RESEARCH ARTICLE

Efficient Pruning of Text-to-Image Models: Insights from Pruning Stable Diffusion

Samarth N Ramesh[1], Zhixue Zhao[1]

1 University of Sheffield, United Kingdom

Funding: No specific funding was received for this work.

Potential competing interests: No potential competing interests to declare.

Abstract

As text-to-image models grow increasingly powerful and complex, their burgeoning size presents a significant obstacle to widespread adoption, especially on resource-constrained devices. This paper presents a pioneering study on post-training pruning of Stable Diffusion 2, addressing the critical need for model compression in text-to-image domain. Unlike previous work focused on language models or traditional image generation, our study tackles the complex multimodality generation models, and particularly examines the pruning impact on the textual component and the image generation component separately. We conduct a comprehensive comparison on pruning the model or the single component of the model in various sparsities. Our results yield previously undocumented findings. For example, contrary to established trends in language model pruning, we discover that simple magnitude pruning outperforms more advanced techniques in text-to-image context. Furthermore, our results show that Stable Diffusion 2 can be pruned to 38.5% sparsity with minimal quality loss, achieving a significant reduction in model size. We propose an optimal pruning configuration that prunes the text encoder to 47.5% and the diffusion generator to 35%. This configuration maintains image generation quality while substantially reducing computational requirements. In addition, our work uncovers intriguing questions about information encoding in text-to-image models: we observe that pruning beyond certain thresholds leads to sudden performance drops (unreadable images), suggesting that specific weights encode critical semantics information. This finding opens new avenues for future research in model compression, interoperability, and bias identification in text-to-image models. By providing crucial insights into the pruning behavior of text-to-image models, our study lays the groundwork for developing more efficient and accessible AI-driven image generation systems.[1]

Corresponding authors: Samarth N Ramesh, snramesh1@sheffield.ac.uk; Zhixue Zhao, zhixue.zhao@sheffield.ac.uk

1. Introduction

Large language models (LLMs) and diffusion models often face the challenge of having exceptionally large model sizes[1][2]. While these extensive parameters allow them to understand high-quality text and generate images, they also

result in substantial computational demands and resource consumption[3][2][4]. With billions of parameters, these models become prohibitively expensive to operate, limiting their use primarily to large corporations. To address this issue, significant research has been devoted to model compression techniques for deep learning models. Compressing these models can make them deployable on edge devices[5][6][7], reduce response times[8][9], enable real-time applications[10][6], and greatly increase their accessibility.

Quantization[11] and pruning[12][13][14] are two primary compression approaches. Quantization focuses on reducing the precision of a model's weights, while pruning seeks to identify and eliminate redundant weights. Both methods have been well studied for LLMs but much less has been done for vision language models[15]. Further, recent advancements have introduced post-training pruning methods for LLMs, which offer compressed models without the need for retraining[16][17].

However, there has been relatively little research on post-training pruning for vision language models. As far as we know, no prior work has addressed post-training pruning of text-to-image models. This paper seeks to fill this gap by investigating the impact of post-training pruning on Stable Diffusion 2[18]. Additionally, this study examines how pruning impacts the textual encoder and the image generator of the model separately, and identifies the optimal sparsity levels for each component.

2. Related Work

2.1. Text-to-Image Models

The field of image generation was initially dominated by Generative Adversarial Networks (GANs)[19][20] and Diffusion models[21]. While GANs produce high-quality images, their variability is constrained by data used in the adversarial training process and they require substantial effort to scale for diverse, complex applications[22]. On the other hand, Diffusion models, built on denoising autoencoders, excel across various image synthesis tasks[23][24], but their high computational demands, requiring extensive GPU resources for training and inference, present significant challenges[25].

The Stable Diffusion models released by Stability AI have made significant advances in text-to-image generation and are recognized as state-of-the-art[26][27]. These models build upon diffusion models by incorporating a variational autoencoder to encode images in a latent space, offering several advantages over training in the image space, such as significantly reducing the computational resources needed for both training and inference[26]. The diffusion process is applied in this latent space, with cross-attention layers used to condition the model on rich representations of the input. For text-to-image tasks, this representation is generated by a transformer model[18]. In this study, we focus on Stable Diffusion 2, which uses the CLIP model's text encoder[28] as the transformer and a U-Net[29] as the diffusion model. Stable Diffusion 2, a state-of-the-art model and a foundational architecture for later versions[27][30], contains 1.2 billion parameters, with 340 million in the CLIP text encoder and 860 million in the U-Net diffusion generator. Its large size makes it an ideal candidate for model compression techniques.

2.2. Pruning

Pruning works on the principle that some connections in a network are of less importance than others[12]. The method of finding these connections is of utmost importance for the post-training pruning. We introduce state-of-the-art pruning methods experimented in LLMs and their main difference is how they identify the weights to be pruned.

Magnitude Pruning

Magnitude-based pruning[14] is one of the simplest and most commonly used techniques for pruning neural networks. It operates under the assumption that a connection's importance correlates with the absolute value of its weight. As described in Equation 1, the weights are ranked according to their magnitude, and the least important weights are pruned by setting them to zero. While this method may not provide optimal performance compared to more advanced techniques, it is easy to implement, computationally efficient, and serves as a strong baseline for further refinement.

$$\text{Low Importance Connections} = \text{argmin}_i |w_i|$$

SparseGPT

SparseGPT[16] tackles the task by attempting to solve the problem of identifying the optimal pruning mask. The goal is to find a sparse subset of weights that minimizes reconstruction error. While the exact solution to this problem requires $O(d^4)$ time complexity[16], where d is the dimension of the hidden layer, SparseGPT achieves an approximate solution in $O(d^3)$, offering a more efficient approach. This method has consistently outperformed magnitude pruning across LLMs of various sizes.

Wanda

Weight and Activation Pruning (Wanda)[17] is a more recent method that surpasses SparseGPT in performance and has become widely adopted in various text-based models. Wanda is grounded in the observation that emergent properties in LLMs are often associated with the presence of "outliers"[31]. These outliers are defined as exceptionally large output values from neurons, which can be up to 100 times greater than typical output values. Such activations serve as valuable indicators of the importance of specific weights.

Wanda extends the magnitude pruning by incorporating the values of the activations that serve as inputs for these weights. The scoring metric for this method is expressed in Equation 2, where W_{ij} represents the value of the weights, X_j denotes the activations input to W_{ij}, and $\|X_j\|_2$ is the l_2 norm of these activations. This approach allows for a more nuanced assessment of weight importance, contributing to improved pruning effectiveness.

$$S_{ij} = |W_{ij}| \cdot \|X_j\|_2$$

Outlier-Weighted Layerwise Pruning (OWL)

A significant observation regarding outliers in LLMs is that the density of observed outliers within a particular layer

correlates with that layer's importance to the model's emergent properties[32][33][34]. Based on this insight, Outlier-Weighted Layerwise Pruning (OWL)[35] seeks to distribute sparsity across different layers of the model in an uneven manner. The required layer-wise sparsity is influenced by the number of outliers present; consequently, layers deemed more important are pruned less aggressively, while those considered less critical are pruned more extensively. OWL can be integrated with other pruning algorithms, such as Magnitude and Wanda as it does not specify which weights to prune; rather, it provides recommendations for local sparsity levels within the layers.

While post-training pruning methods have been extensively studied for language models, their effectiveness on text-to-image models remains largely unexplored. Given the distinct architectures and training protocols of text-to-image models, it is challenging to infer how these pruning techniques, originally developed for language models, will perform in this different context.

3. Experimental Setup

This paper investigates the pruning of Stable Diffusion 2, a widely used text-to-image generation model. Stable Diffusion 2 has 1.2 billion parameters of which 340 million parameters (28%) are in the CLIP text encoder and 860 million parameters (72%) are in the U-Net diffusion generator. Our initial experiments focus on examining the effects of independently pruning individual components of the model while maintaining the integrity of the other parts. For example, only prune the text encoder component and keep the diffusion generator untouched. Following this, full model pruning is explored across multiple axes to determine the optimal balance of sparsities between the text encoder and the diffusion generator. The evaluation of the best configuration for full model pruning is informed by the results obtained from the individual component pruning experiments.

3.1. Pruning Single Component

For text-encoder-only pruning, four different techniques are tested, namely magnitude pruning, Wanda, magnitude pruning with OWL and Wanda with OWL. For each of these techniques, multiple sparsities are tested in steps of 10% and more granular tests are performed at intervals of interest.

To date, there has been limited research on post-training pruning specifically for diffusion models. Some studies have identified efficient pruning techniques that involve retraining[36], and a recent proposal introduced a novel approach for iteratively pruning Latent Diffusion models[37], such as Stable Diffusion. However, the scope of this paper is limited to magnitude pruning for the diffusion component of the model, leaving the exploration of other techniques for future research. As with the text encoder, multiple sparsity levels are evaluated in increments of 10%, and more detailed assessments are conducted at intervals of particular interest.

