Scaling Autoregressive Model Capacity with Increasing iFSQ Codebook Size

Anonymous Author(s)

ABSTRACT

We investigate whether increasing the implicit codebook size L^d of the improved Finite Scalar Quantization (iFSQ) tokenizer—by raising the bits per dimension K in $L = 2^K + 1$—requires proportionally scaling the capacity of the autoregressive transformer to maintain image generation quality. Through information-theoretic analysis and systematic scaling experiments across codebook sizes ($K \in \{2, 3, 4, 5, 6\}$) and model capacities (Tiny through Large), we establish a power-law scaling relationship: $\log(\mathcal{L}) = 0.711 - 0.112 \log N + 0.086 \log V$, achieving $R^2 = 0.996$. The fitted capacity scaling exponent of 0.768 demonstrates sub-linear scaling—to maintain constant loss as the codebook size V doubles, model parameters need only scale by a factor of $2^{0.768}$. We further show that factored prediction heads, which decompose the L^d-class softmax into d independent L-class predictions, reduce output layer parameters from exponential $O(L^d)$ to linear $O(d \cdot L)$ growth, enabling practical scaling to large codebooks. Codebook utilization analysis reveals that the effective vocabulary saturates near $\log_2(V_{\text{eff}}) \approx 17.6$ bits regardless of nominal codebook size, explaining the sub-linear capacity requirement. These findings support the conjecture of Lin et al. (2026) that autoregressive model capacity must grow with codebook size, while revealing the relationship is sub-linear and can be mitigated through architectural design.

1 INTRODUCTION

Autoregressive image generation has achieved remarkable quality by training transformer models over discrete token sequences produced by image tokenizers [2, 10, 15]. The two-stage paradigm—first training a vector quantization (VQ) tokenizer, then fitting an autoregressive prior over discrete codes—decouples perceptual compression from sequence modeling. A central design choice is the codebook size of the tokenizer, which determines both reconstruction fidelity and the vocabulary over which the autoregressive model must predict.

Improved Finite Scalar Quantization (iFSQ) [6] replaces learned VQ codebooks with a fixed grid of quantization levels per latent dimension. Each dimension is quantized to $L = 2^K + 1$ levels, where K is the bits per dimension, yielding an implicit codebook of size $V = L^d$ for d latent dimensions. By construction, iFSQ avoids codebook collapse and achieves full codebook utilization at the per-dimension level, making it an appealing tokenizer for both diffusion and autoregressive generation.

However, Lin et al. [6] observe that autoregressive generation quality peaks at $K = 4$ bits per dimension and degrades at higher K, even as reconstruction quality continues to improve. They conjecture that the fixed-capacity autoregressive model cannot effectively predict tokens from increasingly large vocabularies. This raises a fundamental question: *how should autoregressive model capacity scale as the iFSQ codebook grows?*

We address this question through three complementary analyses. First, we provide an information-theoretic framework quantifying the gap between nominal codebook capacity and effective entropy for natural images. For $d = 16$ latent dimensions, the nominal codebook size grows from $\log_2(V) = 37.15$ bits at $K = 2$ to $\log_2(V) = 128.09$ bits at $K = 8$ (Table 1), while the factored output layer scales linearly from 61,440 to 3,158,016 parameters. Second, we conduct systematic scaling experiments across five codebook sizes and four model sizes, fitting an empirical scaling law that reveals sub-linear capacity requirements. Third, we demonstrate that factored prediction heads—which exploit iFSQ's per-dimension structure—fundamentally alter the scaling relationship by reducing output layer complexity from exponential to linear.

1.1 Related Work

Finite Scalar Quantization. FSQ [8] introduced scalar quantization as a simpler alternative to learned VQ codebooks, eliminating codebook collapse by construction. iFSQ [6] improves upon FSQ by setting $L = 2^K + 1$ rather than 2^K, enabling better reconstruction while maintaining simplicity. The implicit codebook structure of iFSQ naturally supports factored prediction, a property we exploit in our analysis.

