_json_path	String	The path to the json file that will be used to load the parameters of the model. This is useful for continuing training from a previous checkpoint.	The file specified by the load_json_path must exist and be a valid json file.
json_load_skip_list	list of strings	A list of strings specifying keys in the model config to not load from a saved .json file.	If a required key is accidentally included in the json_load_skip_list, the model may not function as intended.
multi_run_folder	string	The name of the folder to store the results of multiple training runs, in the format {multi_run_folder}/run_{run_number}/.	If multiple runs are performed without changing the multi_run_folder name, the previous results will be overwritten.
save_json_only	boolean	If set to True, only the model config file (in .json format) will be saved, instead of the full model checkpoint.	The model cannot be resumed from a .json file alone and must be trained from scratch if save_json_only is set to True.
caption_dropout_rate	float (0 to 1)	Specifies the rate of random dropout for captions during training	If this value is set too high, important information from the

		captions during training.	important information from the captions may be lost, leading to lower quality results.
caption_dropout_every_n_epochs	int	Specifies how often (in terms of number of epochs) to perform caption dropout during training.	If this value is set too high, the model may not be exposed to enough diversity in the captions during training, leading to overfitting. If it is set too low, the model may not have enough time to learn from the captions before they are dropped out.
caption_tag_dropout_rate	float (0 to 1)	Controls the rate of random dropout of the captions when training the model. A high value means more captions will be dropped, while a low value means fewer captions will be dropped.	Setting a high value could result in loss of important information in the captions, leading to lower quality results.
net_dim	integer	This setting determines the number of hidden units in the model's network architecture. A larger net_dim value results in a more complex model with greater capacity, but also requires more computational resources and can lead to overfitting if the model is too large for the amount of data it is trained on.	Overfitting, increased computational resources
alpha	float	This setting determines the learning rate used during training. A larger alpha value results in faster convergence, but can also cause the model to overshoot the optimal solution or converge to a suboptimal solution if set too high. A smaller alpha value can result in slow convergence or failure to converge at all if set too low.	Suboptimal solutions, slow convergence
scheduler	string	This setting determines the learning rate schedule to use during training. Common choices include "step", "cosine", and "plateau". A step scheduler decreases the learning rate by a fixed factor after a specified number of iterations, while a cosine scheduler decreases the learning rate in a cosine function. A plateau scheduler reduces the learning rate when the validation loss stops improving.	Suboptimal solutions, slow convergence, difficulty in choosing appropriate schedule
cosine_restarts	integer	The number of times the cosine annealing schedule should restart. A higher number of restarts allows the learning rate to change more frequently and reduces the risk of getting stuck in a suboptimal learning rate.	Increasing the number of restarts can lead to more frequent changes in the learning rate, which can make the training process more unstable and harder to tune.
scheduler_power	float	The power parameter for the scheduler. A larger power value means that the learning rate changes more slowly.	Setting a high power value can result in a slow learning rate that fails to converge in a reasonable amount of time. On the other hand, setting a low power value can result in an overly aggressive learning rate that causes the model to overfit to the training data.
warmup_lr_ratio	float	The ratio of the maximum learning rate to the initial learning rate during the warm-up period. The learning rate gradually increases from the	A high warm-up learning rate ratio can cause the model to converge slowly or not

		initial value to the maximum value during this period.	converge at all. On the other hand, a low warm-up learning rate ratio can result in a learning rate that is too low to effectively train the model.
learning_rate	float	This option sets the learning rate for the optimizer used to train the model. It determines the step size at which the optimizer updates the model parameters. The default value is 0.0001.	A high learning rate can cause the model to converge too quickly to a suboptimal solution, while a low learning rate can cause the training process to be slow and potentially converge to a poor solution. The learning rate must be set carefully to balance these trade-offs.
text_encoder_lr	float	This option sets the learning rate specifically for the text encoder component of the model. If this value is set to a value different than the learning_rate, it allows for fine-tuning the text encoder specifically.	Setting the text_encoder_lr to a value different than the learning_rate can result in overfitting to the text encoder, which may not generalize well to new data.
unet_lr	float	This option sets the learning rate specifically for the UNet component of the model. If this value is set to a value different than the learning_rate, it allows for fine-tuning the UNet specifically.	Setting the unet_lr to a value different than the learning_rate can result in overfitting to the UNet, which may not generalize well to new data.
num_workers	Integer	Specifies the number of worker threads to load data. Increasing the number of workers can speed up data loading and training, but may also increase memory usage.	Too many workers may cause memory overflow and slow down the training process.
persistent_workers	Boolean	Determines whether to use persistent worker threads. Persistent worker threads maintain a queue of data samples, allowing for more efficient data loading.	Can result in decreased performance on systems with limited resources such as memory or disk I/O.
batch_size	Integer	Specifies the number of samples to include in each batch. Larger batch sizes can lead to more efficient training, but may also increase memory usage and slow down convergence.	Too large of a batch size can cause memory overflow and slow down the training process, while too small of a batch size can result in slow convergence.
num_epochs	integer	Specifies the number of complete passes through the training data that should be made. A higher number of epochs will result in a more accurate model, but will also take more time to run.	Longer training time, may over-fit the data if too many epochs are used.
save_every_n_epochs	integer	Specifies how frequently the model should be saved during training. For example, setting this to 5 means the model will be saved every 5 epochs.	Takes up more storage space, as the model will be saved more frequently.
shuffle_captions	Boolean	Specifies whether the training data should be shuffled between epochs. Shuffling can help prevent the model from getting stuck in a local minimum, but can also make training less consistent.	May result in less consistent training if the order of the training data has important implications.
keep_tokens	Integer	The number of the most frequent tokens to keep in the text corpus for training. Tokens that occur less frequently than keep_tokens are replaced with an unknown token ("<unk>"). A	May result in information loss if keep_tokens is set too low.

