

Pretraining LLMs at Scale: Tuning Strategies and Performance Portability

Adrián Pérez Diéguez

Staff Engineer, Qualcomm AI Research

SC PMBS Workshop

November 17th, 2025, St. Louis, Missouri, USA



Pretraining LLMs at Scale

Goal: Optimize LLM training at scale

Contribution: Provide a tuning methodology that accelerates multi-node training across different platforms

More details in paper:

Pretraining LLMs at Scale: Tuning Strategies and Performance Portability.

Adrián Pérez Diéguez, Àlex Batlle Casellas, Aleix Torres-Camps, Harris Teague,
Jordi Ros-Giralt
Qualcomm AI Research*

ABSTRACT

Training large language models (LLMs) at scale presents challenges that demand careful co-design across software, hardware, and parallelization strategies. In this work, we introduce a communication-aware tuning methodology for optimizing LLM pretraining, and adapt the performance portability metric to evaluate LLM-training efficiency across our systems. Our methodology, validated through LLM pre-training workloads at a leading global technology enterprise, delivered up to 1.6x speedup over default configurations. We further provide six key insights that challenge prevailing assumptions in LLM training performance, including the trade-offs between ZeRO stages, the default DeepSpeed communication collectives, and the critical role of batch size choices. Our findings highlight the need for platform-specific tuning and advocate for a shift toward end-to-end co-design to unlock performance efficiency in LLM training.

Converged Ethernet) instead of InfiniBand, raising questions about its efficiency. The second major bottleneck is *GPU memory bandwidth and capacity*, as GPUs must handle activations, optimizer states, and gradients, creating memory pressure, which also prevents theoretical peak GPU FLOPs utilization. Beyond hardware, software environments (e.g., NCCL/RCCL/MPI libraries, driver versions, Python packages) can cause large performance variations. Therefore, training configurations (e.g., selection of batch size, gradient accumulation, optimizer, etc.) play a central role in shaping overall performance, directly interacting with the exposed bottlenecks in LLM training and requiring targeted tuning.

Due to the high cost of performance tuning, practitioners often rely on simulators [38, 48] or ML-based performance models [29, 41] to tune trainings. Despite these approaches, performance tuning remains highly challenging due to the vast search space and training costs. Additionally, quanti-

We will challenge some common assumptions made by practitioners, highlighted in frames and referred as Takeaways, during the presentation.

Outline

- Motivation
- LLM Training
- Used Platforms
- Performance Analysis (Model 1)
- Tuning Methodology
- Methodology Evaluation: Experimental (Model 2)
- Conclusions

Motivation

Finding the optimal training configuration is challenging and expensive

What factors influence performance?

- **Internode communication**
 - Partition of data, models, gradients and optimizer states across nodes.
 - Costly collectives (all-reduce, all-gather, ...): communication and synchronization.
 - Network stack and topology
- **GPU memory bandwidth and capacity**
 - Memory footprint: $12 \times \# \text{parameters} + \text{Data} + \text{Activations} + \text{Buffers}$
- **Software environment and framework setup**
 - Framework and package versions: bugs / slowdowns
 - Containers, libraries, drivers.

What/How to tune?

- Complex AI stack + interdependencies
- Simulators / performance models: expensive, vast search space.

Motivation

What factors influence performance?

- **Internode communication**
 - Partition of data, models, gradients and optimizer states across nodes.
 - Costly collectives (all-reduce, all-gather, ...): communication and synchronization.
 - Network stack and topology
- **GPU memory bandwidth and capacity**
 - Memory footprint: $12 \times \# \text{parameters} + \text{Data} + \text{Activations} + \text{Buffers}$
- **Software environment and framework setup**
 - Framework and package versions: bugs / slowdowns
 - Containers, libraries, drivers.

