GraspLDM

Algorithm-1

Overview

Trained from scratch instead of using pre-trained checkpoints, which made notable changes in the plots.

Evaluation part was finished on testing data (newly trained model was used)

Modifications

In GraspLDM implemented code Good/bad grasp prediction is handled by the class_logits output (the "confidence" head), which is trained with a classification loss to predict grasp success.

However, Qualities are additional outputs (e.g., epsilon, force closure, etc.) that the model can predict if num_output_qualities > 0. These are trained with a separate quality loss and are not used by default to decide if a grasp is good or bad.

Therefore, with quality metrics, we can set custom thresholds (e.g., epsilon > 0.2) to define what counts as a "good" grasp for our application, rather than relying only on binary success labels. And the model is trained not only with the classification loss (for predicting success/failure, i.e., object_in_gripper) ,but also with the quality loss (for predicting the physical quality metrics, e.g., epsilon, force closure, etc.).

Finally, during our evaluation, we can compare both the predicted success and the predicted qualities to our ground truths, giving you a much richer and more interpretable assessment of grasp quality.

Including qualities means our model is supervised by both binary and continuous (or multi-class) grasp metrics, making it more robust and informative for real-world grasping tasks.

Parameters of qualities

1. object_in_gripper
    - Type: Integer (0 or 1)
    - Meaning: Indicates whether the grasp was successful in simulation.
        - 1: The object was successfully grasped (good grasp).
        - 0: The object was not grasped (bad grasp).
    - Use: This is the ground-truth label for binary classification

2. epsilon
    - Type: Float (usually between 0 and 1, but can be higher)
    - Meaning: The epsilon metric (Ferrari-Canny metric) measures the smallest wrench (force/torque) that cannot be resisted by the grasp.
        - Higher values mean the grasp is more robust to external disturbances.
        - Physical interpretation: The minimum force (in any direction) required to break the grasp.
    - Use: Used as a continuous quality score for grasp robustness.

Parameters of qualities

3. force_closure
   - Type: Integer (0 or 1)
   - Meaning: Indicates whether the grasp achieves force closure.
     - 1: The grasp can resist arbitrary external forces/torques (force closure).
     - 0: The grasp cannot resist all disturbances.
   - Use: Binary quality metric; often used as a threshold for "physically valid" grasps.

4. Volume
   - Type: Float
   - Meaning: The volume of the grasp wrench space.
     - Measures the set of wrenches (forces/torques) that the grasp can resist.
     - Higher values mean the grasp can resist a wider range of disturbances.
   - Use: Continuous quality metric for grasp versatility.

Parameters of qualities

5. Distance_to_center
   - Type: Float
   - Meaning: The Euclidean distance from the grasp contact point(s) to the object's center of mass.
     - Lower values are generally better, as grasps closer to the center are more stable.
   - Use: Used to penalize grasps that are far from the object's center.

These matrices are valuable for real-life world applications. The above are mentioned for reference if wanted. Currently our DNJPs script cannot simulate the interactions between robot and the object, so we don't use any of these metrices for classification.

Evaluation

generate_grasps.py only generates grasp for one random sample from the dataset and (optionally) visualizes them.

There is no calculation or printing of accuracy, success rate, or any quantitative evaluation of the generated grasps against ground truth.

However, the same file can be modified so that instead of visulasing an random grasp we can loop it over our testing data and calculate the accuracy.

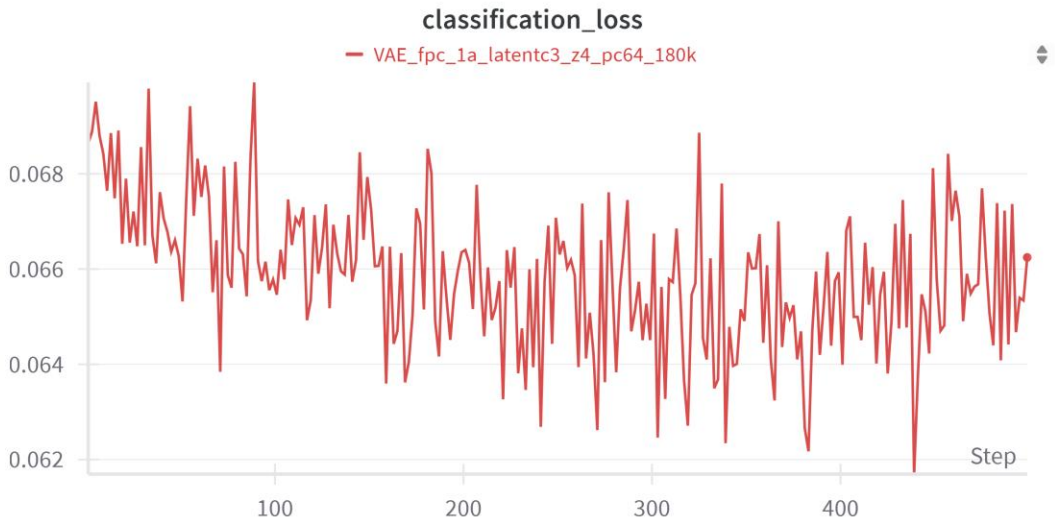
Initially the main function returns a dict **results** of the randomly generated grasps

results keys: ['grasps', 'grasp_tmrp', 'confidence', 'qualities', 'pc', 'inputs']

Evaluation

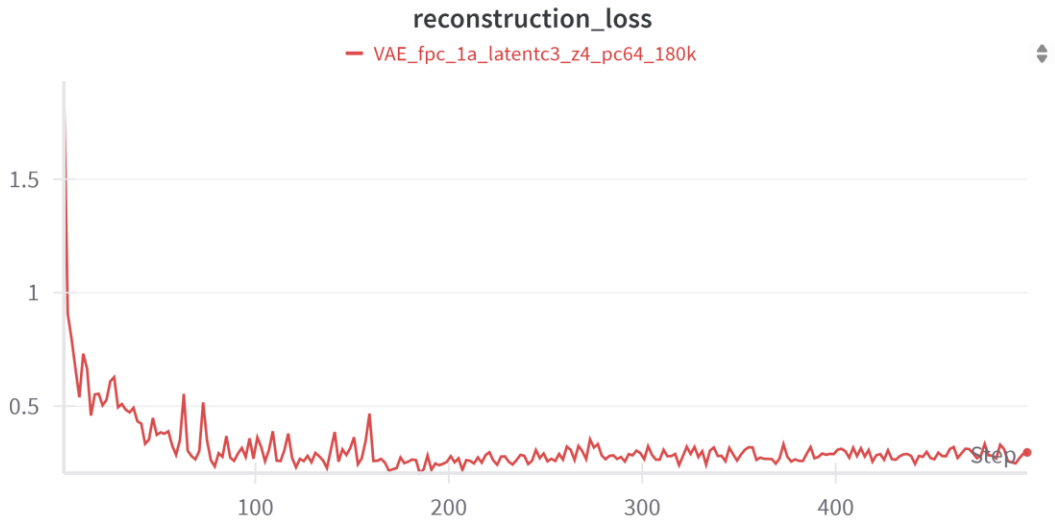
- The ‘confidence’ key is results is nothing but probabilistic value telling how good the grasp is.
- Updated the main function such that now the model generates 100 grasps, but their order and poses are not guaranteed to match the 100 ground truth grasps. So, comparing pred_labels[i] to gt_labels[i] is not meaningful unless you have a 1-to-1 correspondence.
- So we re-evaluate the new generated grasp (Acronym dataset class which say whether the grasp is good/bad) and take it as ground truth. Wheareas, the pred_label for it comes from the confidence value of that grasp.

