Data-Free Transformer Quantization Using Parameter-Space Symmetry

Anonymous Authors[1]

Abstract

Transformer models have seen widespread use in many learning tasks but incur large memory and compute costs, limiting their deployability. Post-Training Quantization (PTQ) is a promising solution but can lead to significant performance degradation. Many PTQ methods estimate weight and activation distributions with calibration data to account for outliers and maintain quantized performance. We propose a data-free approach to improve quantization by exploiting parameter space symmetries. We address outliers and high variability in weights by finding a transformation of the model weights that minimizes quantization error variance. Our approach is light-weight, data-free, and can be integrated as a pre-processing step within other PTQ methods. We evaluate our approach by testing quantized large language models on several benchmark tasks.

1. Introduction

Transformer models (Vaswani et al., 2023) have found widespread success as generative models for language modeling and computer vision tasks. Transformers have become increasingly complex incurring large computational and memory storage costs far beyond other models, limiting their usability. The highest performing models have hundreds of billions of parameters (Radford et al., 2019; Zhang et al., 2022) requiring immense training time and massive GPU memory. Even inference on pre-trained models can be prohibitively slow and exceed memory capacity of resource constrained systems. Effective model compression is essential for addressing these limitations.

Many model compression methods such as model-pruning (Zhu et al., 2024) and low-bit quantization (Chen et al., 2024; Ma et al., 2024) require re-training which is infeasible for models with billions of parameters. Post-training quantization (PTQ) which compresses models without re-training is a promising solution but can result in significant performance degradation. Many PTQ methods utilize calibration data and specialized heuristics to preserve model performance (Bondarenko et al., 2021; Nagel et al., 2020). This requires access to high-quality calibration sets and can incur additional overhead for inference of the quantized model. Data-free methods for improving quantization performance have been proposed for MLPs and CNNs (Meller et al., 2019; Nagel et al., 2019) but to our knowledge there are no similar methods for transformers.

In this paper, we develop a data-free method for improving post-training quantization of transformers by leveraging the symmetry of attention weights. Instead of designing a new quantization process, we provide a pre-quantization algorithm which finds equivalent weight configurations which are less sensitive to quantization. An equivalent weight configuration is a transformation of the weights which does not change the layer output. Our approach works by finding a linear transformation of the weights which minimizes the expected quantization error variance. This results in a new set of weights which when quantized results in lower quantization error during inference. There are several advantages to this strategy. First, we operate directly on the weights without any forward passes through the model. Second, our method is a pre-processing step which is compatible with any quantization algorithm allowing it to be stacked with existing techniques. This allows our method to be very lightweight, needing only enough memory for each layer's weights individually, while also being highly flexible and fast.

Our contributions include:

- A closed-form approximation of quantization error variance in attention.
- An optimization algorithm for finding optimal weight transformations.
- Empirical evaluation of our method showing its impact on simple linear quantization.

2. Related Work

Quantization of large language models (LLMs) Quantization reduces the numerical precision of neural network parameters to decrease model size and accelerate inference. This is essential for deploying LLMs efficiently across various hardware platforms. Common quantization techniques

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

include quantization-aware training (QAT) and post-training quantization (PTQ) (Nagel et al., 2021; Zhu et al., 2024). QAT simulates quantization during training and adjusts model parameters to minimize quantization-induced error (Jacob et al., 2018; Esser et al., 2019). PTQ methods directly quantize pre-trained models. PTQ techniques include analytical methods that adjust weight distributions, such as range equalization and bias correction, enabling accurate quantization without access to training data (Nagel et al., 2019; Meller et al., 2019). Other PTQ approaches optimize quantization parameters on small calibration sets (Nagel et al., 2020; Hubara et al., 2021; Li et al., 2021). Recently, PTQ has become prominent for quantizing transformers and large language models (Frantar et al., 2022; Yao et al., 2022; Xiao et al., 2023; Dettmers et al., 2022). Our work follows the post-training quantization paradigm, aiming to further reduce quantization-induced accuracy loss through optimized parameter symmetry transformations.