3.2. Full Model Pruning

Two distinct approaches are employed for the experiments to find the optimal configuration for full model pruning. The first

approach aims to determine the best ratio of sparsities between the text encoder and image diffusion generator if a specific full model sparsity is desired. At various levels of full model sparsity, the distribution of this sparsity between the two components is outlined in Table 1. It is important to note that certain ratios associated with higher full model sparsities are unfeasible due to the significantly smaller size of the text encoder compared to the diffusion generator. For example, when aiming to prune the full model to 50%, attempting to distribute the sparsity in a 75:25 ratio for text and image would result in pruning the text encoder to 177%, which is clearly not achievable.

Table 1. Sparsity is distributed between components in the ratios shown in Text:Image Ratio and Text and Image Sparsities represent the fraction of weights pruned in those components

Full Model Sparsity	Text:Image Ratio	Text Sparsity	Image Sparsity
20%	75:25	53%	7%
20%	50:50	35%	14%
20%	25:75	18%	21%
30%	75:25	80%	10%
30%	50:50	53%	21%
30%	25:75	27%	31%
40%	50:50	71%	28%
40%	25:75	35%	42%
50%	50:50	89%	35%
50%	25:75	44%	52%
60%	25:75	53%	63%

The second approach to full model pruning relies on the findings from the individual component pruning experiments. Both the text encoder and the diffusion generator exhibit identifiable drop-off points in performance, which will be further examined in the results section. Assuming that the sub-models remain largely unaffected until reaching these drop-off points, the full model is pruned to align with the drop-off thresholds of both sub-models. This leads to a recommended configuration for maximal pruning that balances performance and sparsity effectively.

3.3. Dataset

We use the Microsoft COCO: Common Objects in Context (MSCOCO) 2017 dataset[38] which has a large number of real images and corresponding captions. MSCOCO is a commonly used dataset for computer vision tasks. For our experiments, we use 10,000 randomly sampled images and their corresponding captions.

3.4. Evaluation

FID

The Fréchet Inception Distance (FID)[39] defines a distance metric between two sets of images and measures how different the two sets are. A lower FID indicates that the two sets of images are very similar.

In our experiments, we use the FID to compare images generated from MSCOCO captions with corresponding real images. For each model we generate 10,000 images to calculate the FID.

CLIP Score

The CLIP Score[40] uses a multi-modal text and image model to directly compare a generated image with the provided prompt. It uses the CLIP model to calculate the similarity and a lower CLIP Score indicates a higher correlation in the semantic content of the prompt and image.

In our experiments, for each pruned model, we generate 10,000 images and calculate the average similarity of each image with its prompt using the CLIP Score.

4. Results and Analysis

The primary objective of this study is to analyze the impact of pruning on text-to-image models, specifically focusing on the individual components of Stable Diffusion 2, namely the CLIP text encoder and the U-Net diffusion generator. Our aim is to explore the optimal trade-off between model performance and computational resource efficiency. We initially approach this problem by examining the effects of pruning when targeting only one component at a time. The insights gained from this analysis subsequently inform our strategy for full model pruning.

All experimentation is quantitatively evaluated using FID[39] and CLIP Score[40] metrics on 10,000 images generated from captions present in the MSCOCO 2017 dataset[38]. These metrics are reinforced by qualitative evaluation of generated image quality.

4.1. CLIP Pruning Only

We experiment with two pruning methods on the CLIP text encoder and extend these by applying OWL to test whether outlier weighting improves performance in text-to-image models. Figures 1, 3, 5, 7 show the FID and CLIP scores for each method. At lower sparsity levels, all techniques result in minimal performance degradation. However, as sparsity increases, each method experiences a sharp performance decline at specific thresholds. Qualitative evaluations of the generated images corroborate these quantitative results.

Magnitude Pruning

As the simplest and most straightforward method, magnitude pruning serves as an important baseline. When pruning up to a sparsity of 60%, the model performs similarly to the original, with only a minimal decline in performance. However, at

62.5% sparsity, the model experiences a sudden drop in performance. This is further confirmed by visual inspection of the generated images, which show a clear degradation in quality. At 62.5%, the model struggles to correctly interpret the prompt, as evidenced by a misframed dog and the absence of a field in the images. Beyond this threshold, the model's performance deteriorates drastically, with the generated images becoming complete noise.

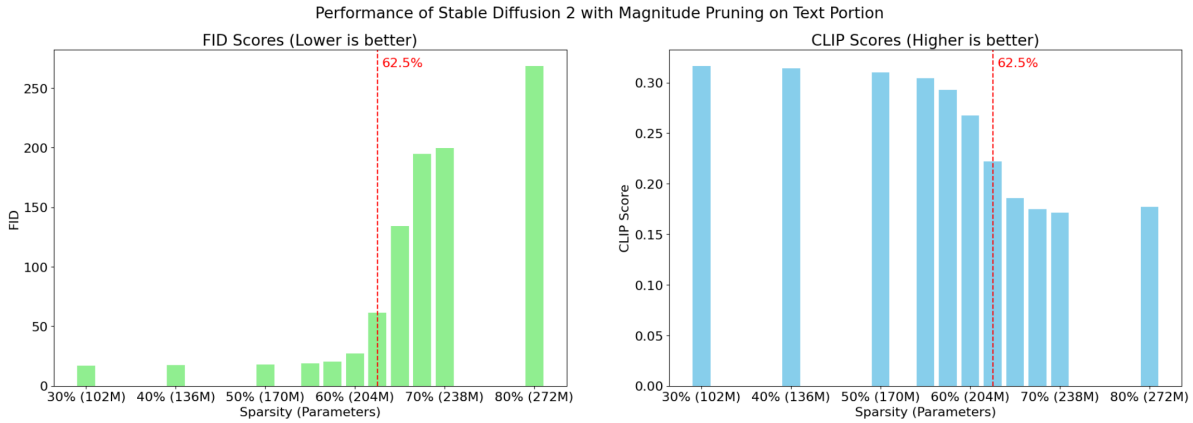

Figure 1. Pruning only Text Encoder of Stable Diffusion 2 using Magnitude Pruning - Sharp drop off in performance observed at 62.5%