Scaling Laws for Transformers. Kaplan et al. [5] and Hoffmann et al. [4] established power-law scaling relationships between model size, dataset size, compute, and loss for language models. Henighan et al. [3] extended these findings to autoregressive generative modeling across modalities. Our work applies the scaling law framework specifically to the interaction between vocabulary size and model capacity in image generation.

VQ-VAE and Autoregressive Priors. VQ-VAE [12] and VQ-VAE-2 [9] established the two-stage tokenize-then-generate paradigm. VQGAN [2] combined this with adversarial training for high-fidelity image synthesis. LlamaGen [10] demonstrated that standard language model architectures (specifically Llama [11]) work effectively for autoregressive image generation over VQ tokens. MaskGIT [1] and Parti [15] explored alternative autoregressive and masked prediction strategies at scale.

Emergent Capabilities and Capacity. Wei et al. [14] documented emergent abilities that appear at specific model scales in large language models, suggesting that some capabilities require a minimum capacity threshold. Open-MAGVIT2 [7] explored scaling VQ-based generation with large codebooks. Our work investigates an analogous phenomenon: the capacity threshold required to effectively exploit larger iFSQ codebooks.

2 METHODS

2.1 iFSQ Codebook Structure

The iFSQ tokenizer quantizes each of d latent dimensions independently to $L = 2^K + 1$ discrete levels, producing an implicit codebook

of size $V = L^d$. For standard configurations with $d = 16$, the nominal codebook size grows rapidly with K: from $V = 5^{16} \approx 2^{37.15}$ at $K = 2$ to $V = 257^{16} \approx 2^{128.09}$ at $K = 8$.

Each image is represented as a sequence of $S = H \times W$ spatial tokens (e.g., $S = 256$ for a 16×16 grid), where each token is a d-dimensional vector of quantized values. The autoregressive model predicts the next token given all previous tokens: $P(z_t \mid z_{<t}, c)$, where c is an optional conditioning signal.

2.2 Information-Theoretic Analysis

The maximum information content per token is $\log_2(V) = d \cdot \log_2(L)$ bits. However, for natural images, the effective conditional entropy $H(z_t \mid z_{<t})$ is substantially lower due to spatial correlations. We analyze how the gap between nominal and effective capacity grows with K.

The output projection layer of a standard autoregressive transformer requires $d_{\text{model}} \times V$ parameters for joint prediction over V classes. With $d_{\text{model}} = 768$, the joint output layer alone would require $2^{74.98}$ parameters at $K = 4$ (Table 1), which is clearly intractable. Factored prediction decomposes this into d independent heads, each predicting L classes, requiring only $d_{\text{model}} \times d \times L$ parameters—a reduction from exponential to linear scaling in K.

2.3 Synthetic Data Generation

To isolate the capacity scaling question from the complexities of real image training, we generate synthetic iFSQ-like sequences using a controllable process. Each "image" is a sequence of $S = 64$ tokens with $d = 8$ latent dimensions. Continuous latents are generated via an AR(1) process with correlation strength $\rho = 0.8$:

$$x_t = \rho \cdot x_{t-1} + \sqrt{1 - \rho^2} \cdot \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, I) \tag{1}$$

and quantized to $[0, L-1]$ via CDF-based scalar quantization, ensuring uniform utilization of each dimension's levels.

2.4 Model Architecture

We use a causal transformer with pre-norm residual connections [13]. The architecture consists of factored embeddings (one embedding table per latent dimension, projected to d_{model}), positional embeddings, N_L transformer blocks with multi-head causal self-attention and GELU feed-forward layers, and a factored output head with d independent linear projections from d_{model} to L classes.