Classification Loss



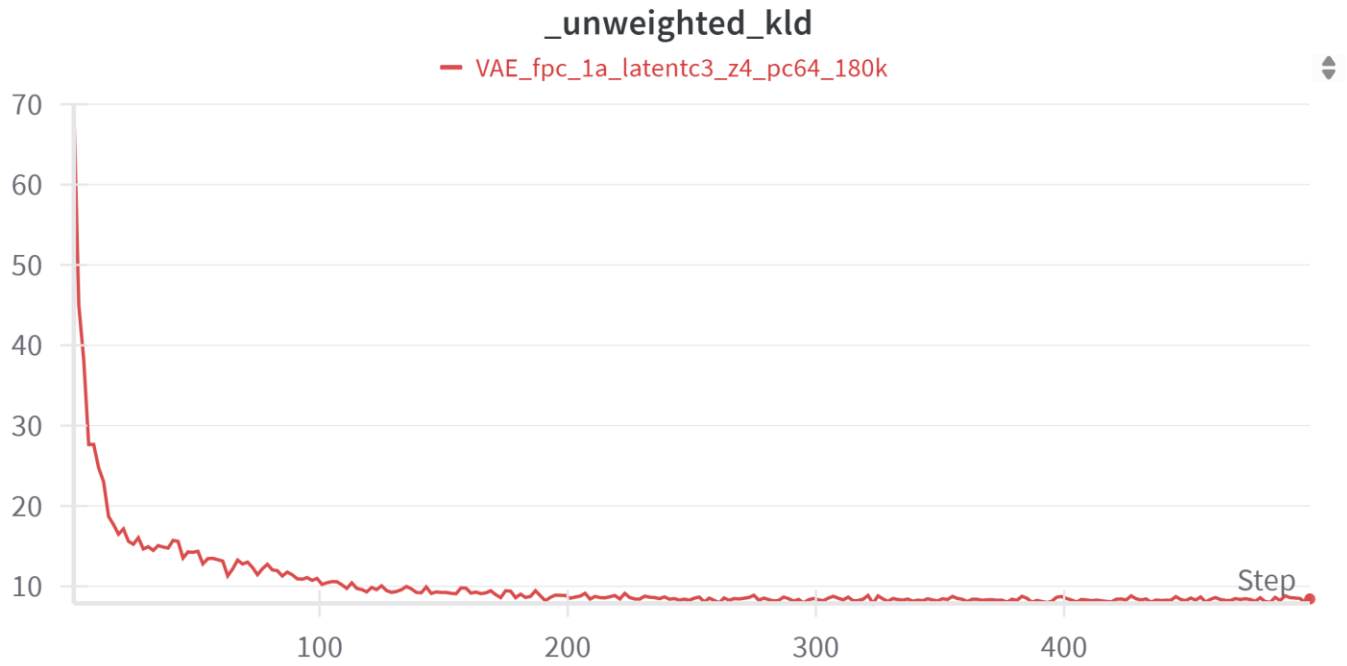
- The binary cross-entropy predicting whether a grasp is successful or not (measures how well the model predicts the correct label (good/bad grasp) for each sample.
- As training progresses, the model learns to better classify grasps as successful or unsuccessful, so the loss should decrease.
- The loss values are quite low (close to zero), which usually means the model is making good predictions (for binary cross-entropy, a perfect classifier would have a loss near zero).

Reconstruction Loss



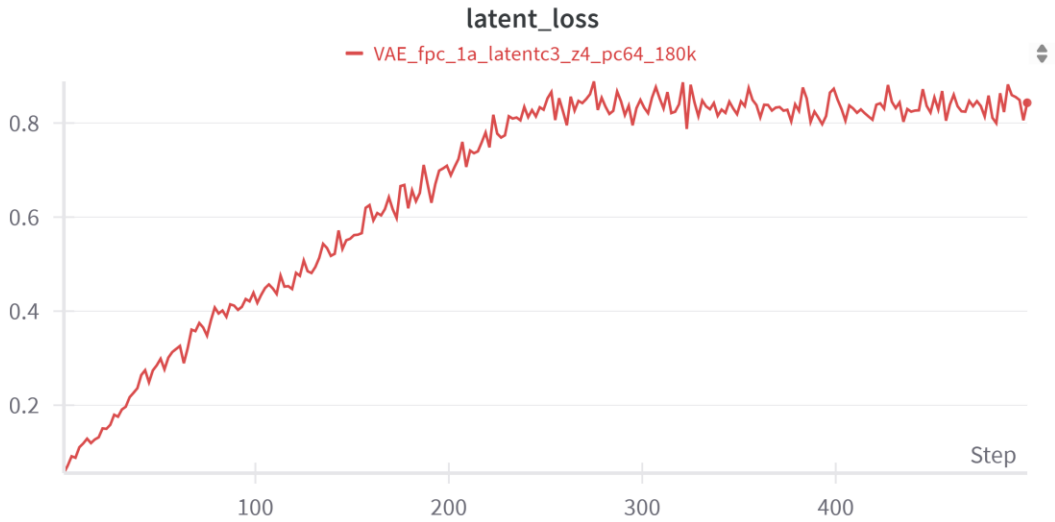
- Mean Squared Error (MSE) or Negative Log Likelihood (NLL) between the input and the reconstructed output. Measures how well the model can reconstruct its input from the latent representation.
- As training progresses, the VAE learns to encode and decode the data more accurately, so the reconstruction error drops.
- **Why is the initial drop so steep?**
  - The model starts with random weights and quickly learns the basic structure of the data, leading to a rapid improvement in reconstruction.
- The final loss value (~0.25) is much lower than the initial value (>1.5), indicating the model has learned to reconstruct the data well.