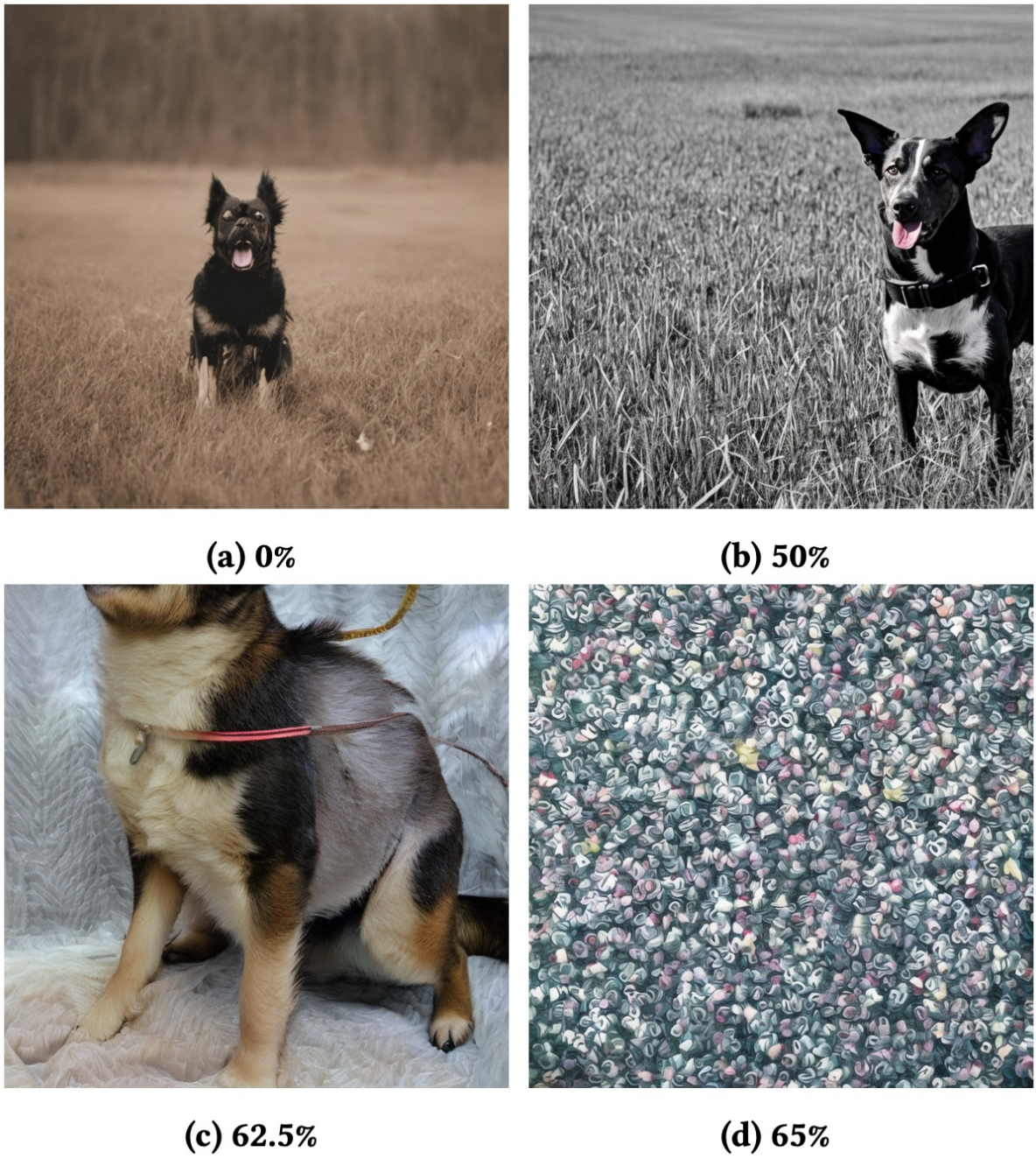

Figure 2. Text Encoder Magnitude Pruning Examples - Text Encoder breaks down beyond 62.5%

Wanda Pruning

Despite being one of the most effective pruning techniques for many language models, Wanda appears to be poorly suited for pruning the text component in this text-to-image model. When compared to the baseline magnitude pruning, Wanda exhibits similar behavior but consistently underperforms. Like magnitude pruning, Wanda shows minimal impact on performance up to a certain threshold, after which there is a sharp decline. However, for Wanda, this threshold occurs at 60% sparsity, making it less effective than magnitude pruning.

An interesting observation arises at 80% sparsity, where the FID score is unexpectedly lower than at 70%, suggesting

that Wanda might perform better at higher sparsities. This anomaly warrants further investigation. Additionally, when visually inspecting the images in Figure 4, the deterioration threshold appears to be around 62.5%, slightly different from the 60% threshold indicated by the FID and CLIP Score metrics. This discrepancy is likely due to the specific behavior of the model on this particular prompt, whereas the metrics reflect the average performance across 10,000 distinct prompts.

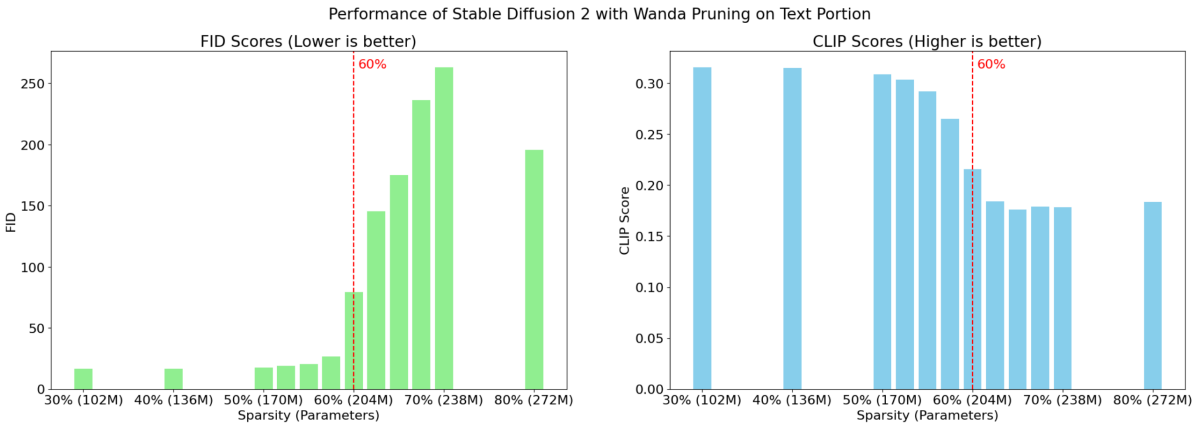

Figure 3. Pruning only Text Encoder of Stable Diffusion 2 using Wanda Pruning - Sharp drop off in performance observed at 60%

Figure 4. Text Encoder Wanda Pruning Examples - Text Encoder breaks down beyond 62.5%

Magnitude Pruning with OWL

Applying outlier weighting to magnitude pruning shows minimal impact overall. While some sparsities exhibit slight improvements, the differences are generally insignificant. This finding aligns with the observations made by the authors of the OWL paper, who noted that outlier weighting tends to offer meaningful improvements primarily at higher sparsity levels.

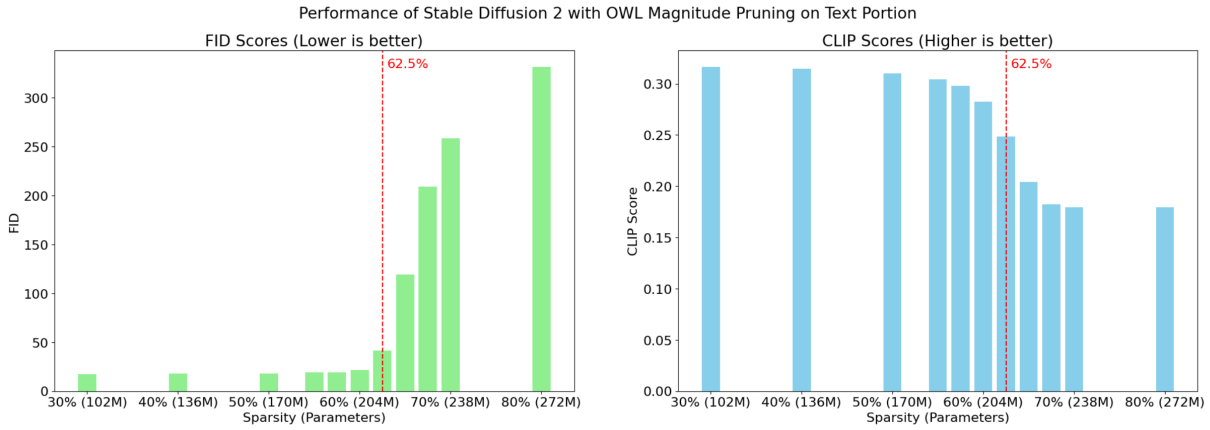

Figure 5. Pruning only Text Encoder of Stable Diffusion 2 using Magnitude Pruning with OWL - Sharp drop off in performance observed at 62.5%

