NSNQuant: A Double Normalization Approach for Calibration-Free Low-Bit Vector Quantization of KV Cache

Donghyun Son, Euntae Choi, Sungjoo Yoo

CMALab, Seoul National Univerity

Preliminary

Preliminary: KV cache

- In a transformer decoder architecture, key and value (**KV**) are **cached** to avoid redundant computations
- The size of KV cache **increases linearly** with the batch size and the sequence length.
- This leads to huge memory overhead and becomes the main bottleneck of LLM inference

Preliminary: Vector Quantization

- A group of values (vector) is quantized by mapping it into an integer index in a codebook
- example) quantize an 8-dimensional vector using a codebook of size 256.
  - log(256) = 8 bits are needed to compress a 8-dimensional vector.
  - average bit width = 8 / 8 = **1**

$$\mathrm{VQ}(v) = \underset{i}{\operatorname{argmin}}\, \mathrm{D}(v, \mathbb{C}[i])$$

Preliminary: Hadamard transform

Hadamard matrix: recursively defined orthogonal matrix

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad \text{and} \quad \mathbf{H}_{2^n} = \mathbf{H}_2 \otimes \mathbf{H}_{2^{n-1}} .$$

Hadamard transform: Compute multiplication of the Hadamard matrix in O(n log n) complexity.

Known properties:

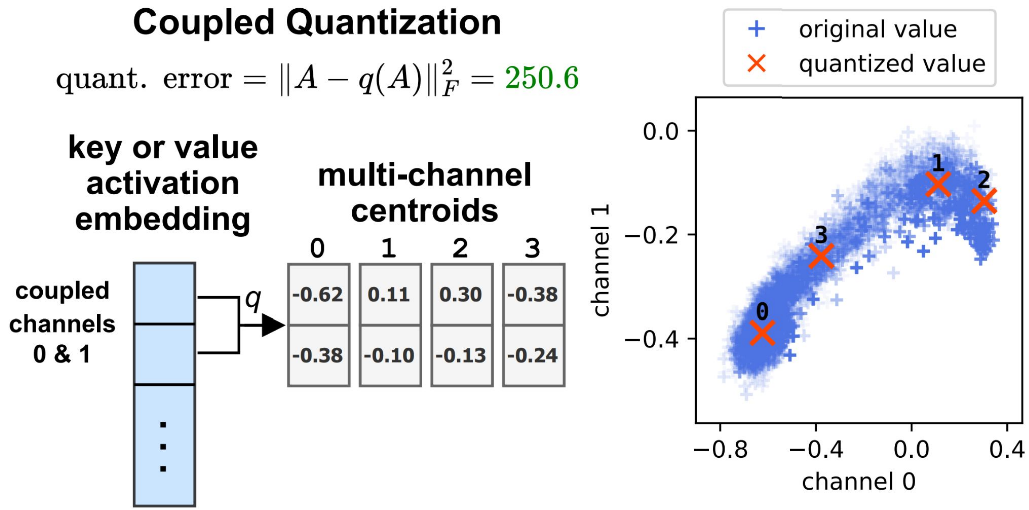
- suppress outliers
- yield gaussian-like outputs

Motivation

Motivation 1: success of VQ in LLM quantization

- State-of-the-art methods in weight-only quantization: QuIP#, AQLM, VPTQ
- For KV cache: CQ (Coupled Quantization) applies VQ to KV Cache

CQ (Coupled Quantization)

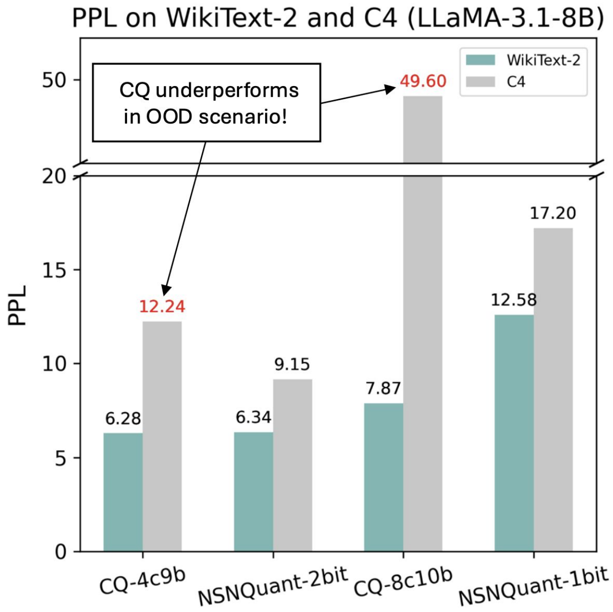


Idea: Generate a codebook to compress “coupled” channels.