Using symmetry in quantization Neural networks often have parameter space symmetries, meaning certain transformations of their parameters leave the network's loss unchanged. Examples include the scaling symmetry in networks with ReLU or linear activations (Badrinarayanan et al., 2015), and permutation symmetry among neurons within a hidden layer (Hecht-Nielsen, 1990). Several works have explicitly used such weight transformations to reduce quantization error. A common strategy is to exploit scale invariances to adjust the range of weights or activations before quantization. For example, Nagel et al. (2019) and Meller et al. (2019) propose equalizing weight ranges across layers in ReLU-based networks using the scaling symmetry. (Xiao et al., 2023) improve speed and reduce memory during inference for linear operations, defined as computing the product of activations (output from previous computations) and weights, by applying a loss-invariant scaling on both parts before quantization. While this transformation is defined jointly on parameters and activations, it can be expressed as a parameter symmetry when the activation is the output of a linear operation. Similarly, Kim et al. (2024) scales activation and weights in CNN-transformer hybrid architectures to align parameter distributions with hardware-friendly quantization constraints, thereby improving inference efficiency. Our approach extends these ideas by considering the full general linear group, optimizing over a broader class of symmetry transformations to achieve superior quantization accuracy.

Optimization in transformer model level sets Recent works have also explored optimization over the loss level sets in transformers for applications other than quantization. This optimization is often done on symmetry group orbits, leveraging the general linear group symmetry in self-attention layers. For example, Zhang et al. (2025) improves model fusion by minimizing the distance between two self-attentions without affecting their loss. Their method first finds an optimal rotation of key and query matrices, followed by an optimal scaling. Similarly, Wu et al. (2025) accelerates the training of transformers by searching in the loss level set for points better suited for optimization. We also optimize over the symmetry group orbits of transformer models, but with the specific goal of finding transformations that minimize accuracy loss in quantization.

3. Background

3.1. Transformer Attention

A standard transformer layer consists of two main modules: a multi-head attention(MHA) module and a multi-layer perceptron(MLP). In this work we focus on improving the quantization of the attention module. The attention module has four weight matrices W_q, W_k, W_V, W_O. For a given transformer layer with input $x \in \mathbb{R}^{n \times d}$ the attention scores are computed as:

$$A = xW_qW_k^Tx^T \tag{1}$$

A softmax is applied after to normalize the scores and the final layer output is computed:

$$\text{MHA}(x) = \text{softmax}\left(\frac{A}{\sqrt{d}}\right)xW_VW_O \tag{2}$$

We focus on quantizing W_q, W_k although we believe our results may be generalized to include W_v and W_O.

3.2. Quantization

At a high-level, quantization works by mapping full-precision floating point values into a smaller set of low-bit numbers (e.g. 8-bit, 4-bit integers). The low-bit numbers are used during computation and then the resulting output is reconstructed by de-quantization which uses the inverse map to recover the approximate floating point value.

Uniform Quantization A common mapping used in quantization is uniform quantization. Uniform quantization splits the range R of a tensor Y uniformly onto a set of b-bit integers. The range R is defined as the difference between the minimal and maximal values of Y. This mapping is defined:

$$\text{Quant}(Y) = \text{Clamp}\left(\text{Round}\left(\frac{Y}{R}\right), -2^b, 2^b - 1\right) \tag{3}$$

Quantization error is computed between the original tensor Y and the de-quantized reconstruction $\hat{Y}$. $\hat{Y}$ is obtained by the inverse map $\text{DeQuant}() = \text{Quant}^{-1}()$. Since quantization is surjective, there can be errors in the reconstruction. We write this element-wise quantization error as