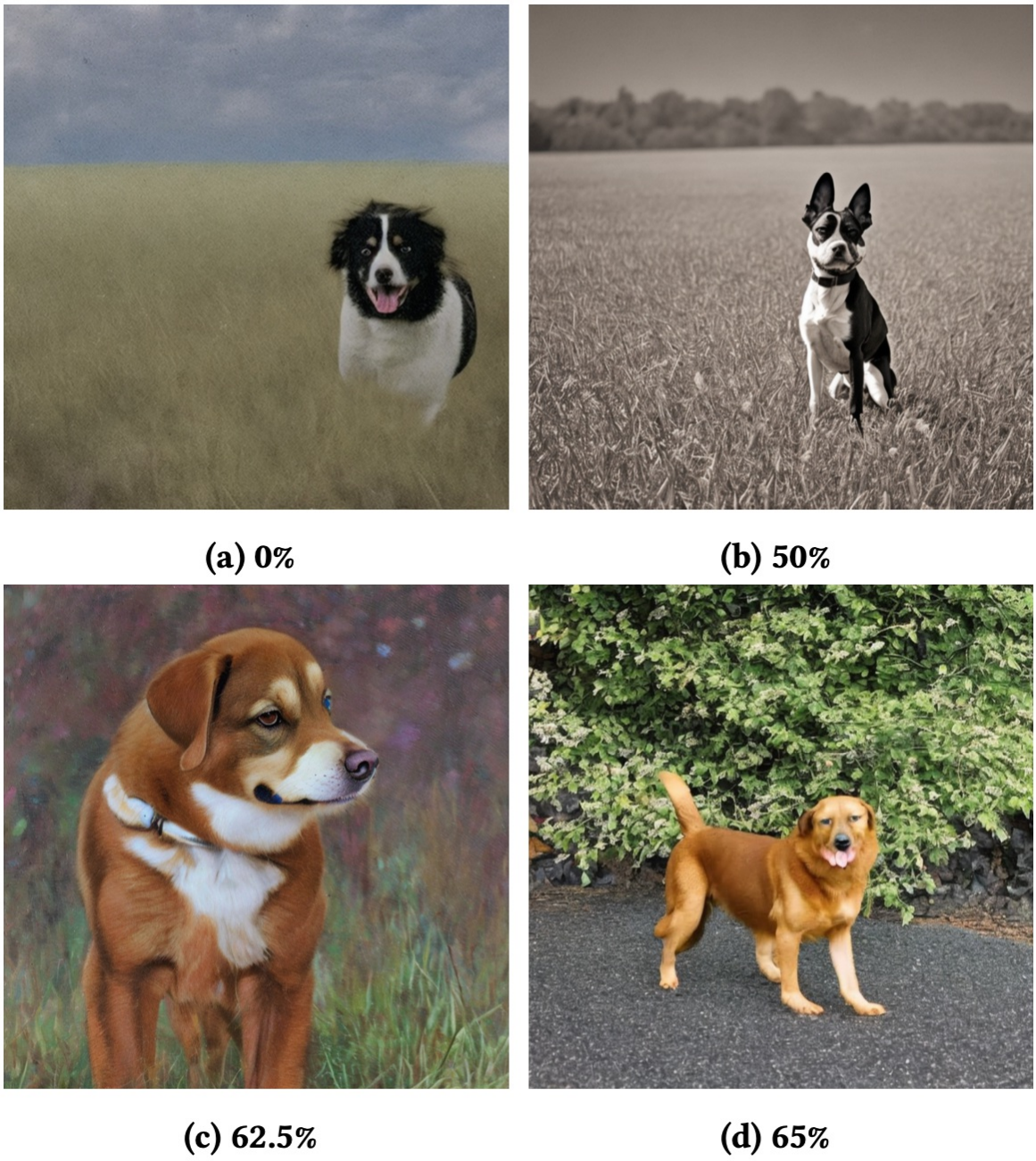

Figure 6. Text Encoder Magnitude Pruning with OWL Examples - Text Encoder breaks down beyond 62.5%

Wanda Pruning with OWL

As with magnitude pruning, applying outlier-based weighting to Wanda pruning does not result in significant improvements.

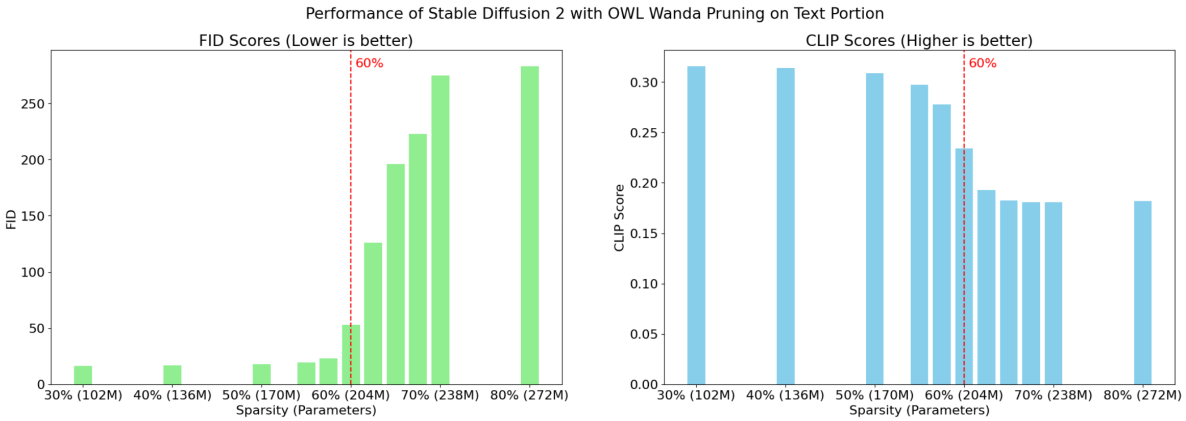

Figure 7. Pruning only Text Encoder of Stable Diffusion 2 using Wanda Pruning with OWL - Sharp drop off in performance observed at 60%

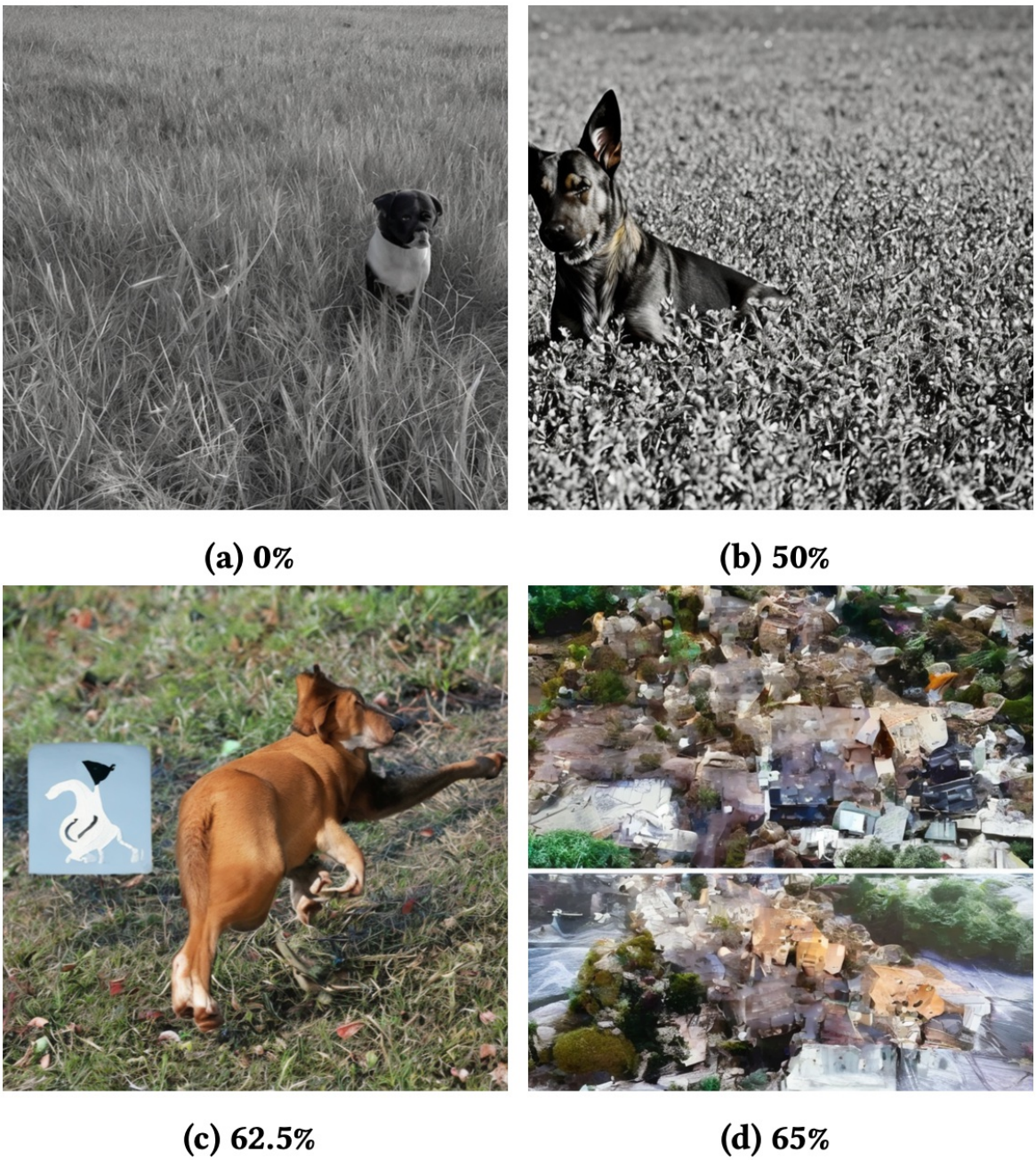

Figure 8. Text Encoder Wanda Pruning with OWL Examples - Text Encoder breaks down beyond 60%

Comparison of Pruning Techniques

The finding that Wanda pruning is outperformed by magnitude pruning is quite surprising and underscores the need for pruning techniques tailored specifically to text-to-image models. As shown in Figure 9, magnitude pruning consistently outperforms Wanda at every sparsity level. This is particularly notable given that, in most NLP models, magnitude pruning is typically the weakest among post-training pruning methods. This suggests a significant opportunity to develop specialized pruning algorithms for text-to-image models.

Another key observation is that Outlier Weighted Layerwise (OWL) pruning, when applied on top of both magnitude and

Wanda pruning, results in only marginal improvements. Figure 9 also highlights the sharp performance drop-off, a pattern observed across all pruning algorithms tested.