Unweighted KLD



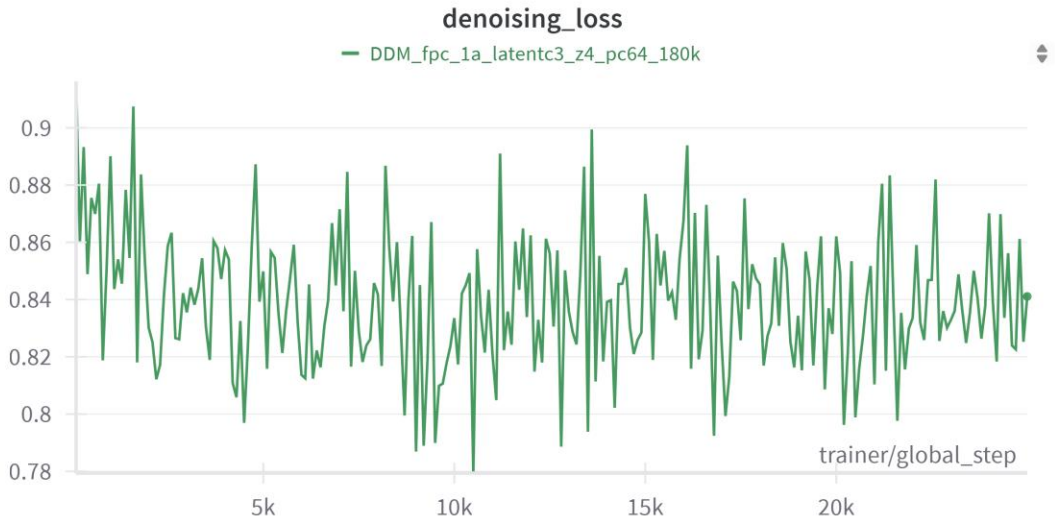
- Measures how much the learned latent distribution diverges from the prior distribution.
- **Why does the KLD decrease?**
  - At the start, the encoder's output distribution is far from the prior, so the KLD is high. As training progresses, the encoder learns to produce latent variables closer to the prior, reducing the KLD.
- The final KLD value (~9–10) is much lower than the initial value (>65), indicating the latent space is now well-regularized.

Latent Loss



- Measures how well the latent space of a model (like a VAE) is being used or regularized.
- Why does the loss increase?
  - This is the opposite of most losses, which decrease. An increasing latent loss can mean the model is learning to encode more information in the latent space, or that the regularization is becoming stronger as training progresses.
- The final value (0.85) is much higher than the initial value (0.08), indicating the model is now encoding more information in the latent space.

Denoising Loss



- Measures how well the model can predict or reconstruct the original (clean) data from a noisy version at each diffusion step.
- In diffusion models, it's common for the denoising loss to decrease early in training and then plateau with some noise, especially if the model is well-tuned.
- The loss values are low (close to zero), which is good for MSE-based denoising loss.

Overall Loss

- DDM loss is same denoising loss
- VAE (red), this is a combination of reconstruction loss, classification loss, and KLD.
- VAE Loss
  - The sharp initial drop is typical as the model quickly learns basic data structure.
  - The subsequent increase and plateau may be due to the KLD or latent loss term increasing as the model starts to use the latent space more (as seen in your earlier latent loss plot).

Threshold

- Lower Threshold (e.g., 0.1):
  - More grasps are classified as “good” (even those with low confidence).
  - Physically, you are being less strict—accepting more grasps as successful, even if the model is not very confident.
- Higher Threshold (e.g., 0.9):
  - Fewer grasps are classified as “good” (only those with very high confidence).
  - Physically, you are being more strict—only the grasps the model is very sure about are accepted as successful.