- use a small set of data, called **calibration dataset** to obtain KV distribution.

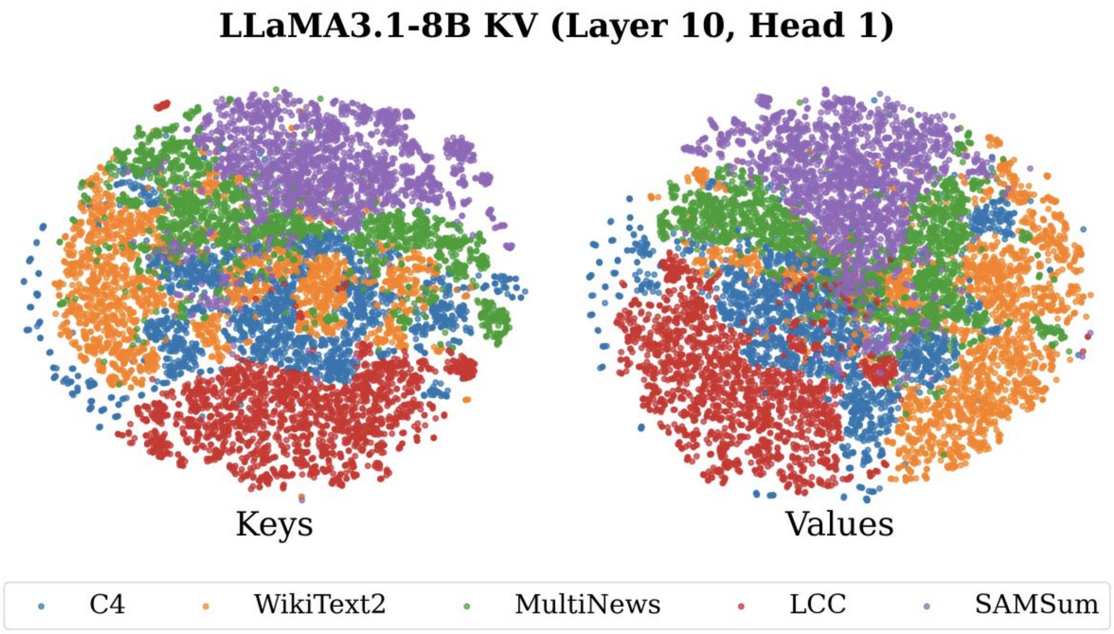
Motivation 2: failure of CQ in OOD (out-of-distribution) scenario

- CQ builds codebooks using the calibration dataset (16 samples, WikiText-2)
- Although it works well in an ID (in-distribution) scenario, it performs worse in OOD scenarios.



Why CQ is susceptible to distribution shift

- We find that the KV distribution relies heavily on the input distribution



Our approach: remove a calibration process from quantization

- CQ: build a codebook that fits well to KV distribution using a small set of data (calibration dataset)

- Ours (NSNQuant): align the KV distribution with a **well-known prior** (standard normal distribution) through the normalization
  - does not rely on any external data

Method

How to align KV distribution to a well-known prior?