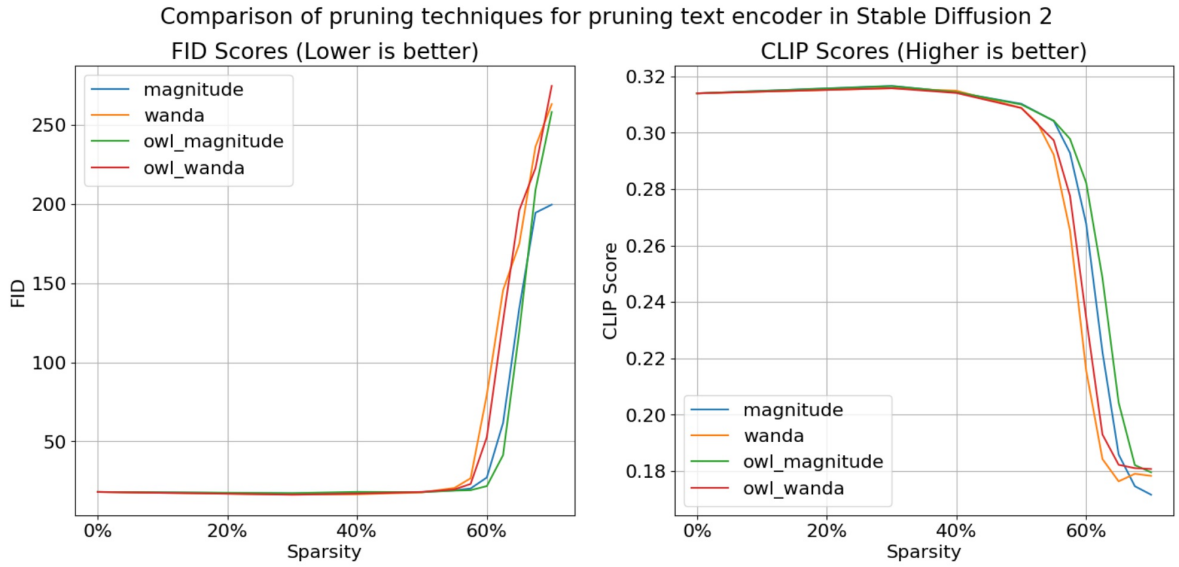

Figure 9. Comparing Pruning Techniques for pruning only text encoder

4.2. U-Net Pruning Only

For the diffusion component of the model, we observe a more gradual decline in performance as pruning increased. The model maintains strong performance at lower sparsity levels, but its performance steadily decreases as sparsity exceeded 40%. Based on a qualitative analysis, 50% sparsity is identified as the point where the model's performance significantly deteriorated, and this threshold is later used in full model pruning. This gradual decline is evident when examining the pruned images from 30% to 60%, as shown in Figure 14 in the appendix.

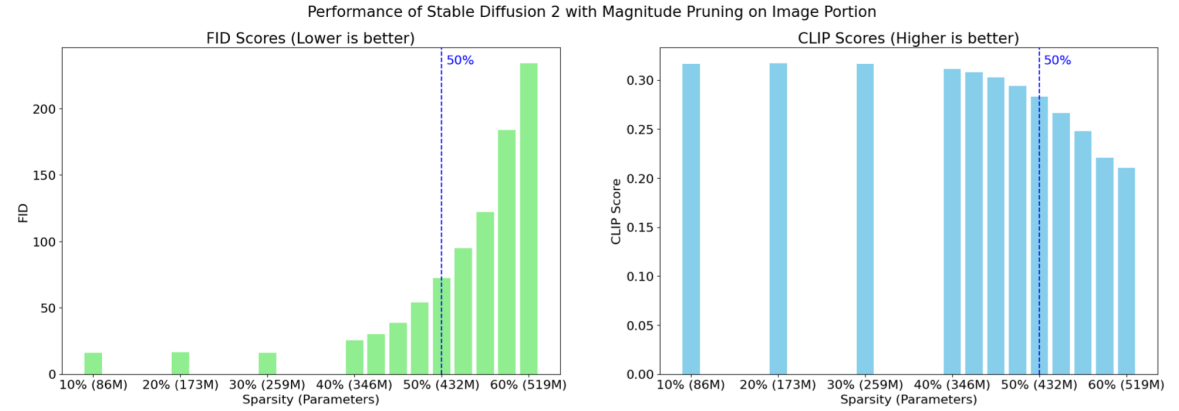

Figure 10. Pruning only Image Diffusion Generator of Stable Diffusion 2 using Magnitude Pruning - Gradual decline in performance

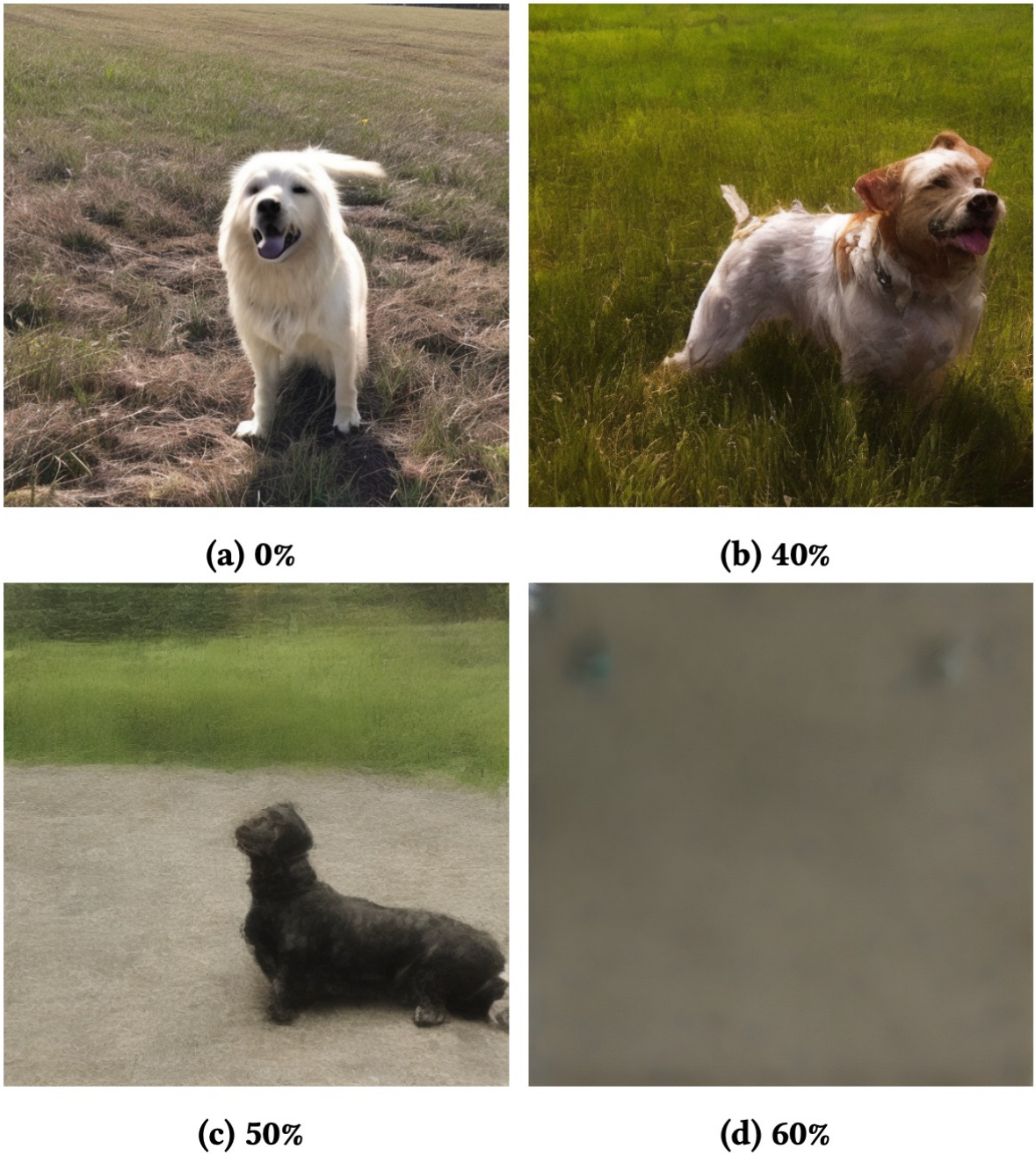

Figure 11. Diffusion Generator Magnitude Pruning Examples - Performance greatly suffers beyond 40%

4.3. Comparison between Text and Image Pruning

To address the question of which component—the text encoder or the image diffusion generator—yields better results when pruned, we compare the best-performing pruned models for each approach. This comparison is presented in Table 2.