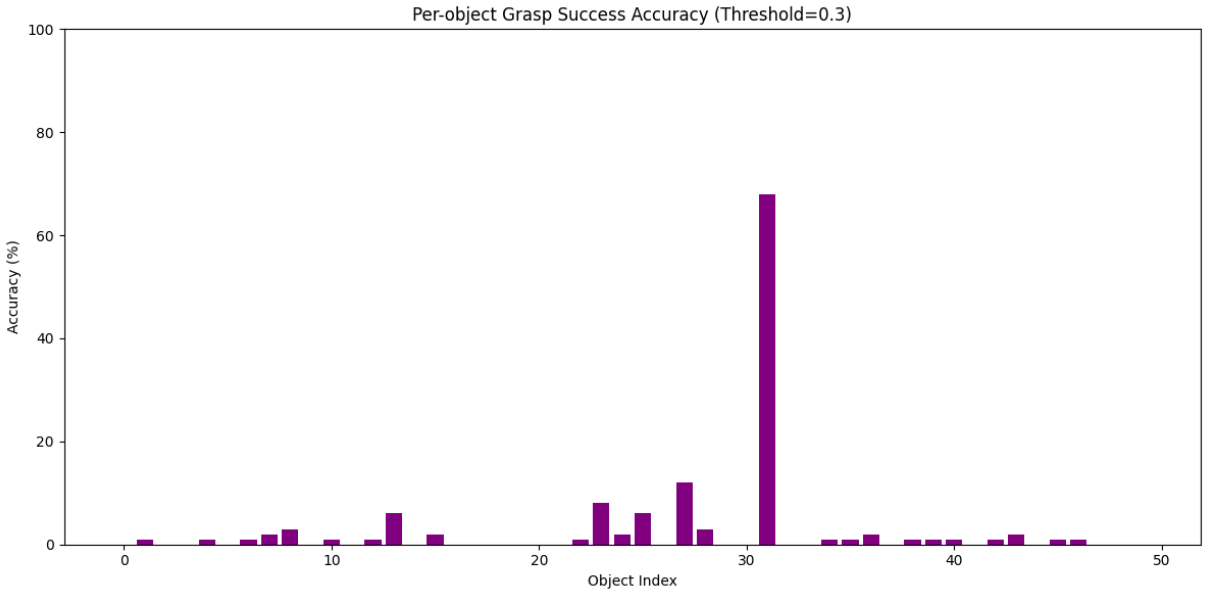
Threshold

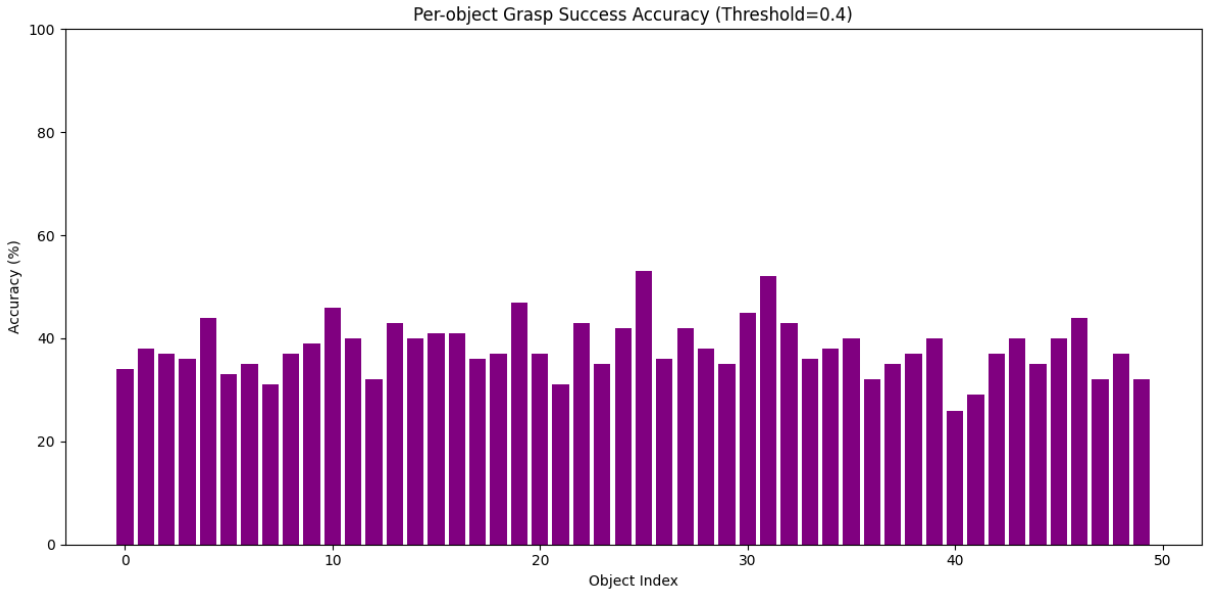
Trade-off Analysis:	By plotting accuracy (or distribution accuracy) vs. threshold, you can see the trade-off between being strict and lenient in classifying grasps.
Best Operating Point:	The threshold with the highest mean distribution accuracy is often the best choice for deployment.
Distribution Matching:	If your generated good/bad counts match the ground truth at a certain threshold, it means your model's confidence scores are well-calibrated for that threshold
Model Calibration:	If accuracy is highest at a low threshold, your model may be under-confident. If it's highest at a high threshold, your model may be over-confident.

Acronym Data Evaluation

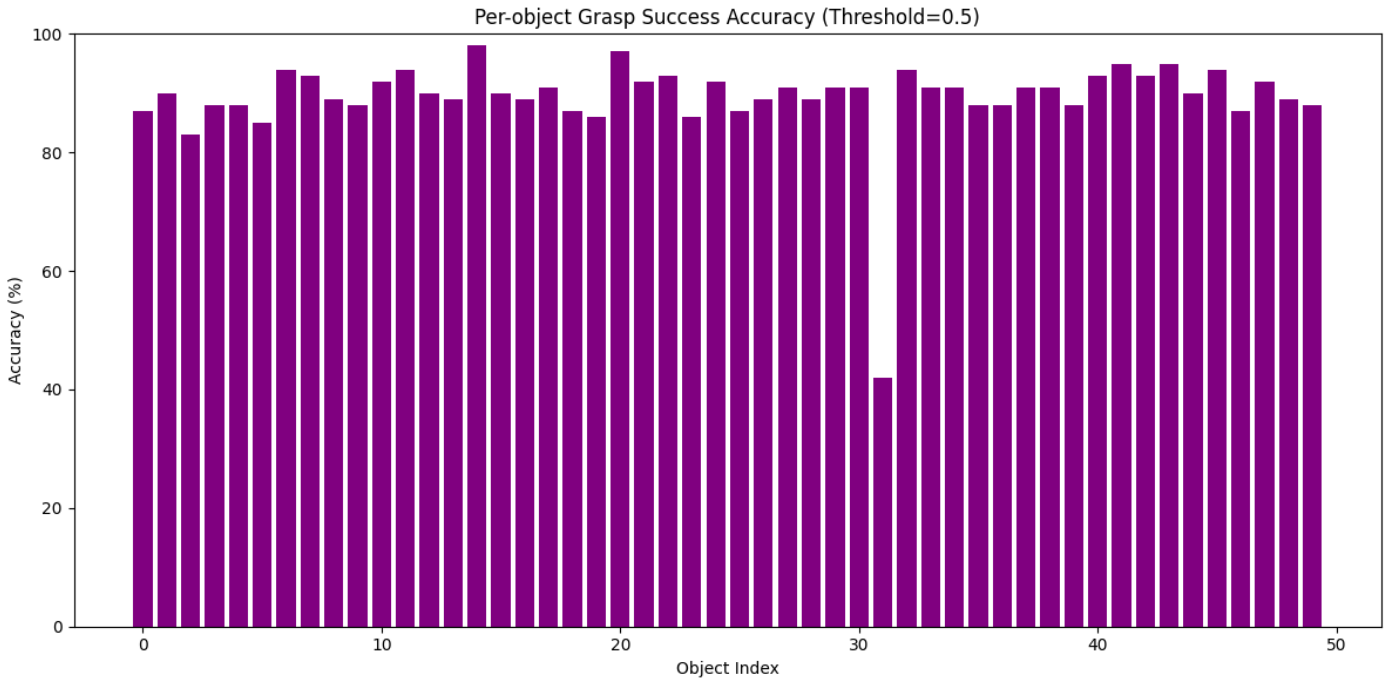
- GraspLDM evaluates by comparing the predicted success of each generated grasp to the ground truth label for that grasp. Accuracy is the fraction of generated grasps that are successful according to the ground truth. No pose-level matching is performed; the evaluation is purely label-based for the generated set.
- Correct Evaluation (per GraspLDM and similar works):
  - Generate N grasps for each object.
  - For each generated grasp: Use the dataset's success-checking function (or simulator) to determine if it is successful.
  - This gives you the "ground truth" label for that generated grasp.
  - Compare the model's predicted label (from confidence thresholding) to this re-evaluated ground truth label.
  - The gripper_mesh_path is the file path to the 3D mesh of your robot gripper (usually in .ply, .stl, or .obj format). This mesh is used for collision checking in the ACRONYM analytic metric.