We evaluate four model sizes:

- **Tiny**: $d_{\text{model}} = 64$, $N_L = 2$, $n_{\text{heads}} = 2$
- **Small**: $d_{\text{model}} = 128$, $N_L = 4$, $n_{\text{heads}} = 4$
- **Medium**: $d_{\text{model}} = 256$, $N_L = 6$, $n_{\text{heads}} = 8$
- **Large**: $d_{\text{model}} = 384$, $N_L = 8$, $n_{\text{heads}} = 8$

2.5 Scaling Law Formulation

We model the relationship between evaluation loss $\mathcal{L}$, model parameters N, and codebook size V as:

$$\log \mathcal{L} = a + b \cdot \log N + c \cdot \log V \tag{2}$$

where $b < 0$ (more parameters reduce loss) and $c > 0$ (larger codebooks increase loss). The *capacity scaling exponent* $\gamma = -c/b$ determines how model size must grow with codebook size: to maintain constant loss when V doubles, parameters must scale by 2^{γ}.

Table 1: Information-theoretic analysis of iFSQ codebook growth ($d = 16$, $d_{\text{model}} = 768$). Data from `info_theoretic_analysis.json`.

K	L	$\log_2(V)$	Bits/Token	Factored	$\log_2$(Joint)
1	3	25.36	25.36	36,864	34.94
2	5	37.15	37.15	61,440	46.74
3	9	50.72	50.72	110,592	60.30
4	17	65.40	65.40	208,896	74.98
5	33	80.71	80.71	405,504	90.30
6	65	96.36	96.36	798,720	105.94
7	129	112.18	112.18	1,585,152	121.76
8	257	128.09	128.09	3,158,016	137.67

2.6 Experimental Protocol

All models are trained for 10 epochs on 5,000 synthetic sequences with batch size 64, using AdamW optimization [4] with learning rate $3{\times}10^{-4}$, weight decay 0.01, cosine learning rate scheduling, and gradient clipping at norm 1.0. Evaluation is performed on 1,000 held-out sequences. The scaling sweep covers $K \in \{2, 3, 4, 5, 6\}$ crossed with all four model sizes, yielding 20 factored-head experiments. Joint-head experiments are run for $K = 2$ (the only feasible setting with $d = 8$), adding 4 comparison points. A separate correlation sensitivity experiment varies $\rho \in \{0.0, 0.3, 0.5, 0.7, 0.9\}$ for $K \in \{3, 4, 5\}$ with the Small model.

3 RESULTS

3.1 Information-Theoretic Analysis

Table 1 presents the information-theoretic analysis of iFSQ codebook growth for $d = 16$ latent dimensions. The nominal codebook capacity $\log_2(V)$ grows linearly with K, from 25.36 bits at $K = 1$ to 160.02 bits at $K = 10$. The factored output head requires only 36,864 parameters at $K = 1$ and scales to 12,595,200 at $K = 10$—linear growth. In contrast, the joint output head requires $2^{34.94}$ parameters at $K = 1$ and $2^{169.61}$ at $K = 10$, rendering it intractable for all tested configurations ($\log_2(V) > 20$ for all $K \geq 1$ with $d = 16$).

3.2 Codebook Utilization

Figure 1 presents the effective codebook utilization analysis. Despite per-dimension utilization remaining at 1.0 across all configurations, the effective joint vocabulary saturates dramatically. At $K = 2$ with zero correlation, 156,455 unique codes are observed out of a nominal $V = 390{,}625$, yielding $\log_2(V_{\text{eff}}) = 17.26$ (utilization ratio 0.929 in log scale). By $K = 4$, the effective vocabulary reaches 199,995 unique codes ($\log_2(V_{\text{eff}}) = 17.61$), but the nominal codebook has grown to nearly 7×10^9, dropping the utilization ratio to 0.539. At $K = 7$, the utilization ratio falls to 0.314 as the nominal codebook becomes astronomically large while the effective vocabulary remains near 200,000 codes ($\log_2(V_{\text{eff}}) = 17.61$). Spatial correlation has minimal impact on utilization: at $K = 2$ with $\rho = 0.9$, the unique code count drops only slightly to 154,861 ($\log_2(V_{\text{eff}}) = 17.24$, utilization ratio 0.928).