- first idea: use **Hadamard transform**, which yields **normal-like** (gaussian-like) outputs
  - limitation: although the shape of the distribution is normal-like, its **mean** and **variance** are not controlled

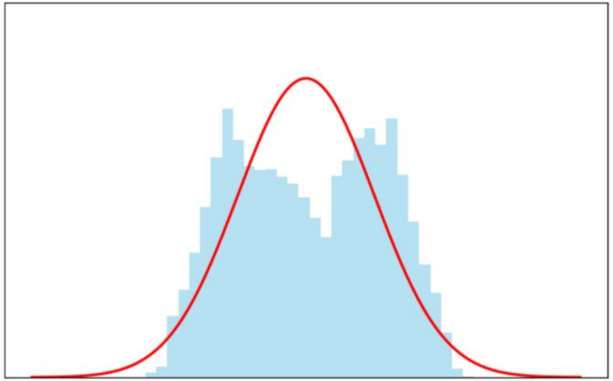
Our approach: 3-step normalization (NSN)

- Hadamard transform: yields **normal-like** (gaussian-like) outputs

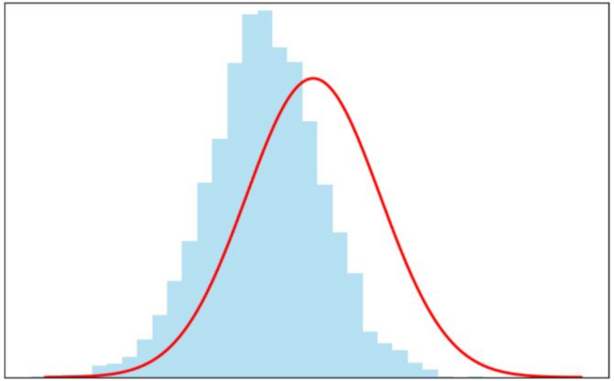
- NSN (**N**ormalize-**S**hift-**N**ormalize): **standardize** the outputs, when used with Hadamard transform.
  1. (Normalize) **token-wise normalization**: scale each key/value of a token to have a scale of sqrt(d)
  2. (Shift) **channel-wise centering**: set mean of each channel to zero
  3. (Normalize) **token-wise normalization**: scale each key/value of a token to have a scale of sqrt(d)

Put together: output distribution is aligned with the standard normal distribution!