LDMs Accuracy for different thresholds



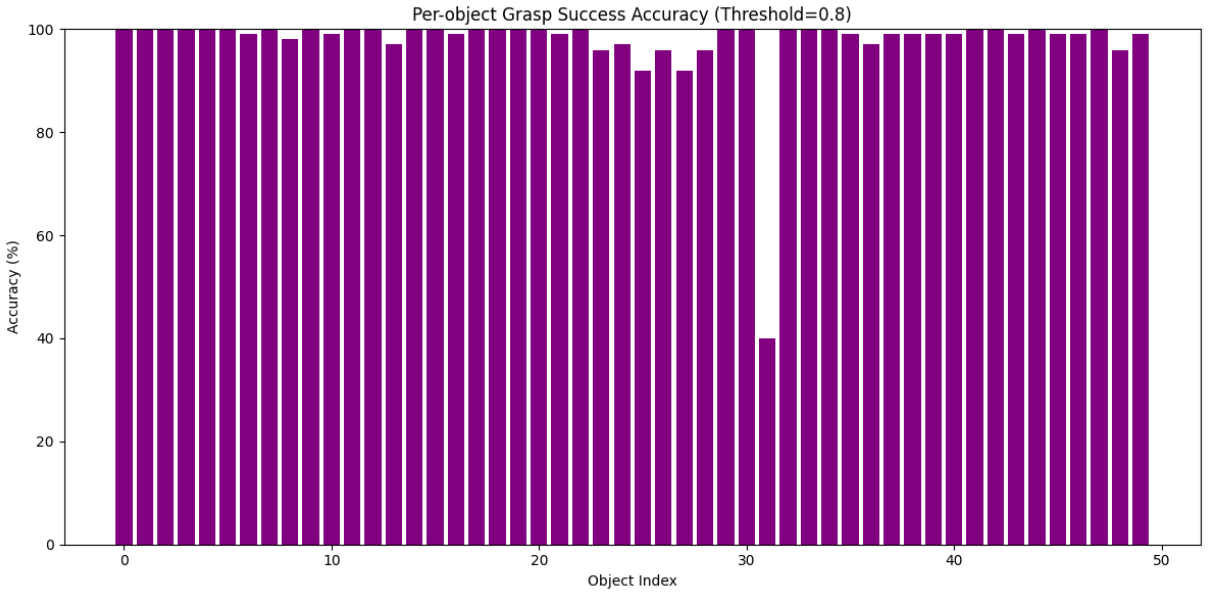


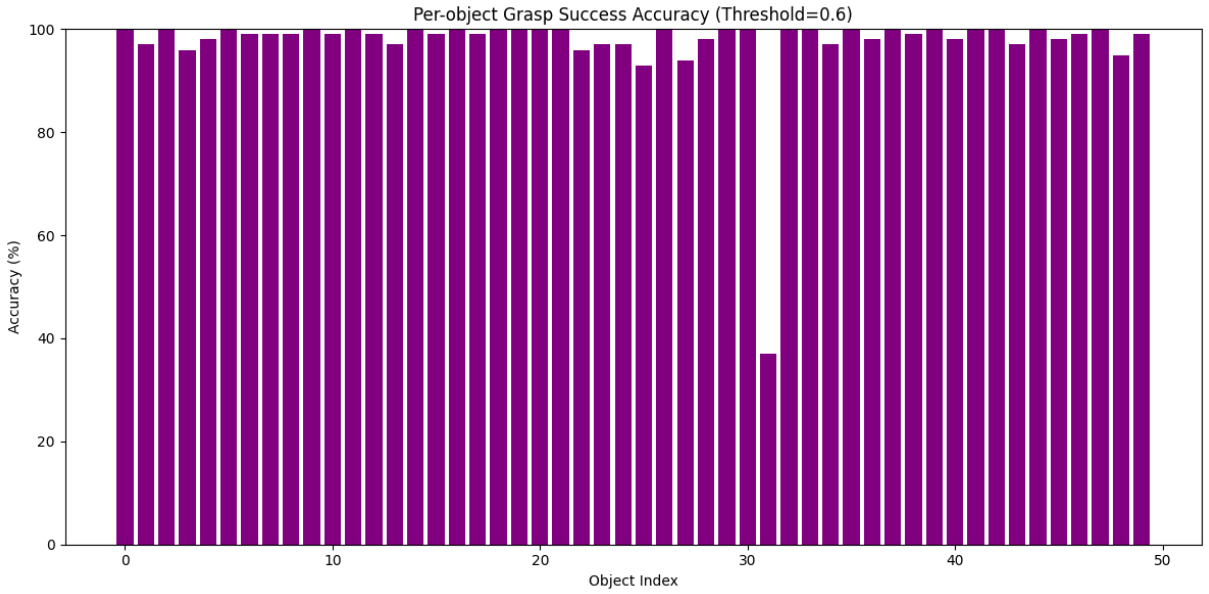
Typical Value: 0.5

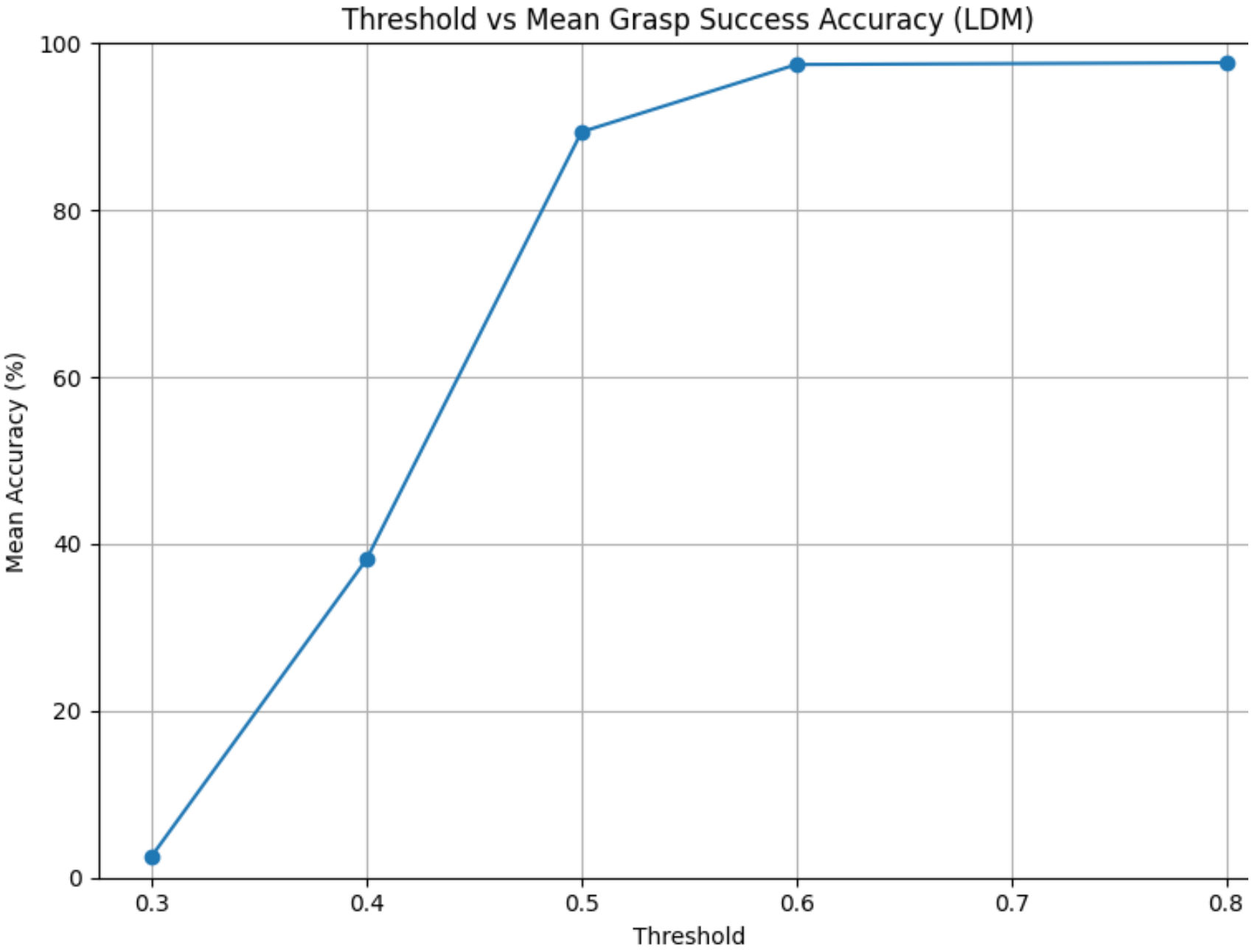


- Plot
  - X-axis: Object Index (each bar represents a different test object)
  - Y-axis: Grasp Success Accuracy (%) for each object, at a threshold of 0.5
- The accuracy is fairly consistent for most objects, showing that the model generalizes well and does not overfit to specific object types.
- It achieves high grasp success accuracy (mostly above 85%) on the vast majority of objects.