$\Delta Y = \hat{Y} - Y$. The full tensor quantization error is defined as the L2 norm of the per-element error $||\Delta Y||_2^2 = |\Delta Y|^2$. Uniform quantization depends heavily on the range R as a larger range results in a lower resolution mapping leading to higher uncertainty in the reconstruction. This means quantization error is driven primarily by the extremal values of Y, so outlier values can dramatically impact quantization. Under uniform quantization, the quantization error is approximately distributed uniformly (Marco & Neuhoff, 2005; Lin et al., 2016):

$$\Delta Y \sim \text{Uniform}\left(\frac{-R}{2^{b+1}}, \frac{R}{2^{b+1}}\right) \tag{4}$$

4. Data-Free Estimation of Quantization Noise

To improve quantization performance, we take a similar approach to Meller et. al (Meller et al., 2019) by analyzing quantization noise. We compute an analytic expression for the quantization noise, which gives a data-free objective for minimizing the error under quantization. In what follows $Y = W_q W_k^T$ which when quantized and reconstructed gives $\hat{Y} = \hat{W}_q \hat{W}_k{}^T$. Rewriting this in terms of the quantization error we get an expression for ΔY:

$$\begin{aligned} Y + \Delta Y =& (W_q + \Delta W_q)(W_k + \Delta W_k)^T \\ \Delta Y =& W_q \Delta W_k^T + \Delta W_q W_k^T + \Delta W_q \Delta W_k^T \end{aligned} \tag{5}$$

The element-wise quantization errors $\Delta W_q, \Delta W_k$ are both random tensors approximately distributed as:

$$\Delta W_q \sim \text{Uniform}\left(\frac{-R_q}{2^{b+1}}, \frac{R_q}{2^{b+1}}\right) \tag{6}$$

$$\Delta W_k \sim \text{Uniform}\left(\frac{-R_k}{2^{b+1}}, \frac{R_k}{2^{b+1}}\right) \tag{7}$$

where R_q, R_k are the ranges of W_q, W_k respectively and b is the quantization bit-width. This means ΔY is also a random tensor which depends on $\Delta W_q, \Delta W_k$. We define the quantization noise as the average element-wise variance $\text{mean}(\mathbf{E}(|\Delta Y|^2))$ which is the expected magnitude of the full tensor quantization error. Intuitively higher quantization noise corresponds to higher uncertainty in the de-quantized reconstruction $\hat{Y}$ which is driven by outliers which pose significant challenges to effective quantization. This makes minimizing quantization noise a promising data-free objective that can lead to fewer outliers and better quantization.

We now show how to compute the quantization noise, for a full proof see Appendix A. In what follows $\odot$ is element-wise multiplication. Expanding and simplifying $\mathbf{E}(|\Delta Y|^2)$ yields a sum over over 6 term matrices. Equations for each of these terms is included in Appendix 1.

$$\begin{aligned} \mathbf{E}(|\Delta Y|^2) = \mathbf{E}[& |W_q \Delta W_k^T|^2 + |\Delta W_q W_k^T|^2 \\ & + |\Delta W_q \Delta W_k^T|^2 \\ & + |W_q \Delta W_k^T| \odot |\Delta W_q W_k^T| \\ & + |W_q \Delta W_k^T| \odot |\Delta W_q \Delta W_k^T| \\ & + |\Delta W_q W_k^T| \odot |\Delta W_q \Delta W_k^T|] \end{aligned} \tag{8}$$

Since we only need the element-wise mean of this matrix expression, these terms can be further reduced giving the following proposition.

Proposition 4.1. *Let $W_q, W_k \in \mathbb{R}^{n\times m}$ with elements denoted q_{ij}, k_{ij}. Let $\Delta W_q, \Delta W_k$ be their quantization error matrices respectively. If $\Delta W_q \sim$ Uniform $(-r_q, r_q)$ and $\Delta W_k \sim$ Uniform $(-r_k, r_k)$ then the mean of the elements in the matrix expression in Equation 8 is:*