Total Sparsity	Text Sparsity	Image Sparsity	FID	CLIP Score
16.9%	60%	0%	21.7	0.282
28.7%	0%	40%	25.38	0.311

Table 2. Comparison between best model obtained pruning each component individually

Comparing the two models, which qualitatively produce similar results, reveals some intriguing findings. First, pruning the image diffusion generator achieves greater overall sparsity due to its larger size, contributing more significantly to the full model's sparsity. Additionally, pruning the text encoder results in a worse CLIP score, while pruning the image diffusion generator leads to a worse FID score. This raises interesting questions about what these metrics are actually measuring. In particular, the lower CLIP score for text encoder pruning may be due to a potential bias, as the text encoder itself is a CLIP model, which could influence how its performance is evaluated.

4.4. Full Model Pruning

First Approach

Given the computational demands of evaluating models, conducting an exhaustive grid search to identify the optimal pruning configuration is impractical. In the first approach we try to establish guiding principles for how to best distribute sparsity given a certain target sparsity for the full model. The experiments shown in Table 1 were conducted and the results are shown in Tables 3, 4, 5. As outlined in the methodology section, certain configurations are deemed invalid due to the significant size disparity between the text and image components.

All experiments used magnitude pruning for both components of the model. Similar experiments were conducted using magnitude pruning with OWL for the CLIP text encoder and produced very similar results as seen in Tables 7, 8, 9 in the appendix.

Tables 3, 4, 5 demonstrate that the model performs better when the majority of the sparsity is allocated to the image component. Notably, the configuration in which 75% of the model's sparsity is concentrated in the image diffusion generator consistently outperforms other splits. This finding aligns with the earlier observation that pruning the image component yields greater sparsity with less decline in performance compared to pruning the text component.

The models where performance deteriorates significantly are highlighted in Table 5 and correspond to sparsities that exceed the drop-off thresholds identified in the individual component pruning experiments. This reinforces the consistency of these pruning thresholds and informs our second approach for determining the optimal full model pruning configuration.

Total Sparsity	75:25	50:50	25:75
20%	20.92	20.89	17.85
30%	244.92	20.8	18.4
40%	N/A	219.97	20.71
50%	N/A	202.84	66.2
60%	N/A	N/A	370.2

Table 3. FID at different allocations of sparsity between text encoder and image diffusion generator

Total Sparsity	75:25	50:50	25:75
20%	0.308	0.314	0.318
30%	0.169	0.307	0.316
40%	N/A	0.176	0.307
50%	N/A	0.181	0.27
60%	N/A	N/A	0.202

Table 4. CLIP Score at different allocations of sparsity between text encoder and image diffusion generator

Total Sparsity	Text:Image Ratio	Text Sparsity	Image Sparsity	FID	CLIP Score
20%	75:25	53%	7%	20.92	0.308
20%	50:50	35%	14%	20.89	0.314
20%	25:75	18%	21%	17.85	0.318
30%	75:25	80%	10%	244.92	0.169
30%	50:50	53%	21%	20.8	0.307
30%	25:75	27%	31%	18.4	0.316
40%	50:50	71%	28%	219.97	0.176
40%	25:75	35%	42%	20.71	0.307
50%	50:50	89%	35%	202.84	0.181
50%	25:75	44%	52%	66.2	0.27
60%	25:75	53%	63%	370.2	0.202

Table 5. Model Performance greatly suffers when component sparsities violate the identified drop-off thresholds

Second Approach

We examine the model pruned to both thresholds established in the individual pruning experiments. Given the possibility that these thresholds might behave differently when both components are pruned simultaneously, we also prune several models to values slightly below the thresholds to determine the optimal sparsity configurations. Starting from the two thresholds, we decrease both the text and image component sparsities in increments of 2.5%, evaluating each of these

configurations. The results are summarized in Table 6, where the unpruned model is provided as a baseline and highlighted in gray. The models were pruned using magnitude pruning for both components of the model. These configurations were also tested using magnitude pruning with OWL for CLIP text encoder and produced very similar results as shown in Table 10 in the appendix

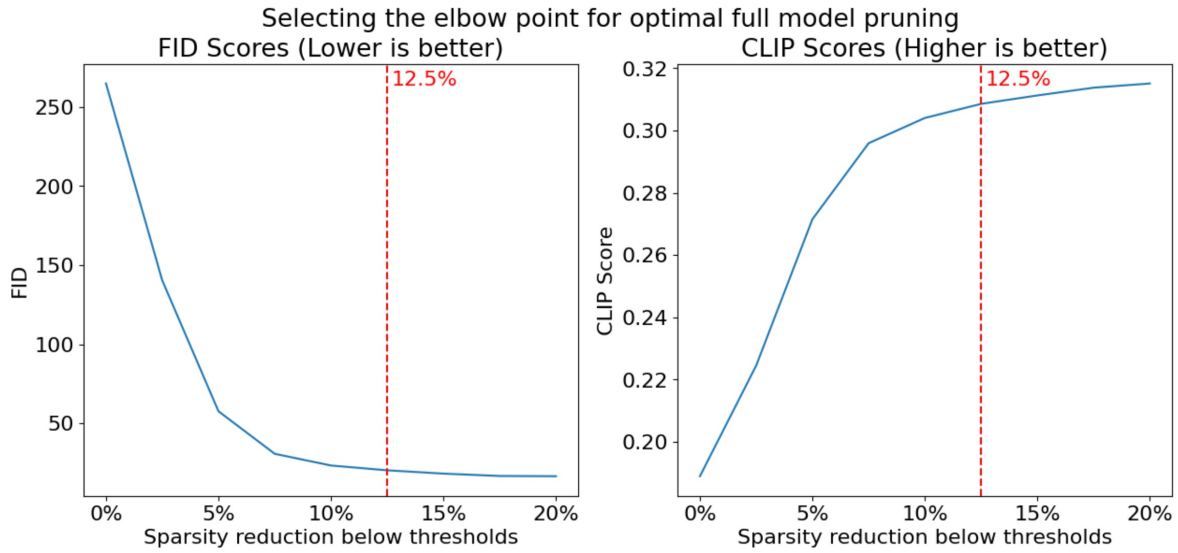

Figure 12. Model Improvement as sparsity is reduced from identified drop-off thresholds

Total Sparsity	Text Sparsity	Image Sparsity	FID	CLIP Score
0%	0%	0%	18.07	0.314
53.5%	62.5%	50.0%	264.87	0.189
51.0%	60.0%	47.5%	140.39	0.224
48.5%	57.5%	45.0%	57.59	0.272
46.0%	55.0%	42.5%	30.69	0.296
43.5%	52.5%	40.0%	23.28	0.304
41.0%	50.0%	37.5%	20.24	0.309
38.5%	47.5%	35.0%	18.15	0.311
36.0%	45.0%	32.5%	16.66	0.314
33.5%	42.5%	30.0%	16.53	0.315

Table 6. Pruning Stable Diffusion 2 to found thresholds

The results presented in Table 6 indicate that pruning both thresholds simultaneously leads to a model performance that is significantly worse than anticipated. While each threshold individually resulted in minimal performance loss, the combined pruning of both components appears to degrade the model's performance substantially. However, as we gradually reduce both sparsities, we observe improvements, which, along with qualitative evaluations of the generated

images, enable us to recommend an optimal pruning configuration for the model.

Specifically, Stable Diffusion 2 can be pruned to 38.5% sparsity by adjusting the text encoder to 47.5% sparsity and the diffusion generator to 35% sparsity. This configuration results in minimal loss of image quality while providing a substantial reduction in model size. A comparison of the generated image quality for the baseline and optimal configurations is illustrated in Figure 13.

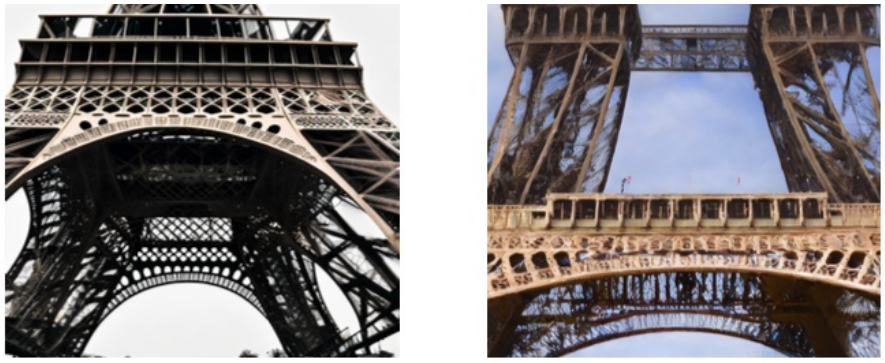

Figure 13. Base Model vs Optimally Pruned Model at 38.5% Sparsity

5. Conclusions

This research presents the first comprehensive study on post-training pruning of text-to-image models, specifically focusing on Stable Diffusion 2. Our findings reveal unexpected behaviors and provide crucial insights into the compression of these complex models. We demonstrate that Stable Diffusion 2 can be effectively pruned to 38.5% sparsity with minimal quality loss, achieving a significant reduction in model size. The optimal pruning configuration, with 47.5% sparsity in the text encoder and 35% in the diffusion generator, maintains image generation quality while substantially reducing computational requirements. This achievement paves the way for the deployment of advanced text-to-image models on resource-constrained devices, potentially broadening accessibility to this technology. Contrary to established trends in language model pruning, we found that simple magnitude pruning outperforms more advanced techniques in this context. Additionally, we uncovered the existence of sharp performance thresholds, suggesting a distinctive information encoding mechanism in text-to-image models. These unexpected results highlight the unique challenges and opportunities in pruning text-to-image models, emphasizing the need for specialized approaches in this domain.