LDMs Accuracy for different thresholds

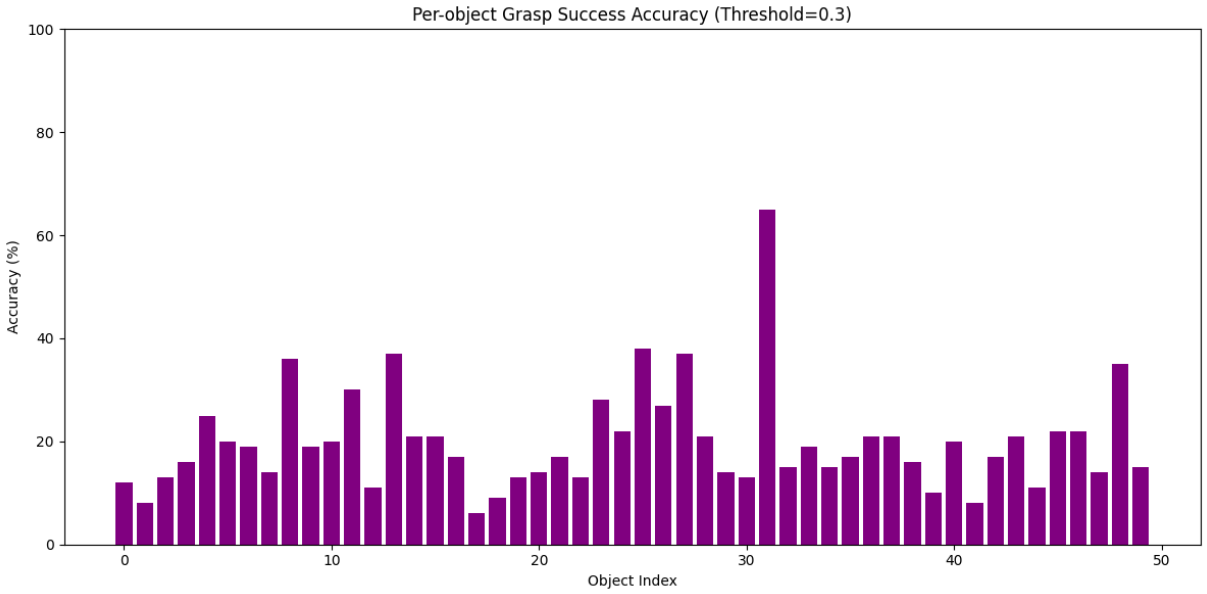


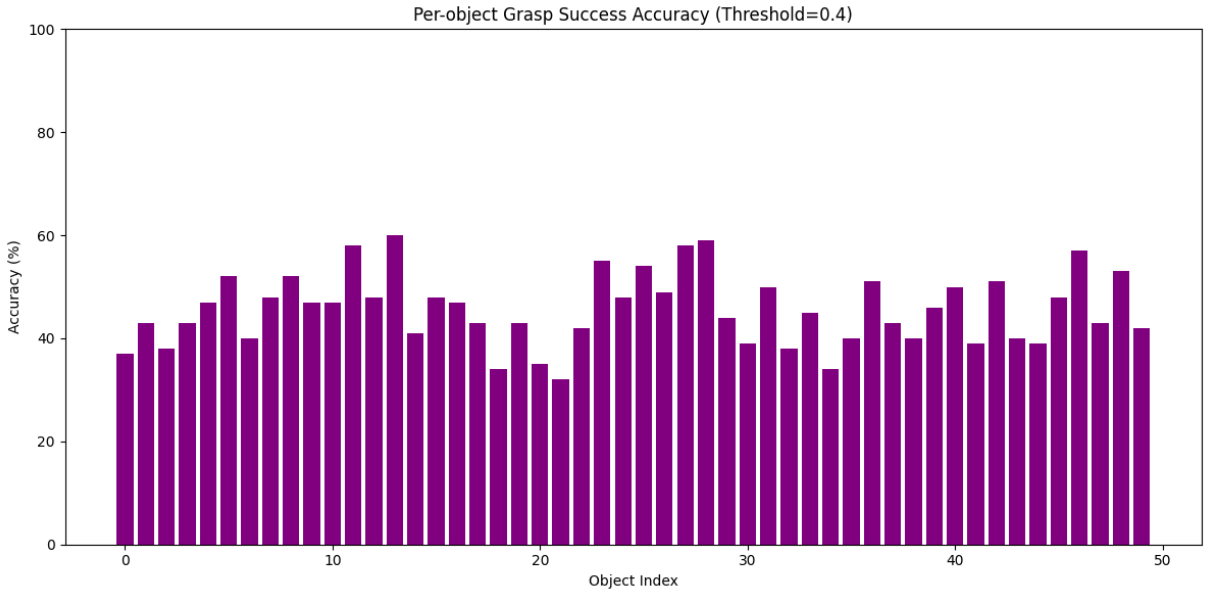




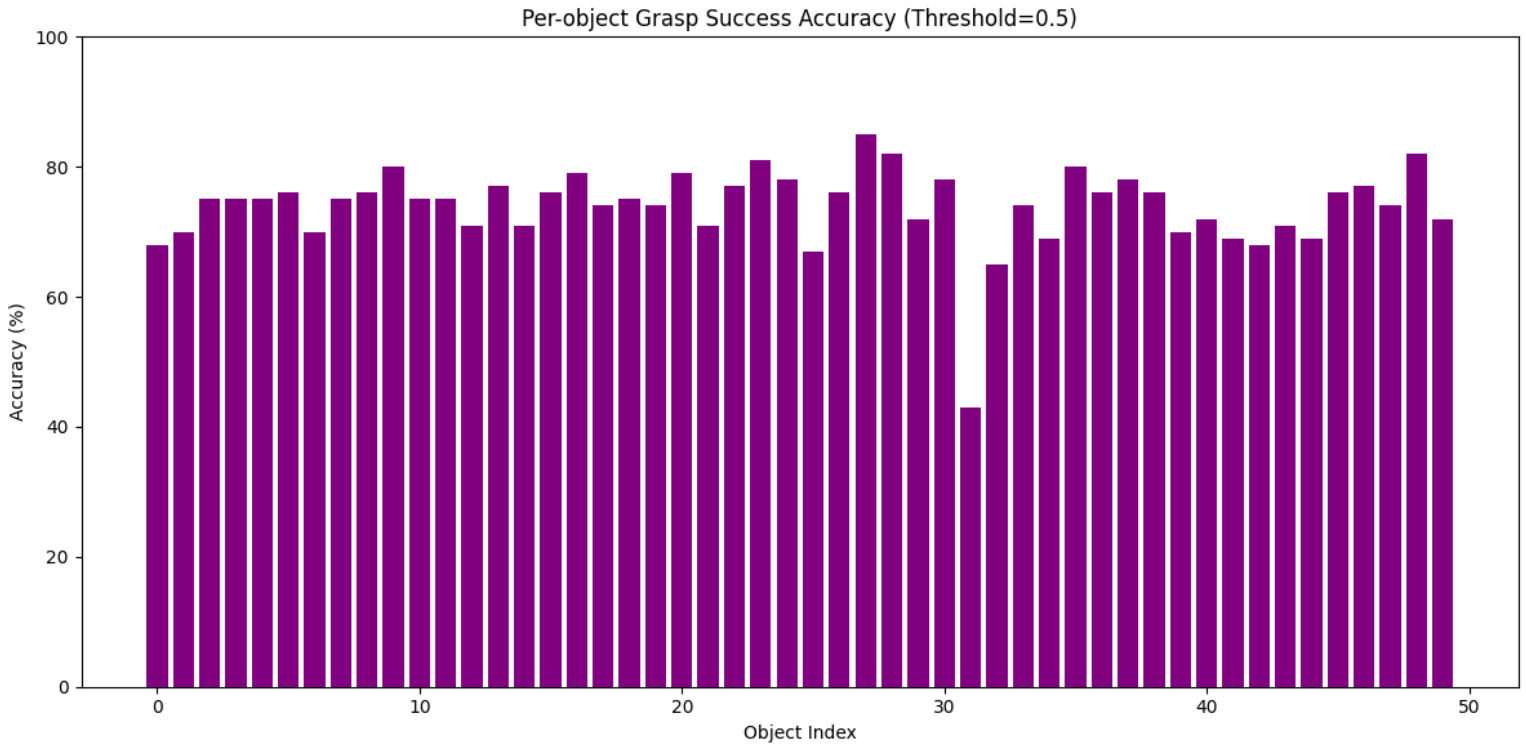
- The model's mean accuracy is highly sensitive to the chosen threshold in the lower range (0.3–0.5), but becomes insensitive at higher thresholds
- The model is likely overconfident in its predictions at higher thresholds, as almost all grasps above these thresholds are successful.
- Setting the threshold at or above 0.5 yields the best mean accuracy, suggesting this is an optimal operating point for your model.