$$\begin{aligned} & \frac{r_k^2}{n}\left(\frac{\sum_{i,j} q_{ij}^2}{12} + \frac{\sum_{i,j,t} q_{ij} q_{it}}{4}\right) + \frac{r_q^2}{n}\left(\frac{\sum_{i,j} k_{ij}^2}{12} + \frac{\sum_{i,j,t} k_{ij} k_{it}}{4}\right) \\ & + \frac{m r_q^2 r_k^2}{16}\left(m + \frac{7}{9}\right) + \frac{r_q r_k}{2n^2}\left(\sum_{i,j} q_{ij}\right)\left(\sum_{i,j} k_{ij}\right) \\ & + \frac{r_q r_k^2 (3m+1)}{12n} \sum_{i,j} q_{ij} + \frac{r_q^2 r_k (3m+1)}{12n} \sum_{i,j} k_{ij} \end{aligned}$$

where the summands correspond to those in Equation 8.

5. Method

We introduce our algorithm for finding a transformation of W_q, W_k which minimizes the quantization noise without changing the layer function. From Equation 1, the attention scores are computed as $A = x W_q W_k^T x^T$ where $W_q, W_k \in \mathbb{R}^{n\times m}$. An invertible matrix $g \in \mathbb{R}^{m\times m}$ and its inverse can be inserted between W_q and W_k giving an equal attention score:

$$A = x W_q g g^{-1} W_k^T x^T \tag{9}$$

Replacing the original weights with $W'_q = W_q g$, and $W'_k = W_k (g^{-1})^T$ gives a new set of weights without changing the layer functionally.

Our goal is to find such a transformation g which minimizes the quantization noise for the new weights. Instead of searching over the group $\text{GL}(m)$, all invertible $m \times m$ matrices, we restrict g to be orthogonal. The group $O(m)$ is compact, which assures the existence of a global minimum, making the optimization problem well posed. Due to orthogonality the new weights are $W'_q = W_q g$ and $W'_k = W_k g$. Our objective more concretely is to solve the following minimization:

$$g = \text{argmin}_{g \in O(m)} \text{mean}(\mathbf{E}(\Delta Y'^2)) \tag{10}$$

Model	SST-2 - Acc.	MNLI - Acc.
Full-Prec.	92.2%	84.1%
Stand. 8-bit	91.9%	84.1%
Mod. 8-bit	91.9%	84.1%
Stand. 4-bit	91.7%	84.0%
Mod. 4-bit	91.9%	84.1%

Table 1. MNLI and SST-2 quantization performance results. Stand. 8-bit and Stand. 4-bit were quantized without weight modification. Mod. 8-bit and Mod. 4-bit had weights modified before quantization.

This is solvable by gradient descent using the expression from Proposition 4.1 as a loss function. We parameterize g by instantiating a square random matrix M and setting g as the orthogonal component of the QR decomposition QR(M). Since the QR decomposition is differentiable, this makes for a suitable parameterization. We perform this procedure for each layer of the transformer model and for each head in multi-headed attention layers which can be batched to improve efficiency.

6. Experimental Evaluation

As a proof of concept, we tested our approach by validating the performance impact of quantization with and without our transformation. We used Bert$_{\text{base}}$ (Devlin et al., 2018) fine-tuned for two benchmark GLUE tasks, SST-2 and MNLI. The model weights were quantized to 8-bit and 4-bit integers without any activation quantization. The transformation optimization was run for 5,000 iterations for both tasks. The results are summarized in Table 6. 8-bit and 4-bit weight quantization did not degrade performance nearly at all for either task and so our weight modifications had only a marginal impact on quantization. We believe further testing with activation quantization may be necessary to sufficiently test the impact of our approach.

7. Discussion

In this paper we explored using parameter symmetries to improve quantization. We derived an estimate for quantization noise in query and key attention. Our approach for minimizing quantization noise is a highly efficient pre-processing step which is compatible with other downstream quantization approaches and may be a promising technique for outlier mitigation. In the future we plan to evaluate the impacts of our approach on activation quantization and on generative language tasks which have been shown to be more sensitive to quantization. We also plan to generalize our noise estimate to per-group and per-channel quantization which may provide a more fine-grained estimate.