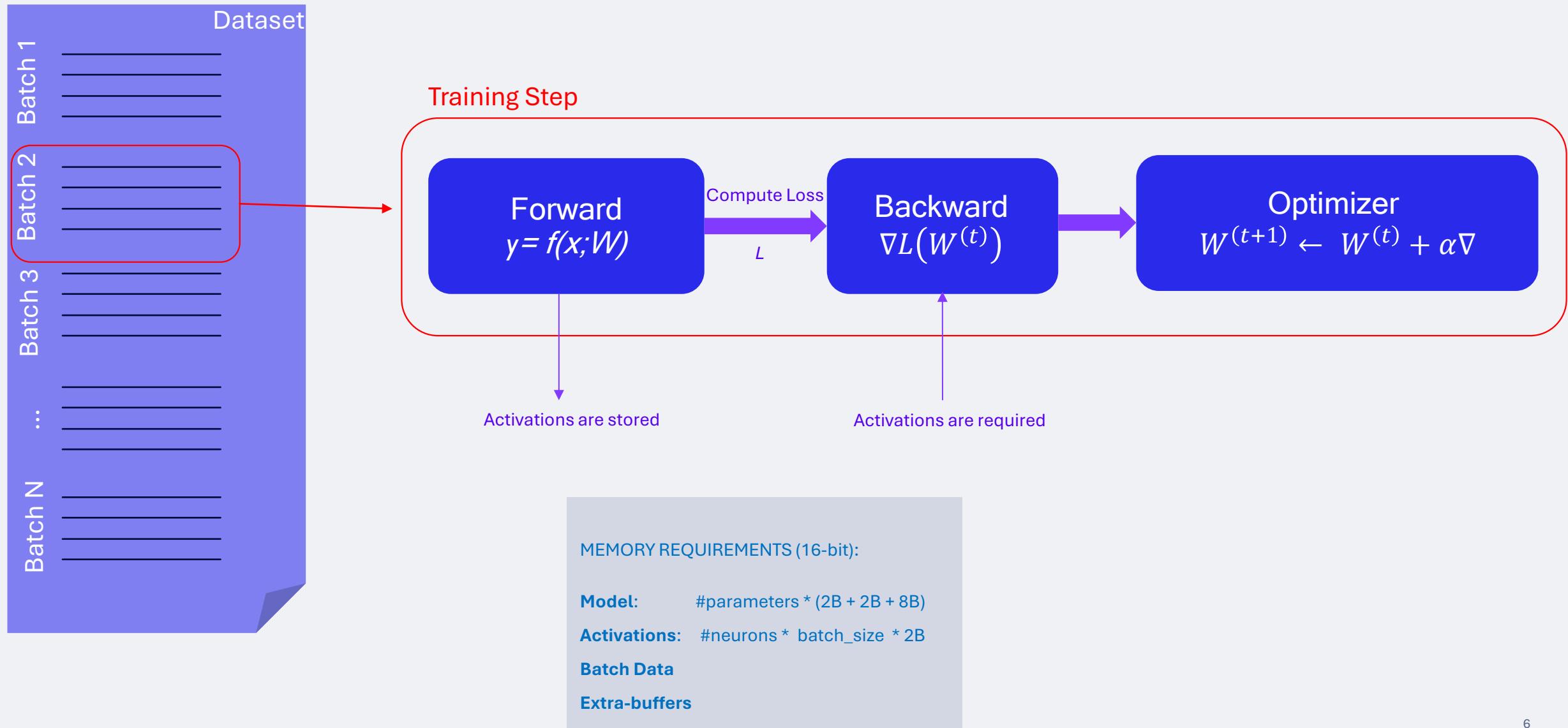
Goal: Optimize LLM training at scale

Contribution: Provide a tuning methodology that accelerates multi-node training across different platforms

What/How to tune?

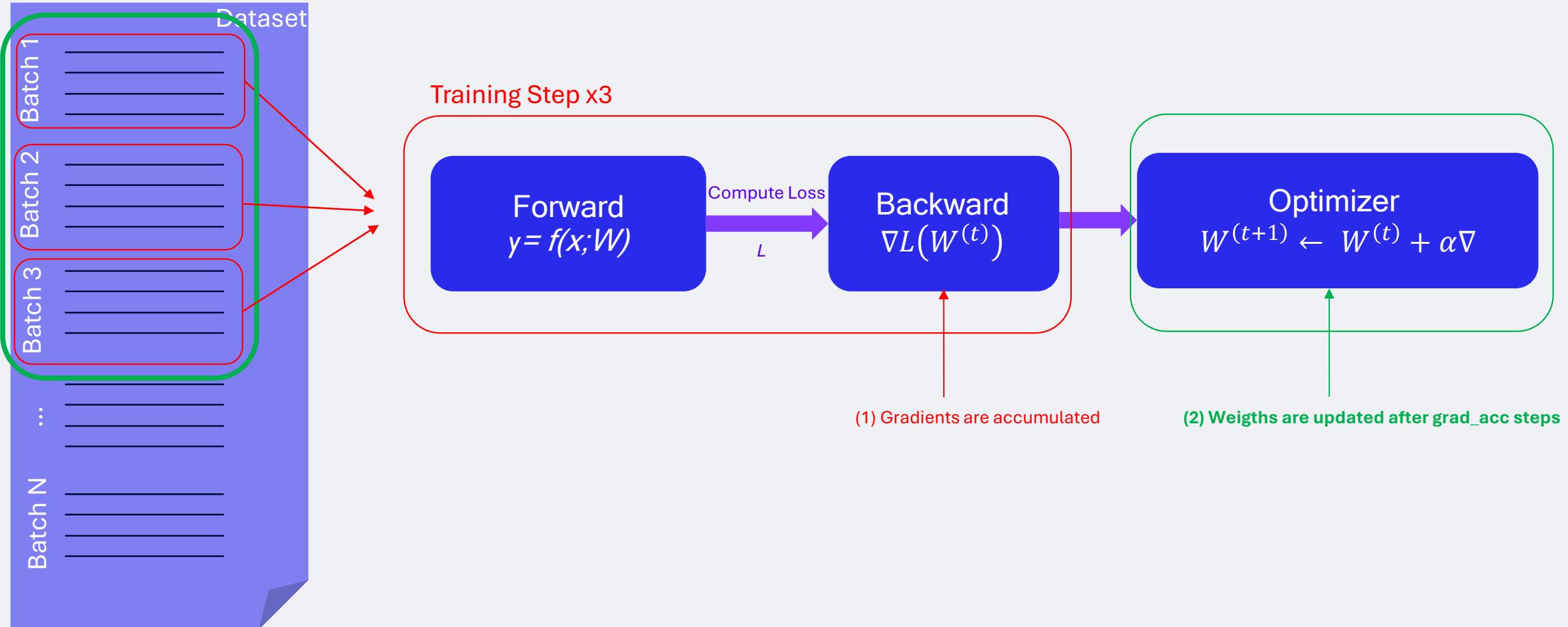
- Complex AI stack + interdependencies
- Simulators / performance models: expensive, vast search space.