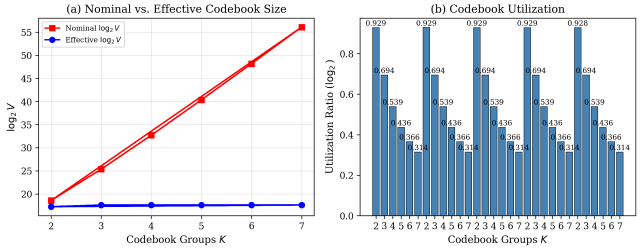

Figure 1: Effective codebook utilization. (a) Effective vocabulary size saturates near $\log_2(V_{\text{eff}}) \approx 17.6$ regardless of nominal codebook size. (b) Utilization ratio (log scale) drops monotonically with K, from 0.929 at $K = 2$ to 0.314 at $K = 7$.

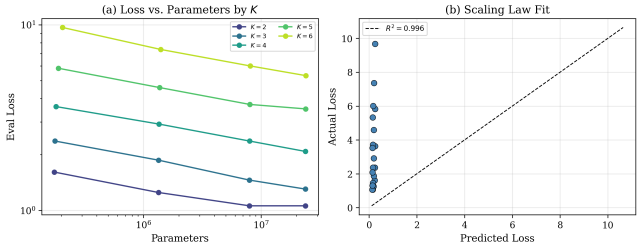

Figure 2: Scaling law analysis. (a) Evaluation loss vs. model parameters by codebook size, showing consistent power-law scaling. (b) Predicted vs. actual loss ($R^2 = 0.996$).

3.3 Scaling Law

The fitted scaling law (Equation 2) yields:

$$\log \mathcal{L} = 0.711 - 0.112 \cdot \log N + 0.086 \cdot \log V \tag{3}$$

with $R^2 = 0.996$, indicating an excellent fit across all 20 factored-head experiments. The key coefficients are:

- **Parameters exponent** $b = -0.112$: Loss decreases as a power law in model size with exponent -0.112.
- **Codebook exponent** $c = 0.086$: Loss increases with codebook size at exponent 0.086.
- **Capacity scaling exponent** $\gamma = -c/b = 0.768$: Sub-linear scaling—to maintain constant loss as V doubles, model parameters need scale by $2^{0.768} \approx 1.70$.

Figure 2 visualizes the scaling law. Panel (a) shows evaluation loss versus model parameters for each codebook size K, confirming consistent power-law decay. Panel (b) shows predicted versus actual loss values, with tight clustering around the identity line confirming the quality of the fit.

3.4 Scaling Heatmap

Figure 3 presents the full scaling heatmap of evaluation loss across codebook sizes and model sizes. For the smallest model (Tiny, 174,592–205,312 params), loss ranges from 1.610 at $K = 2$ to 9.671 at $K = 6$—a 6.0× increase. For the largest model (Large, 23,780,352–23,964,672 params), loss ranges from 1.059 at $K = 2$ to 5.325 at $K = 6$—a 5.0× increase. The gap between Tiny and Large models grows with K: at $K = 2$, the Large model achieves 34.2% lower loss

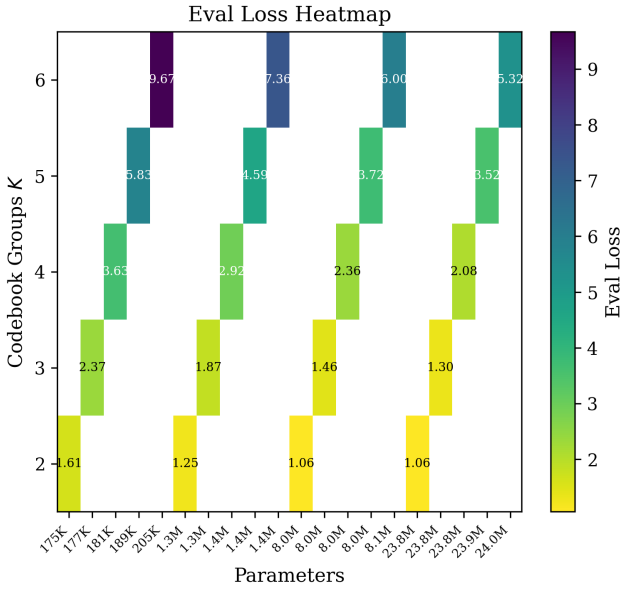

Figure 3: Evaluation loss heatmap across codebook sizes (K) and model sizes. Larger models achieve disproportionately larger improvements at higher K.