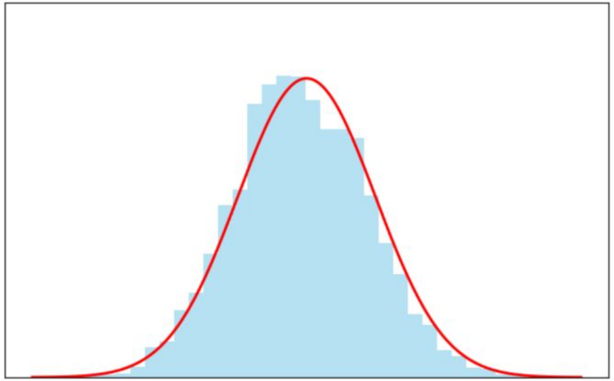
Visual illustration of effects of NSN



Original distribution
unpredictable



Hadamard transform
Gaussian-shaped, but not standardized



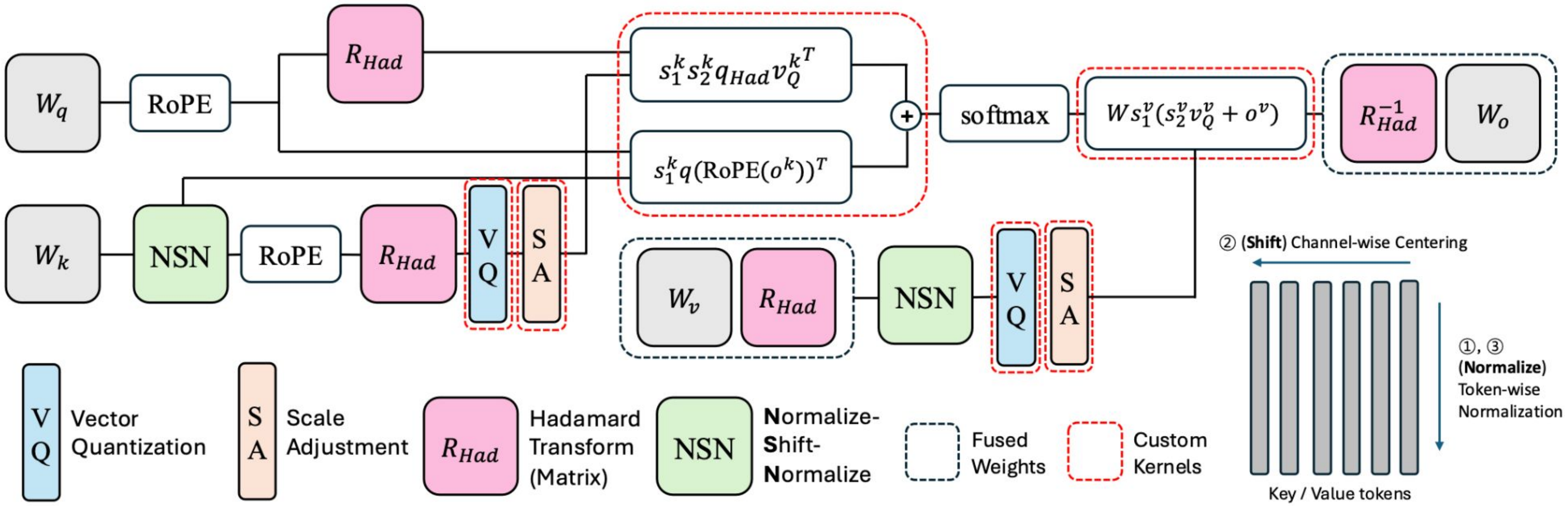
NSN + Hadamard transform
aligned with standard normal distribution!

Codebook construction

- KV distribution is aligned with the standard normal distribution
  - we can build a **single reusable codebook** tailored for the standard normal data, using the **synthetic data**

- Build a codebook using **K-Means algorithm** with **further fine-tuning**

NSNQuant: overall structure



A detailed explanation of attention computation and other components is provided in the paper

Experimental Results

PPL evaluation

Table 2: Perplexity on WikiText-2 and C4 with a context length of 4096. The results of CQ reported in the original paper are marked with †.

Method	Avg. bit width	Dataset	LLaMA2-7B	LLaMA2-13B	LLaMA3-8B	LLaMA3.1-8B	Mistral-7B-v0.3
FP16	16	C4	6.63	6.04	8.32	8.43	7.48
		WikiText-2	5.12	4.57	5.75	5.84	4.95
KIVI-2	2.38	C4	8.00	7.03	16.43	15.80	8.83
		WikiText-2	6.14	5.30	10.93	10.55	6.03
KIVI-2 + Had	2.38	C4	7.57	6.68	12.67	12.54	8.43
		WikiText-2	5.79	5.05	8.69	8.86	5.65
KVQuant-2b + 1%	2.32	C4	7.09	6.37	9.75	9.60	7.93
		WikiText-2	5.52	4.88	6.74	6.71	5.32
CQ-4c9b	2.26	C4	7.12 (7.02†)	6.45 (6.36†)	13.97	12.24	7.86
		WikiText-2	5.36 (5.32†)	4.76 (4.74†)	**6.16**	**6.28**	5.16
NSNQuant-2b	2.23	C4	**6.86**	**6.21**	**9.08**	**9.15**	**7.69**
		WikiText-2	**5.29**	**4.71**	6.23	6.34	**5.12**
KVQuant-1b + 1%	1.32	C4	30.79	14.27	33.17	37.37	12.45
		WikiText-2	13.5	9.91	27.57	33.96	9.06
CQ-8c10b	1.27	C4	9.25 (9.12†)	8.17 (8.01†)	43.78	49.60	**9.60**
		WikiText-2	6.33 (**6.25†**)	5.53 (**5.47†**)	**7.69**	**7.87**	**6.01**
NSNQuant-1b	1.23	C4	**8.70**	**7.55**	**16.69**	**17.20**	9.67
		WikiText-2	6.69	5.70	11.70	12.58	6.66

LongBench evaluation

Table 3: Evaluation results on LongBench. The task subset is selected following KIVI [30]. More results with different models can be found in Table 16.