		smaller value will result in a smaller vocabulary size, potentially reducing the memory requirements of the model, but may also result in information loss.	
max_steps	Integer	The maximum number of steps to take during training. Training will stop once the model has seen max_steps batches of data.	If max_steps is set too low, the model may not be fully trained. If set too high, training may take a long time.
tag_occurrence_txt_file	String	The path to a text file that contains the tag occurrence information. The tag occurrence information is used to weight the loss function during training.	If the tag occurrence information is not available or is not correctly specified, the model may not be trained correctly.
sort_tag_occurrence_alphabetically	True or False	If set to True, the tags in the tag_occurrence_txt_file will be sorted alphabetically. This option can be useful for maintaining consistency in the tag order and ensuring that similar tags are grouped together.	N/A
train_resolution	Integer value	This value determines the resolution of the training images. A higher resolution will result in more detailed images, but will also require more memory and computational resources.	Increasing the resolution can significantly increase the training time and memory requirements, especially if the training data is large.
min_bucket_resolution	Integer value	This value determines the minimum size of the bucket used for training. A smaller bucket size can result in a faster training process, but may also lead to overfitting or lower quality results.	Decreasing the bucket size too much may result in a reduced training efficiency and lower quality results.
max_bucket_resolution	Integer	Specifies the maximum image resolution of the training data. The training data will be downsampled if its resolution is larger than max_bucket_resolution.	A high value for max_bucket_resolution can lead to longer training times and higher memory usage, while a low value may reduce the quality of the generated images.
lora_model_for_resume	String	Specifies the path to a pre-trained LoRA model that will be used to resume training from a previous checkpoint.	Resuming training from a pre-trained model may lead to overfitting if the new training data is significantly different from the original training data.
save_state	Boolean	Specifies whether the training state should be saved after each epoch. If set to True, the training state will be saved in the lora_model_for_resume file.	A high frequency of saving the training state can lead to longer training times and higher disk usage.
load_previous_save_state	True or False	Specifies whether to load the previous saved state of the model during training. If set to True, the training will resume from the previous saved state. If set to False, the training will start from scratch.	If the previous saved state is not available or is corrupted, training will not be able to resume and the training will start from scratch, which could result in longer training time and decreased performance.
training_comment	String	Specifies a comment that will be added to the name of the saved model. This can be used to distinguish between different models that were trained with different settings or parameters.	None
unet_only	True or False	Specifies whether to train only the UNet	Training only the UNet

		component of the model. If set to True, only the UNet component of the model will be trained, and the text encoder component will not be trained. If set to False, both the UNet and the text encoder components of the model will be trained.	component of the model could result in lower performance compared to training both components, as the text encoder component is an important component of the model that helps to encode text information into the training process.
text_only	True or False	Determines if the model should be trained only on text or on both text and images. Setting this to True will result in faster training but lower quality image generation. Setting this to False will result in slower training but higher quality image generation.	If set to True, the generated images will not be as accurate or detailed as when set to False.
reg_img_folder	String	The path to the directory containing the images to be used for training.	This option is only relevant if text_only is set to False. If no images are provided, the model will only be trained on text, and will not be able to generate images.
clip_skip	True or False	Determines if the model should skip over clipped images in the training data. Clipped images are those that are either too small or too large in dimension compared to the train_resolution.	If set to True, the model may not be able to learn from certain images in the training data. If set to False, the training may take longer, as the model will need to process all images, even those that are clipped.
test_seed	integer	Specifies the random seed for test data generation and evaluation. Setting the seed ensures that the same test data is generated each time the script is run.	A different seed may result in different test data and evaluation results, making it difficult to compare performance across runs.
prior_loss_weight	float	Specifies the weight of the prior loss term in the overall loss calculation. The prior loss term is used to encourage the model to generate outputs that are similar to the prior distribution of the training data.	Setting the weight too high may result in outputs that are too similar to the prior, reducing the creativity of the model. Setting the weight too low may result in outputs that are far from the prior and less coherent.
gradient_checkpointing	boolean	Specifies whether to use gradient checkpointing to reduce memory usage during training. Gradient checkpointing involves selectively saving and reloading activations during backpropagation, which reduces memory usage at the cost of increased computation time.	Using gradient checkpointing may slow down the training process and may not be necessary for small models or for devices with sufficient memory.
gradient_acc_steps	Integer	Specifies the number of gradients accumulation steps during training. Increasing this value can reduce memory usage and help with the training stability.	A higher value of gradient_acc_steps increases the number of operations and may slow down the training process.
mixed_precision	Boolean	Specifies whether to use mixed precision training, which trains the model using lower-precision data types to speed up training.	Using mixed precision training may result in lower accuracy and may cause training instability.
save_precision	Float	Specifies the precision to use when saving the	Lower precision values may