References

Badrinarayanan, V., Mishra, B., and Cipolla, R. Symmetry-invariant optimization in deep networks. *arXiv preprint arXiv:1511.01754*, 2015.

Bondarenko, Y., Nagel, M., and Blankevoort, T. Understanding and overcoming the challenges of efficient transformer quantization, 2021. URL https://arxiv.org/abs/2109.12948.

Chen, M., Shao, W., Xu, P., Wang, J., Gao, P., Zhang, K., and Luo, P. Efficientqat: Efficient quantization-aware training for large language models, 2024. URL https://arxiv.org/abs/2407.11062.

Dettmers, T., Lewis, M., Belkada, Y., and Zettlemoyer, L. Llm.int8(): 8-bit matrix multiplication for transformers at scale. *Advances in neural information processing systems*, 35:30318–30332, 2022.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Esser, S. K., McKinstry, J. L., Bablani, D., Appuswamy, R., and Modha, D. S. Learned step size quantization. *arXiv preprint arXiv:1902.08153*, 2019.

Frantar, E., Ashkboos, S., Hoefler, T., and Alistarh, D. Gptq: Accurate post-training quantization for generative pre-trained transformers. *arXiv preprint arXiv:2210.17323*, 2022.

Hecht-Nielsen, R. On the algebraic structure of feedforward network weight spaces. In *Advanced Neural Computers*, pp. 129–135. Elsevier, 1990.

Hubara, I., Nahshan, Y., Hanani, Y., Banner, R., and Soudry, D. Accurate post training quantization with small calibration sets. In *International Conference on Machine Learning*, pp. 4466–4475. PMLR, 2021.

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H., and Kalenichenko, D. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2704–2713, 2018.

Kim, N. J., Lee, J., and Kim, H. Hyq: Hardware-friendly post-training quantization for cnn-transformer hybrid networks. In *IJCAI*, 2024.

Li, Y., Gong, R., Tan, X., Yang, Y., Hu, P., Zhang, Q., Yu, F., Wang, W., and Gu, S. Brecq: Pushing the limit of post-training quantization by block reconstruction. *arXiv preprint arXiv:2102.05426*, 2021.

Lin, D. D., Talathi, S. S., and Annapureddy, V. S. Fixed point quantization of deep convolutional networks, 2016. URL https://arxiv.org/abs/1511.06393.

Ma, S., Wang, H., Ma, L., Wang, L., Wang, W., Huang, S., Dong, L., Wang, R., Xue, J., and Wei, F. The era of 1-bit llms: All large language models are in 1.58 bits, 2024. URL https://arxiv.org/abs/2402.17764.

Marco, D. and Neuhoff, D. The validity of the additive noise model for uniform scalar quantizers. *IEEE Transactions on Information Theory*, 51(5):1739–1755, 2005. doi: 10.1109/TIT.2005.846397.

Meller, E., Finkelstein, A., Almog, U., and Grobman, M. Same, same but different: Recovering neural network quantization error through weight factorization. In *International Conference on Machine Learning*, pp. 4486–4495. PMLR, 2019.

Nagel, M., Baalen, M. v., Blankevoort, T., and Welling, M. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1325–1334, 2019.

Nagel, M., Amjad, R. A., Van Baalen, M., Louizos, C., and Blankevoort, T. Up or down? adaptive rounding for post-training quantization. In *International conference on machine learning*, pp. 7197–7206. PMLR, 2020.

Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., Van Baalen, M., and Blankevoort, T. A white paper on neural network quantization. *arXiv preprint arXiv:2106.08295*, 2021.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

Wu, Z., Dong, J., Aloui, A., and Tarokh, V. Teleportation with null space gradient projection for optimization acceleration. *arXiv preprint arXiv:2502.11362*, 2025.