Model	Method	Bits	Qasper	QMSum	MultiNews	TREC	TriviaQA	SAMSum	LCC	RepoBench-P	Avg.
LLaMA3.1-8B-Instruct	FP16	16	13.11	23.53	26.74	72.50	91.65	43.78	63.04	56.17	48.82
	KIVI-2	2.38	12.04	24.96	26.70	72.00	91.97	43.43	60.85	53.39	48.17
	KIVI-2 + Had	2.38	11.57	24.28	26.51	72.50	92.09	43.21	62.90	55.20	<u>48.53</u>
	KVQuant-2b + 1%	2.32	13.15	23.45	26.24	72.00	91.63	41.39	60.80	54.41	47.88
	CQ-4c9b	2.26	12.25	23.80	25.74	71.50	91.53	41.96	61.18	54.46	47.80
	NSNQuant-2b	2.23	12.44	23.74	26.95	72.50	91.73	44.01	62.05	55.09	**48.56**
	KVQuant-1b + 1%	1.32	9.91	22.19	22.27	47.50	88.92	35.76	50.27	43.79	40.08
	CQ-8cb10	1.27	8.84	21.18	22.40	47.50	87.94	38.86	53.81	45.73	<u>40.78</u>
	NSNQuant-1b	1.23	11.54	24.69	27.16	71.50	92.04	42.36	60.08	49.70	**47.38**
Mistral-7B-Instruct-v0.3	FP16	16	41.13	25.75	27.78	76.00	88.59	47.47	59.52	60.64	53.36
	KIVI-2	2.38	37.86	24.62	26.85	76.00	88.51	45.93	58.72	57.87	52.05
	KIVI-2 + Had	2.38	39.99	25.42	27.50	76.00	88.42	46.52	59.54	60.13	**52.94**
	KVQuant-2b + 1%	2.32	38.98	25.10	27.22	76.00	89.02	45.27	58.57	61.59	52.72
	CQ-4c9b	2.26	39.85	24.50	27.19	76.00	88.86	45.56	58.36	60.26	52.57
	NSNQuant-2b	2.23	39.96	24.91	27.54	76.00	88.96	46.47	58.70	59.45	<u>52.75</u>
	KVQuant-1b + 1%	1.32	28.58	21.89	22.76	50.50	87.75	39.62	54.73	54.46	45.04
	CQ-8c10b	1.27	31.21	22.56	23.12	64.50	88.09	41.71	53.82	52.31	<u>47.16</u>
	NSNQuant-1b	1.23	37.94	25.03	26.81	76.00	89.39	46.37	56.75	55.57	**51.73**

LM-eval results

Table 4: Evaluation results on GSM8K, HumanEval, CoQA, and MMLU. Accuracy is reported for all tasks. Results with LLaMA2-13B-Chat and LLaMA3-8B-Instruct can be found in Table 18

Model	Method	Bits	GSM8K (8-shot, CoT)	HumanEval	CoQA	MMLU (4-shot, CoT)			
						Humanities	STEM	Social	Other
LLaMA3.1-8B-Instruct	FP16	16	76.65	57.93	63.78	71.47	57.96	74.16	72.52
	KIVI-2	2.38	64.59	48.17	63.60	64.44	50.09	66.84	66.13
	KIVI-2 + Had	2.38	65.73	50.61	**63.88**	67.73	53.03	68.50	68.14
	KVQuant-2b + 1%	2.32	70.05	<u>53.05</u>	62.37	<u>68.18</u>	<u>54.78</u>	<u>71.31</u>	<u>69.61</u>
	CQ-4c9b	2.26	<u>72.93</u>	48.78	62.93	67.07	52.66	70.22	69.33
	NSNQuant-2b	2.23	**75.89**	**56.10**	<u>63.83</u>	**71.04**	**55.64**	**73.42**	**70.74**
	KVQuant-1b + 1%	1.32	21.53	23.17	53.55	23.04	11.23	37.59	33.02
	CQ-8c10b	1.27	<u>44.88</u>	<u>25.61</u>	<u>56.58</u>	<u>28.21</u>	<u>21.34</u>	<u>31.97</u>	<u>41.10</u>
	NSNQuant-1b	1.23	**53.45**	**44.51**	**62.70**	**59.82**	**45.83**	**65.34**	**63.77**
Mistral-7B-Instruct-v0.3	FP16	16	53.15	31.10	65.58	65.98	50.46	71.06	68.26
	KIVI-2	2.38	43.75	28.66	64.45	60.96	39.93	63.52	59.05
	KIVI-2 + Had	2.38	46.10	28.05	<u>65.48</u>	<u>63.28</u>	45.11	66.99	63.07
	KVQuant-2b + 1%	2.32	46.63	27.44	64.28	63.00	<u>45.47</u>	67.39	<u>66.27</u>
	CQ-4c9b	2.26	<u>47.84</u>	**31.10**	64.80	62.48	42.48	<u>68.73</u>	63.98
	NSNQuant-2b	2.23	**51.02**	**31.10**	**65.62**	**64.92**	**47.65**	**69.11**	**67.56**
	KVQuant-1b + 1%	1.32	16.30	19.51	55.95	16.48	9.88	17.18	14.21
	CQ-8c10b	1.27	<u>25.93</u>	<u>21.95</u>	<u>59.07</u>	<u>23.77</u>	<u>17.62</u>	<u>27.09</u>	<u>19.78</u>
	NSNQuant-1b	1.23	**38.89**	**27.44**	**63.60**	**58.52**	**40.34**	**62.34**	**58.43**