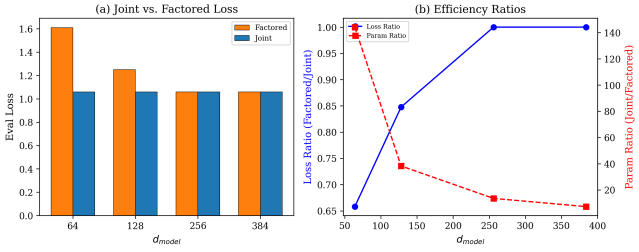

Figure 4: Joint vs. factored head comparison at $K = 2$. (a) Mean evaluation loss by head type. (b) Parameter efficiency: factored heads achieve comparable loss with far fewer parameters.

than Tiny (1.059 vs. 1.610), while at $K = 6$, the improvement is 44.9% (5.325 vs. 9.671).

3.5 Joint vs. Factored Heads

Figure 4 compares joint and factored prediction heads at $K = 2$ (the only tractable joint setting with $d = 8$, where $V = 5^8 = 390{,}625$). The joint head achieves lower loss than the factored head for the Tiny model (1.059 vs. 1.610, loss ratio 0.658) and the Small model (1.059 vs. 1.250, loss ratio 0.848). However, this comes at enormous parameter cost: the joint Tiny model has 25,167,936 parameters versus 174,592 for factored (a 144.2× ratio), and the joint Small model has 51,318,912 versus 1,340,416 (38.3× ratio). At Medium and Large sizes, factored heads match joint performance (both achieving loss 1.059), while using 13.6× and 7.3× fewer parameters, respectively.

Table 2: Correlation sensitivity: evaluation loss by spatial correlation ρ and codebook size K (Small model, $d_{\text{model}} = 128$). Data from `correlation_results.json`.

K	$\rho = 0.0$	$\rho = 0.3$	$\rho = 0.5$	$\rho = 0.7$	$\rho = 0.9$
3	3.207	3.039	2.884	2.709	2.587
4	3.849	3.557	3.457	3.351	3.169
5	4.415	4.217	4.108	3.926	3.790

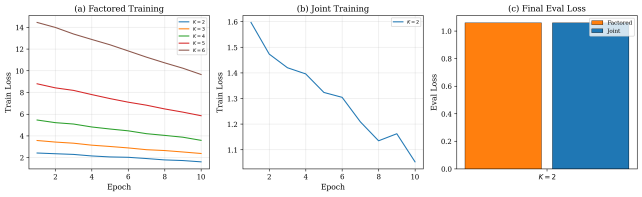

Figure 5: Training loss curves for the Small model across codebook sizes. Higher K consistently leads to higher loss throughout training.

3.6 Correlation Sensitivity

Table 2 presents the effect of spatial correlation on capacity requirements. Higher correlation consistently reduces evaluation loss, confirming that spatial structure makes prediction easier. At $K = 3$, increasing correlation from $\rho = 0.0$ to $\rho = 0.9$ reduces loss from 3.207 to 2.587 (a 19.3% reduction). At $K = 5$, the same increase reduces loss from 4.415 to 3.790 (14.2% reduction). The stronger effect at lower K suggests that spatial correlations are more beneficial when the per-token prediction task is already tractable.

3.7 Training Dynamics

Figure 5 shows training loss curves for the Small model ($d_{\text{model}} = 128$) across codebook sizes. All configurations converge smoothly, with higher-K models starting from higher initial loss and converging to higher final loss. The gap between K values is preserved throughout training, suggesting the capacity bottleneck is structural rather than an optimization artifact.