Xiao, G., Lin, J., Seznec, M., Wu, H., Demouth, J., and Han, S. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International Conference on Machine Learning*, pp. 38087–38099. PMLR, 2023.

Yao, Z., Yazdani Aminabadi, R., Zhang, M., Wu, X., Li, C., and He, Y. Zeroquant: Efficient and affordable post-training quantization for large-scale transformers. *Advances in Neural Information Processing Systems*, 35: 27168–27183, 2022.

Zhang, B., Zheng, Z., Chen, Z., and Li, J. Beyond the permutation symmetry of transformers: The role of rotation for model fusion. *arXiv preprint arXiv:2502.00264*, 2025.

Zhang, S., Roller, S., Goyal, N., Artetxe, M., Chen, M., Chen, S., Dewan, C., Diab, M., Li, X., Lin, X. V., Mihaylov, T., Ott, M., Shleifer, S., Shuster, K., Simig, D., Koura, P. S., Sridhar, A., Wang, T., and Zettlemoyer, L. Opt: Open pre-trained transformer language models, 2022. URL https://arxiv.org/abs/2205.01068.

Zhu, X., Li, J., Liu, Y., Ma, C., and Wang, W. A survey on model compression for large language models. *Transactions of the Association for Computational Linguistics*, 12:1556–1577, 2024.

A. Quantization Noise Estimation Proof

In this section we provide a proof of the equations found in proposition 4.1. In the following proofs, Q^i, K^i denote the m-dimensional i-th row vectors of W_q, W_k and $\delta Q^i, \delta K^i$ are the rows of $\Delta W_q, \Delta W_k$.

1st Term: $|W_q \Delta W_k^T|^2$ We begin by first considering the matrix $\mathbf{E}[|W_q \Delta W_k^T|^2]$. The value $\mathbf{E}[|W_q \Delta W_k^T|_{ij}^2]$ at index i, j is computed as follows:

$$\begin{aligned}\mathbf{E}[|W_q \Delta W_k^T|_{ij}^2] &= \mathbf{E}[|Q^i (\delta K^j)^T|^2] \\ &= \mathbf{E}[|\sum_u Q_u^i \delta K_u^j| \cdot |\sum_v Q_v^i \delta K_v^j|]\end{aligned}$$

Expanding this product, the expectation can be distributed through the sum. In this expanded product there are 2 cases, when $u = v$ and $u \neq v$.

When $u = v$, this gives $\mathbf{E}[|Q_u^i \delta K_u^j|^2] = (Q_u^i)^2 \frac{r_k^2}{3}$ since $|\delta K_u^j| \sim \text{Uniform}(0, r_k)$. Since u, v go from 1 to m, this will give us $\frac{r_k^2}{3} \sum_{u=1}^{m} (Q_u^i)^2$ in the sum.

When $u \neq v$, the value is $\mathbf{E}[|Q_u^i \delta K_u^j|]\mathbf{E}[|Q_v^i \delta K_v^j|]$ since δK_u^j and δK_v^j are independent random values so their expectations are multiplied. This gives $\frac{r_k^2}{4} \sum_{u \neq v} Q_u^i Q_v^i$.

Putting both cases together we get the final value for index i, j of

$$\begin{aligned}\mathbf{E}[|W_q \Delta W_k^T|_{ij}^2] &= \frac{r_k^2}{3} \sum_{u=1}^{n} (Q_u^i)^2 + \frac{r_k^2}{4} \sum_{u \neq v} Q_u^i Q_v^i \\ &= \frac{r_k^2}{3} \sum_{u=1}^{n} (Q_u^i)^2 + \frac{r_k^2}{4} \sum_{u,v} Q_u^i Q_v^i - \frac{r_k^2}{4} \sum_{u=1}^{n} (Q_u^i)^2 \\ &= \frac{r_k^2}{12} \sum_{u=1}^{n} (Q_u^i)^2 + r_k^2 \sum_u Q_u^i \sum_v Q_v^i\end{aligned}$$

Note that this final value does not depend on j meaning all of the values in row i will have this value giving us a total of n copies.