VAEs Accuracy for different thresholds



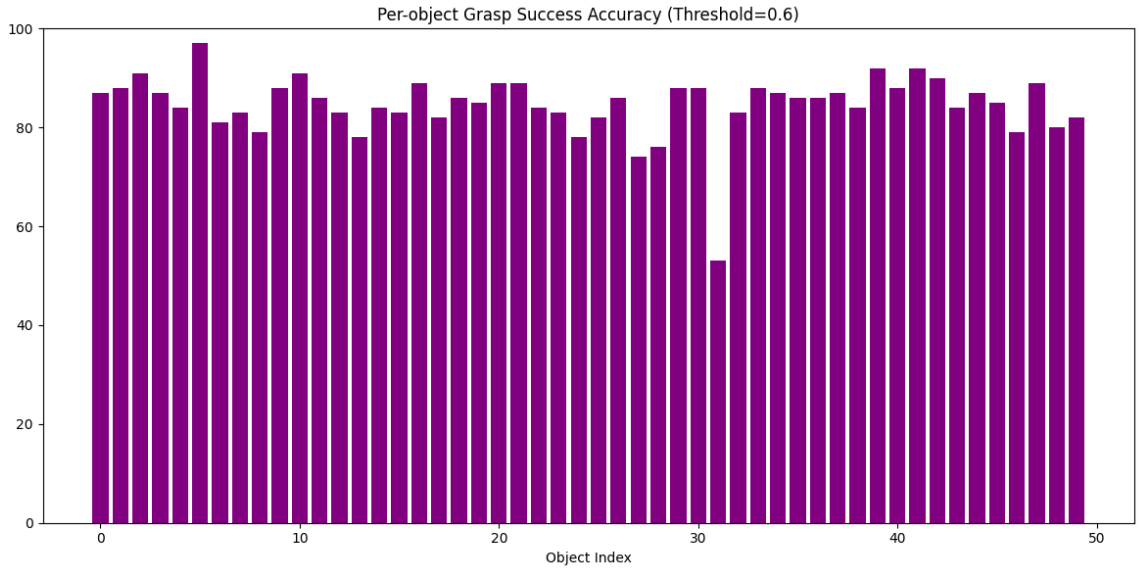


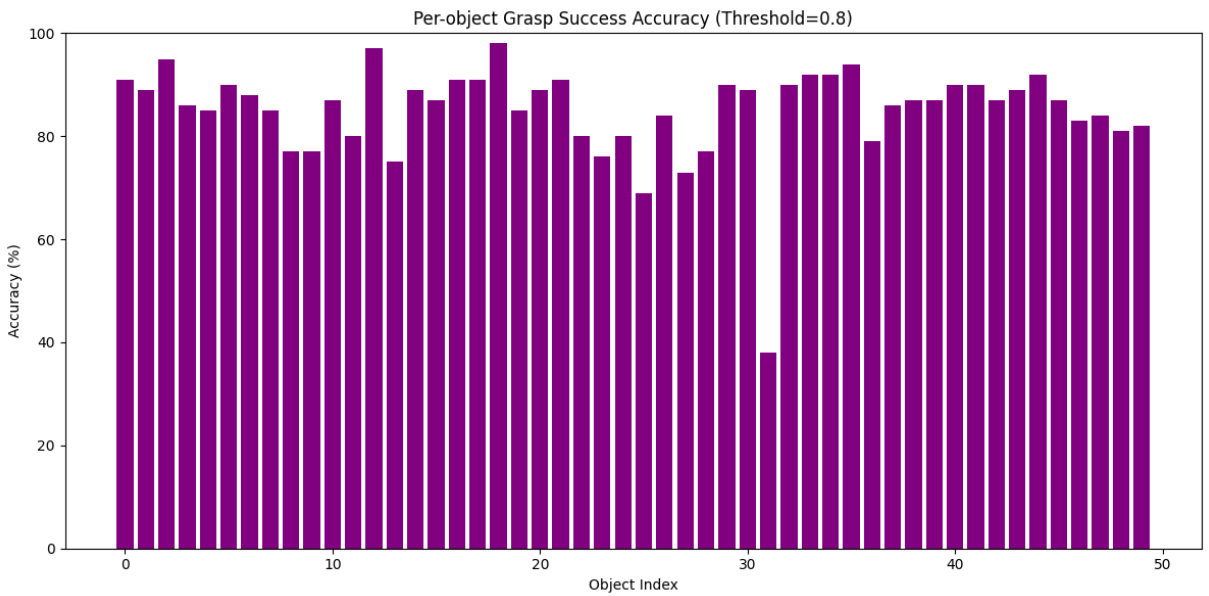
Typical Value: 0.5

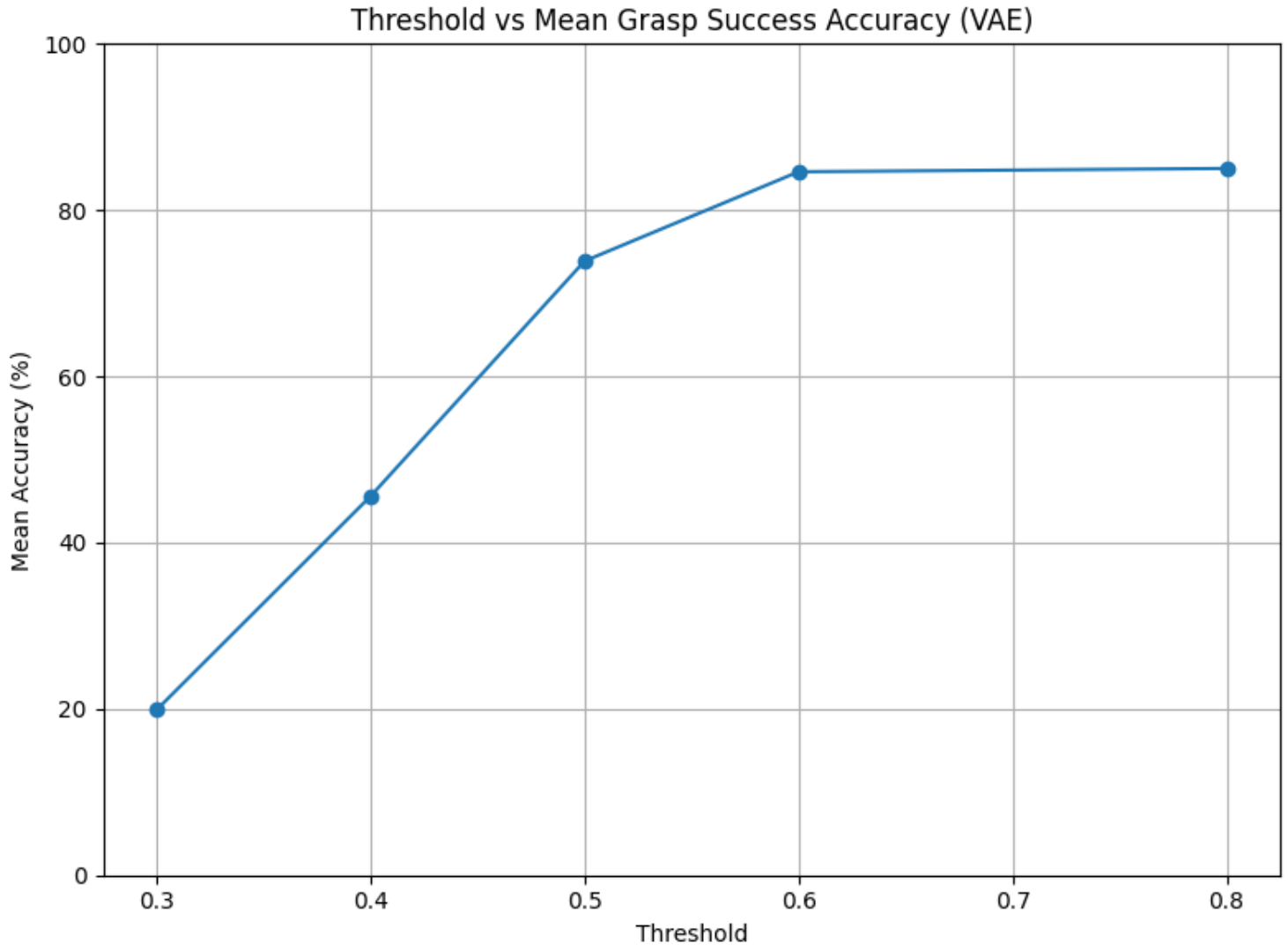


- Most objects have accuracy between 65% and 85%.
- The model achieves moderate success across most objects, but there is clear room for improvement.
- There is a clear outlier (object index ~32) with accuracy just above 40%, and a few others (indices ~34, ~23, ~28) with noticeably lower accuracy (below 70%). Targeting the low-performing objects for further analysis and improvement could significantly boost overall performance.

VAEs Accuracy for different thresholds







- The model's mean accuracy is highly sensitive to the threshold in the lower range (0.3–0.5), with significant gains as the threshold increases.
- The VAE model is less confident overall compared to the LDM, as its accuracy does not reach near 100% even at high thresholds.
- Low thresholds accept many low-confidence (and likely incorrect) grasps, reducing accuracy. Higher thresholds filter out these, increasing accuracy but possibly reducing the number of accepted grasps.