save_precision	Float	Specifies the precision to use when saving the model's weights. It is typically set to 32 or 16 depending on the precision used during training.	Lower precision values may cause loss of information when saving the model's weights, leading to lower accuracy.
save_as	str	Specifies the file format to save the trained model in. Supported formats are: ckpt, safetensors, pt, bin.	The file format should match the type of Stable Diffusion AI art model that the LoRA model will be used with.
caption_extension	str	Specifies the extension of the text file that contains captions for the training data.	The extension must match the actual file extension of the caption file.
max_clip_token_length	int	Specifies the maximum number of tokens allowed in a single caption. Captions that exceed this length will be skipped during training.	Setting a high value may increase the memory usage during training. Setting a low value may result in the loss of important information in the captions.
buckets	List of integers	Specifies the size of the bucketing algorithm. For example, if buckets are set to [5,10,15], the data will be separated into three buckets, with data having length of 5 tokens in one bucket, data having length of 10 tokens in another bucket, and data having length of 15 tokens in the third bucket.	The number of buckets and the size of the buckets must be carefully selected to achieve good results. If the number of buckets is too small, the model may under-perform, while if the number of buckets is too large, the model may over-fit.
xformers	List of strings	Specifies the transformers to use during training. The transformers can be used to apply data augmentation techniques such as random cropping, flipping, rotation, etc.	The choice of transformers can greatly impact the performance of the model, so it is important to carefully select the transformers that are most appropriate for the specific task.
use_8bit_adam	Boolean	Specifies whether to use 8-bit Adam optimizer. This option can be used to reduce the memory requirements of the training process.	If this option is set to True, the memory requirements of the training process will be reduced, but the training may be slower and the model may have lower accuracy.
cache_latents	Boolean	If set to True, the latents of training data will be cached for faster training. This can reduce the time to train the model, but it may also use more memory and increase the time to start training.	Increased memory usage and slower start-up time.
color_aug	Boolean	If set to True, color augmentation will be performed during training. This can increase the diversity of the training data, but may also slow down training.	Slower training time.
flip_aug	Boolean	If set to True, flipping augmentation will be performed during training. This can increase the diversity of the training data, but may also slow down training.	Slower training time.
random_crop	True/False	Specifies whether to apply random cropping to the training images. If set to True, the training images will be randomly cropped to a specified size before being fed into the model.	Using random cropping can increase the diversity of the training data, but it also increases the computational cost of training and may slow down the training process.

vae	True/False	Specifies whether to use a Variational Autoencoder (VAE) as the backbone of the model. If set to True, the model will be trained as a VAE.	Using a VAE can provide a more flexible representation of the data, but it can also increase the difficulty of training and may require more fine-tuning.
no_meta	True/False	Specifies whether to exclude meta data (e.g., class labels, attributes, etc.) from the training process. If set to True, the model will not have access to any meta data during training.	Excluding meta data can simplify the training process, but it may result in a lower-quality model that is not able to make use of the additional information provided by the meta data.
log_dir	String	Path to the directory where log files for the training will be stored	If the directory already exists and is not empty, the training may overwrite previous logs stored in that directory, leading to data loss.
bucket_reso_steps	Integer	Number of steps for increasing the resolution of the image. The image resolution starts from the max_bucket_resolution and increases by a factor of 2 after each step.	Setting this value too high can lead to memory errors and longer training times, as the size of the images increases at each step. Setting this value too low can result in low-quality images.
bucket_no_upscale	Boolean	Indicates whether to restrict the image resolution from increasing beyond its original size.	If set to True, the image resolution will not increase beyond its original size, which may result in lower-quality images.
v2	True or False	This setting specifies whether to use version 2 of the model architecture or not.	Using a different version of the model architecture may change the quality and performance of the generated art, so it's important to experiment and compare the results to determine the best option for a given task.
v_parameterization	"spectral_norm", "instance_norm", or "batch_norm"	This setting determines how the model's parameters are regularized during training. Spectral normalization, instance normalization, and batch normalization are different methods for preventing overfitting, but each has its own trade-offs in terms of computational cost and performance.	Choosing the wrong regularization method can negatively impact the model's performance, so it's important to experiment with different options to determine which works best for a given task.