We now take the average over the n^2 values in $\mathbf{E}[|W_q \Delta W_k^T|_{ij}^2]$ which gives us the desired form:

$$\text{mean}(\mathbf{E}[|W_q \Delta W_k^T|^2]) = \frac{r_k^2}{n} \left(\frac{\sum_{i,j} q_{ij}^2}{12} + \frac{\sum_i (\sum_{j,t} q_{ij} q_{it})}{4} \right)$$

2nd Term: $|\Delta W_q W_k^T|^2$ Following the same reasoning as the previous term, the value $\mathbf{E}[(\Delta W_q W_k^T)_{ij}^2]$ is:

$$\begin{aligned}\mathbf{E}[|\Delta W_q W_k^T|_{ij}^2] &= \mathbf{E}[|\delta Q^i (K^j)^T|^2] \\ &= \mathbf{E}[|\sum_u \delta Q_u^i \delta K_u^j| \cdot |\sum_v \delta Q_v^i K_v^j|]\end{aligned}$$

The exact same simplifications as before occur but since δQ^i is the random vector, we instead will get a formula which does not depend on i:

$$\mathbf{E}[|W_q \Delta W_k^T|_{ij}^2] = \frac{r_q^2}{12} \sum_{u=1}^{n} (K_u^j)^2 + \frac{r_q^2}{4} \sum_u K_u^j \sum_v K_v^j$$

Taking the average over the n^2 values gives the final form:

$$\text{mean}(\mathbf{E}[|\Delta W_q W_k^T|^2]) = \frac{r_q^2}{n} \left(\frac{\sum_{i,j} k_{ij}^2}{12} + \frac{\sum_i (\sum_{j,t} k_{ij} q_{it})}{4} \right)$$

3rd Term: $|\Delta W_q \Delta W_k^T|^2$ This case is much easier since the values of $\Delta W_q, \Delta W_k^T$ are i.i.d. and so every value of the matrix $\mathbf{E}[|\Delta W_q \Delta W_k^T|^2]$ are equal. A single value of this matrix is computed:

$$\mathbf{E}[|\Delta W_q \Delta W_k^T|^2_{ij}] = |\sum_u \delta Q_u^i \delta K_u^j| \cdot |\sum_v \delta Q_v^i \delta K_v^j|$$

In the first case $u = v$, the result is $\mathbf{E}[|\delta Q_u^i \delta K_u^j|^2] = \frac{r_q^2 r_k^2}{9}$. This will happen m times since u, v go from 1 to m.

The second case $u \neq v$ gives $\mathbf{E}[|\delta Q_u^i \delta K_u^j| \cdot |\delta Q_v^i \delta K_v^j|] = \frac{r_q^2 r_k^2}{16}$. This happens for when $u \neq v$ so we will have this $m(m-1)$ times in the sum.

Putting these two together we get a simplified per element value of:

$$\begin{aligned}\mathbf{E}[|\Delta W_q \Delta W_k^T|^2_{ij}] &= m\frac{r_q^2 r_k^2}{9} + (m^2 - m)\frac{r_q^2 r_k^2}{16} \\ &= \frac{m r_q^2 r_k^2}{16}(m + \frac{7}{9})\end{aligned}$$

The average value is exactly equal to the per element value since every element is equivalent under expectation.