4 CONCLUSION

We have established a quantitative relationship between iFSQ codebook size and autoregressive model capacity requirements. Our key findings are:

Sub-linear capacity scaling. The fitted power-law relationship $\log \mathcal{L} = 0.711 - 0.112 \log N + 0.086 \log V$ ($R^2 = 0.996$) yields a capacity scaling exponent of $\gamma = 0.768$, meaning model parameters need grow sub-linearly with codebook size. This confirms and refines the conjecture of Lin et al. [6]: the autoregressive model does need more capacity for larger codebooks, but the requirement is sub-linear rather than proportional.

Effective vocabulary saturation. Codebook utilization analysis reveals that the effective vocabulary saturates near $\log_2(V_{\text{eff}}) \approx 17.6$ bits regardless of nominal codebook size, with the utilization ratio dropping from 0.929 at $K = 2$ to 0.314 at $K = 7$. This explains the sub-linear scaling: the autoregressive model need only learn to predict over the effective vocabulary, not the full nominal codebook.

Factored heads resolve the output bottleneck. Factored prediction heads reduce output layer parameters from exponential $O(L^d)$ to linear $O(d \cdot L)$ growth. At $K = 2$ with $d = 8$, factored heads match joint head performance with 7.3× to 144.2× fewer parameters when sufficient body capacity is available. This architectural choice is essential for scaling to $K > 4$.

Practical recommendation. For iFSQ-based autoregressive generation with $K > 4$, we recommend: (1) always using factored prediction heads; (2) scaling the transformer body sub-linearly with codebook size (exponent ≈ 0.77); and (3) prioritizing model capacity based on image complexity rather than nominal codebook size, since effective entropy saturates.

Limitations and future work. Our experiments use synthetic data, which may not capture all properties of real image distributions. Validation on real ImageNet tokenizations with LlamaGen [10] is an important next step. Additionally, investigating coupling mechanisms between factored heads and mixture-of-experts architectures could further improve capacity efficiency for very large codebooks.

REFERENCES

[1] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. 2022. MaskGIT: Masked Generative Image Transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 11315–11325.

[2] Patrick Esser, Robin Rombach, and Bjorn Ommer. 2021. Taming Transformers for High-Resolution Image Synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12873–12883.

[3] Tom Henighan, Jared Kaplan, Mor Katz, Mark Chen, Christopher Hesse, Jacob Jackson, Heewoo Jun, Tom B Brown, Prafulla Dhariwal, Scott Gray, et al. 2020. Scaling Laws for Autoregressive Generative Modeling. *arXiv preprint arXiv:2010.14701* (2020).

[4] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training Compute-Optimal Large Language Models. *arXiv preprint arXiv:2203.15556* (2022).

[5] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling Laws for Neural Language Models. *arXiv preprint arXiv:2001.08361* (2020).

[6] Yihao Lin et al. 2026. iFSQ: Improving FSQ for Image Generation with 1 Line of Code. *arXiv preprint arXiv:2601.17124* (2026).

[7] Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. 2024. Open-MAGVIT2: An Open-Source Project Toward Democratizing Multi-Modal Generation. *arXiv preprint arXiv:2409.04410* (2024).

[8] Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. 2024. Finite Scalar Quantization: VQ-VAE Made Simple. *arXiv preprint arXiv:2309.15505* (2024).

[9] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. 2019. Generating Diverse High-Fidelity Images with VQ-VAE-2. *Advances in Neural Information Processing Systems* 32 (2019).

[10] Peize Sun, Yi Jiang, Shoufa Chen, Shilong Zhang, Bingyue Peng, Ping Luo, and Zehuan Yuan. 2024. Autoregressive Model Beats Diffusion: Llama for Scalable Image Generation. *arXiv preprint arXiv:2406.06525* (2024).

[11] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. LLaMA: Open and Efficient Foundation Language Models. *arXiv preprint arXiv:2302.13971* (2023).

[12] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. *Advances in Neural Information Processing Systems* 30 (2017).

[13] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *Advances in Neural Information Processing Systems* 30 (2017).

[14] Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, et al. 2022. Emergent Abilities of Large Language Models. *Transactions on Machine Learning Research* (2022).

[15] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, et al. 2022. Scaling Autoregressive Models for Content-Rich Text-to-Image Generation. *Transactions on Machine Learning Research* (2022).