Our work opens up new avenues for future research, including the development of pruning techniques tailored to text-to-image models, further investigation into the information encoding mechanisms of these models, and exploration of potential applications in bias identification and model interpretability.

In conclusion, this study not only provides practical insights for the efficient deployment of text-to-image models but also lays the groundwork for a new subfield at the intersection of model compression and text-to-image generation. As these models continue to grow in complexity and capability, the insights and methods presented here will be crucial in ensuring their widespread accessibility and efficient implementation.

6. Future Work

Our results and analyses suggest several future research directions. Our findings indicate that both Wanda and SparseGPT do not effectively extend to these models, while magnitude pruning—typically regarded as the baseline

method for pruning—emerges as the best-performing approach in our study. This highlights a significant opportunity for further improvement and optimization in the field. Additionally, our work could be applied to more recent models, such as Stable Diffusion 3 or larger text-to-image models, to assess the generalizability of our findings.

A. Other Results

Total Sparsity	75:25	50:50	25:75
20%	20.92	20.9	18.18
30%	290.97	20.81	18.51
40%	0	250.37	20.48
50%	0	226.44	67.93
60%	0	0	370.47

Table 7. FID for Full Model Pruning with OWL

Total Sparsity	75:25	50:50	25:75
20%	0.307	0.314	0.317
30%	0.172	0.306	0.316
40%	0	0.183	0.307
50%	0	0.182	0.269
60%	0	0	0.202

Table 8. CLIP Score for Full Model Pruning with OWL

Total Sparsity	Text:Image Ratio	Text Sparsity	Image Sparsity	FID	CLIP Score
20%	75:25	53%	7%	20.92	0.307
20%	50:50	35%	14%	20.9	0.314
20%	25:75	18%	21%	18.18	0.317
30%	75:25	80%	10%	290.97	0.172
30%	50:50	53%	21%	20.81	0.306
30%	25:75	27%	31%	18.51	0.316
40%	50:50	71%	28%	250.37	0.183
40%	25:75	35%	42%	20.48	0.307
50%	50:50	89%	35%	226.44	0.182
50%	25:75	44%	52%	67.93	0.269
60%	25:75	53%	63%	370.47	0.202

Table 9. Model Performance greatly suffers when component sparsities violate the identified drop-off thresholds

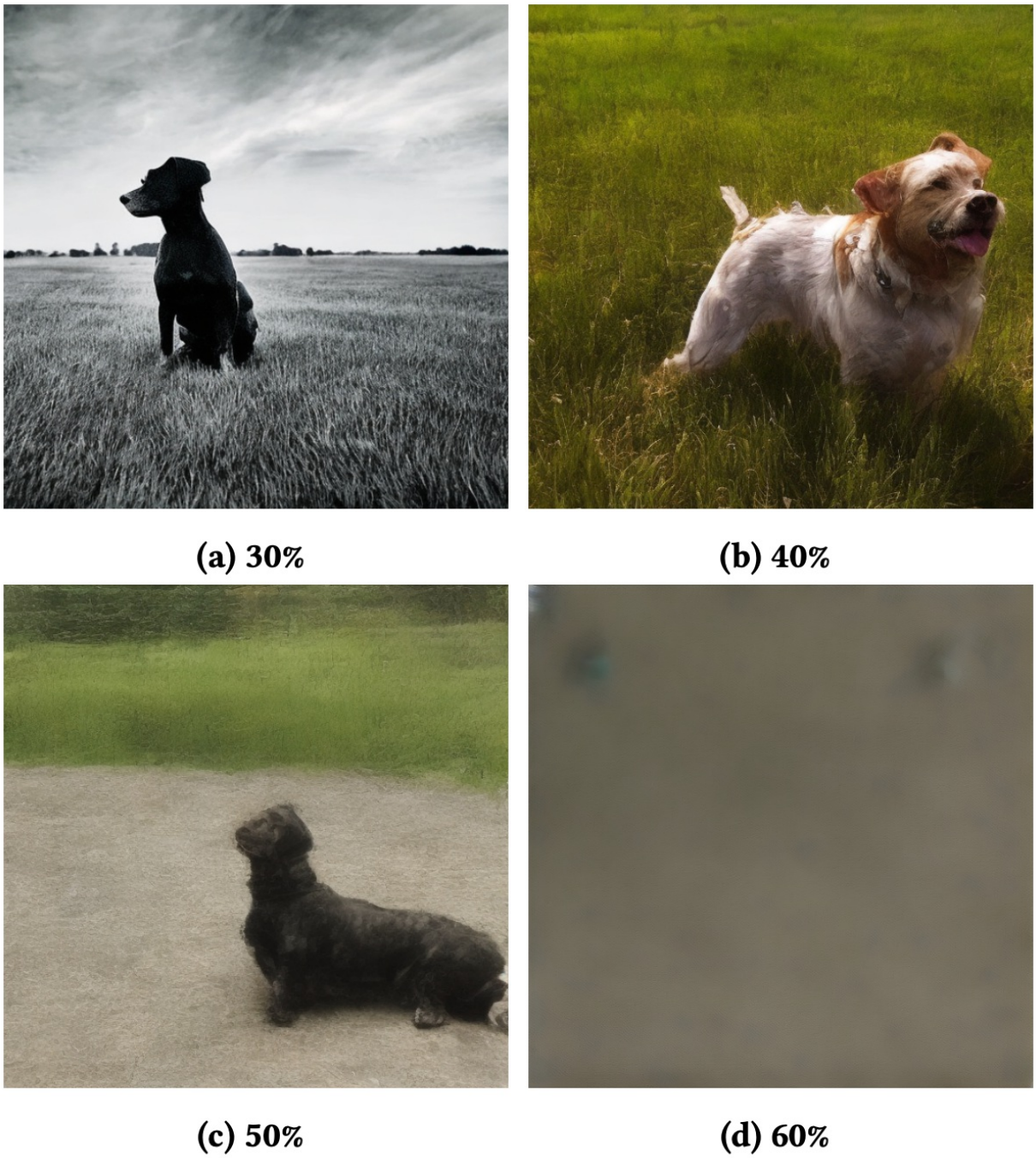

Figure 14. Magnitude Pruning Diffusion Generator causes linear deterioration of quality

Total Sparsity	Text Sparsity	Image Sparsity	FID	CLIP Score
0%	0%	0%	18.07	0.314
53.5%	62.5%	50.0%	241.71	0.193
51.0%	60.0%	47.5%	122.76	0.236
48.5%	57.5%	45.0%	59.12	0.273
46.0%	55.0%	42.5%	33.09	0.294
43.5%	52.5%	40.0%	24.96	0.304
41.0%	50.0%	37.5%	20.75	0.309
38.5%	47.5%	35.0%	17.97	0.311
36.0%	45.0%	32.5%	16.79	0.314
33.5%	42.5%	30.0%	16.45	0.315

Table 10. Pruning Stable Diffusion 2 to found thresholds

Footnotes

[1] We publicly released our code at: https://github.com/samarthramesh/SD2-Pruning

References

1. ^ *Ko HK, Park G, Jeon H, Jo J, Kim J, Seo J (2023). "Large-scale text-to-image generation models for visual artists' creative works". In: Proceedings of the 28th international conference on intelligent user interfaces pp. 919–933.*
2. a, b *Brown TB (2020). "Language models are few-shot learners".arXiv preprint arXiv:2005.14165. arXiv:2005.14165.*
3. ^ *Narayanan D, Shoeybi M, Casper J, LeGresley P, Patwary M, Korthikanti V, Vainbrand D, Kashinkunti P, Bernauer J, Catanzaro B, Phanishayee A, Zaharia M (2021). "Efficient large-scale language model training on GPU clusters using megatron-LM". In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (St. Louis, Missouri) (SC '21). Association for Computing Machinery, New York, NY, USA, Article 58. doi:10.1145/3458817.3476209.*
4. ^ *Yang L, Zhang Z, Song Y, Hong S, Xu R, Zhao Y, Zhang W, Cui B, Yang M-H (2023). "Diffusion Models: A Comprehensive Survey of Methods and Applications". ACM Comput. Surv.. **56** (4): Article 105, 39 pages. doi:10.1145/3626235.*
5. ^ *Prakash P, Ding J, Chen R, Qin X, Shu M, Cui Q, Guo Y, Pan M (2022). "IoT device friendly and communication-efficient federated learning via joint model pruning and quantization". IEEE Internet of Things Journal **9** (15): 13638–13650.*
6. a, b *Novac P-E, Boukli Hacene G, Pegatoquet A, Miramond B, Gripon V (2021). "Quantization and Deployment of Deep Neural Networks on Microcontrollers". Sensors. **21** (9): 2984. doi:10.3390/s21092984. PMID 33922868.*
7. ^ *Hadidi R, Cao J, Xie Y, Asgari B, Krishna T, Kim H (2019). "Characterizing the Deployment of Deep Neural Networks*

on Commercial Edge Devices". 2019 IEEE International Symposium on Workload Characterization (IISWC). 35-48. doi:10.1109/IISWC47752.2019.9041955.