4th Term: $|W_q \Delta W_k^T| \odot |\Delta W_q W_k^T|$ Once again begin with the i, j entry of the matrix:

$$\begin{aligned}\mathbf{E}[|W_q \Delta W_k^T| \odot |\Delta W_q W_k^T|_{ij}] &= \mathbf{E}[|Q^i(\delta K^j)^T| \cdot |\delta Q^i (K^j)^T|] \\ &= \mathbf{E}[|\sum_u Q_u^i \delta K_u^j| \cdot |\sum_v \delta Q_v^i K_v^j|] \\ &= (m\frac{r_k}{2}\sum_u Q_u^i)(m\frac{r_q}{2}\sum_v K_v^j)\end{aligned}$$

Averaging over all i, j elements gives the final form:

$$\text{mean}(\mathbf{E}[|W_q \Delta W_k^T| \odot |\Delta W_q W_k^T|]) = \frac{r_q r_k}{2n^2}(\sum_{i,j} q_{ij})(\sum_{i,j} k_{ij})$$

5th Term: $|W_q \Delta W_k \odot \Delta W_q \Delta W_k|$

$$\begin{aligned}\mathbf{E}[|W_q \Delta W_k^T| \odot |\Delta W_q \Delta W_k^T|_{ij}] &= \mathbf{E}[|Q^i(\delta K^j)^T| \cdot |\delta Q^i (\delta K^j)^T|] \\ &= \mathbf{E}[|\sum_u Q_u^i \delta K_u^j| \cdot |\sum_v \delta Q_v^i \delta K_v^j|]\end{aligned}$$

Once again there are 2 cases when $u = v$ and when $u \neq v$. In the first case $\mathbf{E}[(Q_u^i \delta K_u^j)(\delta Q_u^i \delta K_u^j)] = \frac{r_q}{2}\frac{r_k^2}{3}Q_u^i$. In the second case the random values are all independent so the result is: $\mathbf{E}[(Q_u^i \delta K_u^j)(\delta Q_u^i \delta K_v^j)] = \frac{r_q}{2}\frac{r_k^2}{4}Q_u^i$.

Adding this up and simplifying gives the value for element i, j:

$$\mathbf{E}[|W_q \Delta W_k^T| \odot |\Delta W_q \Delta W_k^T|_{ij}] = m\frac{r_q r_k^2}{6}\sum_u Q_u^i + (m^2 - m)\frac{r_q r_k^2}{8}\sum_u Q_u^i$$

Averaging over all i, j elements gives:

$$\text{mean}(\mathbf{E}[|W_q \Delta W_k^T| \odot |\Delta W_q \Delta W_k^T|] = \frac{r_q r_k^2 (3m+1)}{12n}\sum_{i,j} q_{ij}$$

6th Term: $|\Delta W_q W_k^T| \odot |\Delta W_q \Delta W_k^T|$ This term follows the same reasoning as above. Starting with entry i, j:

$$\begin{aligned}\mathbf{E}[|\Delta W_q W_k^T| \odot |\Delta W_q \Delta W_k^T|_{ij}] &= \mathbf{E}[|\delta Q^i (K^j)^T| \cdot |\delta Q^i (\delta K^j)^T|] \\ &= \mathbf{E}[|\sum_u \delta Q_u^i K_u^j| \cdot |\sum_v \delta Q_v^i \delta K_v^j|]\end{aligned}$$

In the case where $u = v$ we get $\mathbf{E}[(\delta Q_u^i K_u^j)(\delta Q_u^i \delta K_u^j)] = \frac{r_q^2}{3} \frac{r_k}{2} K_u^j$. Similarly for $u \neq v$ gives $\mathbf{E}[(\delta Q_u^i K_u^j)(\delta Q_u^i \delta K_v^j)] = \frac{r_q^2}{4} \frac{r_k}{2} K_u^j$.

Adding both cases up and simplifying gives:

$$\mathbf{E}[|\Delta W_q W_k^T| \odot |\Delta W_q \Delta W_k^T|_{ij}] = m \frac{r_q^2 r_k}{6} \sum_u K_u^j + (m^2 - m) \frac{r_q^2 r_k}{8} \sum_u K_u^j$$

Averaging over all i, j elements gives the final equation:

$$\text{mean}(\mathbf{E}[|\Delta W_q W_k^T| \odot |\Delta W_q \Delta W_k^T|]) = \frac{r_q^2 r_k (3m + 1)}{12n} \sum_{i,j} k_{ij}$$