Efficiency analysis

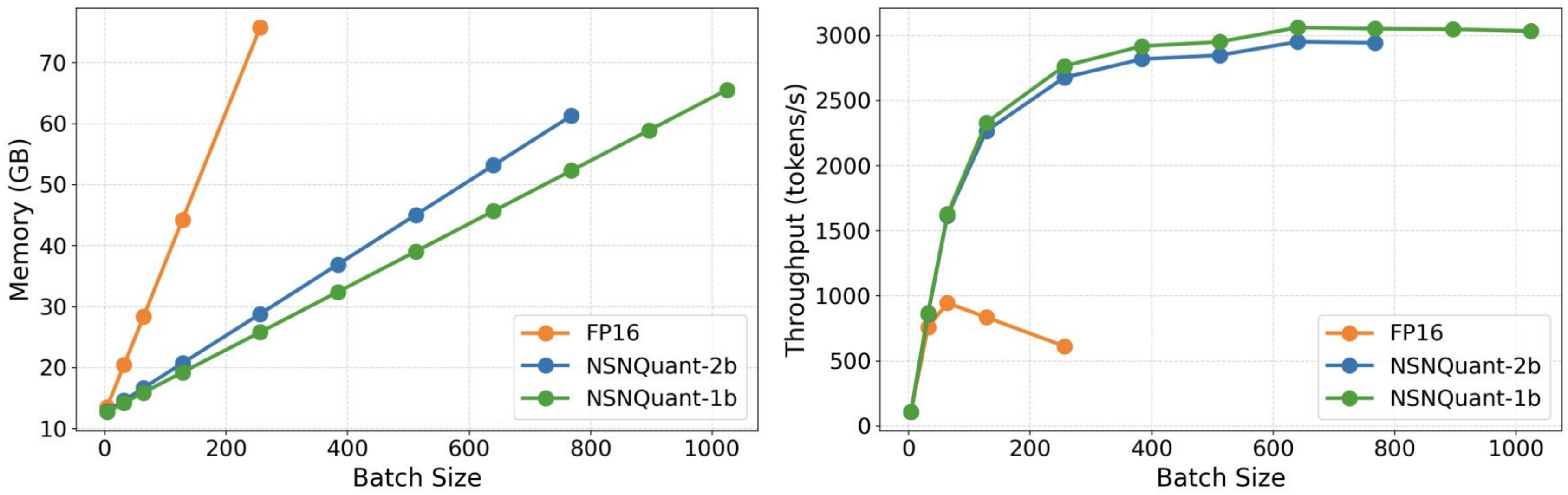


Figure 4: Peak memory usage (left) and throughput (right) measured with varying batch sizes. The residual size is set to 64. Results with varying residual sizes are available in Figure 8.

Thank you!

Donghyun Son, Euntae Choi, Sungjoo Yoo