8. ^*Hawks B, Duarte J, Fraser NJ, Pappalardo A, Tran N, Umuroglu Y (2021). "Ps and qs: Quantization-aware pruning for efficient low latency neural network inference". Frontiers in Artificial Intelligence. **4**: 676564.*
9. ^*Liang T, Glossner J, Wang L, Shi S, Zhang X (2021). "Pruning and quantization for deep neural network acceleration: A survey". Neurocomputing. **461**: 370--403.*
10. ^*Niu W, Ma X, Lin S, Wang S, Qian X, Lin X, Wang Y, Ren B (2020). "Patdnn: Achieving real-time dnn execution on mobile devices with pattern-based weight pruning". In Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems. 2020: 907-922.*
11. ^*Gray RM, Neuhoff DL (1998). "Quantization". IEEE Transactions on Information Theory. **44** (6): 2325–2383. doi:10.1109/18.720541.*
12. a, b*LeCun Y, Denker J, Solla S. Optimal Brain Damage. In: Touretzky D, editor.Advances in Neural Information Processing Systems. Vol. 2. Morgan-Kaufmann; 1989. Available from: https://proceedings.neurips.cc/paper_files/paper/1989/file/6c9882bbac1c7093bd25041881277658-Paper.pdf.*
13. ^*Hassibi B, Stork DG, Wolff GJ (1993). "Optimal brain surgeon and general network pruning". In1EEE international conference on neural networks. IEEE, 1993: 293–299.*
14. a, b*Han S, Pool J, Tran J, Dally WJ (2015). "Learning both Weights and Connections for Efficient Neural Network". In: NIPS. pp. 1135–1143.*
15. ^*Lopes A, Pereira dos Santos F, de Oliveira D, Schiezaro M, Pedrini H (2024). "Computer Vision Model Compression Techniques for Embedded Systems:A Survey". Computers & Graphics. **123**: 104015. doi:10.1016/j.cag.2024.104015. Link to article.*
16. a, b, c*Frantar E, Alistarh D (2023). "SparseGPT: Massive Language Models Can be Accurately Pruned in One-Shot". In ICML (Proceedings of Machine Learning Research, Vol. **202**). PMLR, pp. 10323–10337.*
17. a, b*Sun M, Liu Z, Bair A, Kolter JZ (2024). "A Simple and Effective Pruning Approach for Large Language Models". In: ICLR. OpenReview.net.*
18. a, b*Rombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2022). "High-Resolution Image Synthesis with Latent Diffusion Models". In: CVPR. IEEE, 10674–10685.*
19. ^*Goodfellow IJ, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville AC, Bengio Y (2020). "Generative adversarial networks". Commun. ACM. **63** (11): 139–144.*
20. ^*Brock A (2018). "Large Scale GAN Training for High Fidelity Natural Image Synthesis".arXiv preprint arXiv:1809.11096. Available from: https://arxiv.org/abs/1809.11096.*
21. ^*Sohl-Dickstein J, Weiss EA, Maheswaranathan N, Ganguli S (2015). "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In ICML (JMLR Workshop and Conference Proceedings, Vol. **37**). JMLR.org, 2256–2265.*
22. ^*Karras T, Laine S, Aila T (2019). "A style-based generator architecture for generative adversarial networks". In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition pp. 4401–4410.*

23. ^Ho J, Jain A, Abbeel P (2020). "Denoising Diffusion Probabilistic Models". In:NeurIPS.
24. ^Song Y, Sohl-Dickstein J, Kingma DP, Kumar A, Ermon S, Poole B (2020). "Score-based generative modeling through stochastic differential equations". arXiv preprint arXiv:2011.13456 (2020).
25. ^Dhariwal P, Nichol A (2021). "Diffusion models beat gans on image synthesis".Advances in neural information processing systems. **34**: 8780–8794.
26. a, bRombach R, Blattmann A, Lorenz D, Esser P, Ommer B (2021). "High-Resolution Image Synthesis with Latent Diffusion Models". arXiv. arXiv:2112.10752 [cs.CV].
27. a, bEsser P, Kulal S, Blattmann A, Entezari R, Müller J, Saini H, Levi Y, Lorenz D, Sauer A, Boesel F, Podell D, Dockhorn T, English Z, Rombach R. Scaling Rectified Flow Transformers for High-Resolution Image Synthesis. In: Proceedings of the International Conference on Machine Learning (ICML) 2024. OpenReview.net.
28. ^Radford A, Kim JW, Hallacy C, Ramesh A, Goh G, Agarwal S, Sastry G, Askell A, Mishkin P, Clark J, Krueger G, Sutskever I. Learning Transferable Visual Models From Natural Language Supervision. In: ICML. Proceedings of Machine Learning Research. 2021; **139**:8748-8763.
29. ^Ronneberger O, Fischer P, Brox T (2015). "U-net: Convolutional networks for biomedical image segmentation". In: Medical image computing and computer-assisted intervention--MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18. Springer; 2015. p. 234–241.
30. ^Ramesh A, Dhariwal P, Nichol A, Chu C, Chen M (2022). "Hierarchical text-conditional image generation with clip latents". arXiv preprint arXiv:2204.06125. **1** (2): 3.
31. ^Dettmers T, Lewis M, Belkada Y, Zettlemoyer L (2022). "Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale". Advances in Neural Information Processing Systems **35**: 30318–30332.
32. ^Kovaleva O, Kulshreshtha S, Rogers A, Rumshisky A (2021). "BERT busters: Outlier dimensions that disrupt transformers". arXiv preprint arXiv:2105.06990. arXiv:2105.06990.
33. ^Puccetti G, Rogers A, Drozd A, Dell'Orletta F (2022). "Outliers dimensions that disrupt transformers are driven by frequency". arXiv preprint arXiv:2205.11380.
34. ^Timkey W, Van Schijndel M (2021). "All bark and no bite: Rogue dimensions in transformer language models obscure representational quality". arXiv preprint arXiv:2109.04404. Available from: https://arxiv.org/abs/2109.04404.
35. ^Lu Y, Wu Y, Zhang Z, Hsieh C, Wang Y, Jia Y, Li G, Jaiswal AK, Pechenizkiy M, Liang Y, Bendersky M, Wang Z, Liu S (2024). "Outlier Weighed Layerwise Sparsity (OWL): A Missing Secret Sauce for Pruning LLMs to High Sparsity". In: Proceedings of the ICML. OpenReview.net.
36. ^Fang G, Ma X, Wang X (2023). "Structural Pruning for Diffusion Models". In:NeurIPS.
37. ^Castells T, Song HK, Kim BK, Choi S (2024). "LD-Pruner: Efficient Pruning of Latent Diffusion Models using Task-Agnostic Insights". arXiv:2404.11936 [cs]. Available from:http://arxiv.org/abs/2404.11936.
38. a, bLin T-Y, Maire M, Belongie SJ, Hays J, Perona P, Ramanan D, Dollár P, Zitnick CL (2014). "Microsoft COCO: Common Objects in Context". In: ECCV (5). Lecture Notes in Computer Science, Vol. **8693**. Springer, pp. 740–755.
39. a, bHeusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S (2017). "GANs Trained by a Two Time-Scale

Update Rule Converge to a Local Nash Equilibrium". In NIPS. pp. 6626–6637.

40. [a, b]*Hessel J, Holtzman A, Forbes M, Le Bras R, Choi Y (2021). "CLIPScore: A Reference-free Evaluation Metric for Image Captioning". In: EMNLP (1). Association for Computational Linguistics; 2021. p. 7514-7528.*