LLM Training



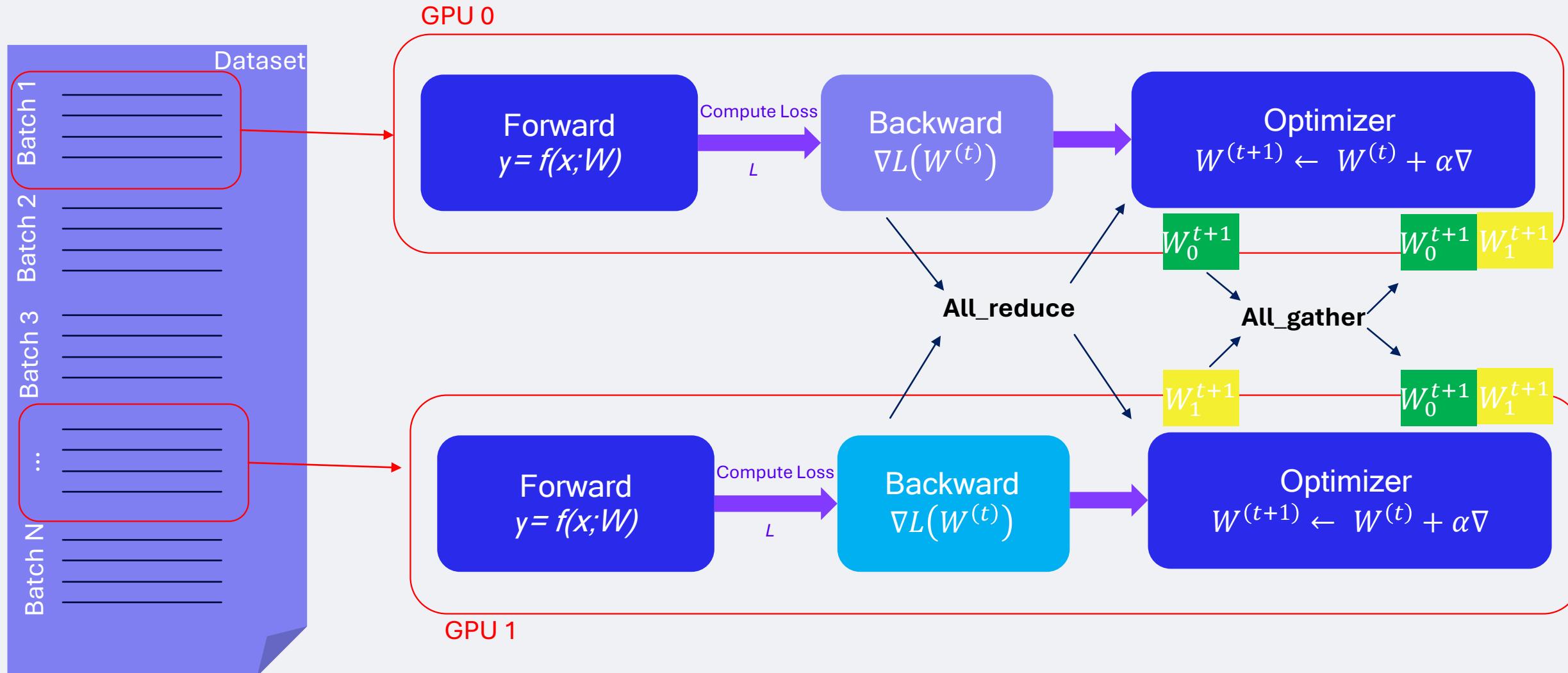
LLM Training

Gradient accumulation accumulates gradients and updates weights after a few steps



Example: Optimizer is partitioned across multiple nodes

Communication matters in multi-node



DeepSpeed Zero: Memory and communication trade-off

Different multi-node partitions for the model

Name	What's Distributed?	Comm. Collectives
Baseline	Data	all-reduce
ZeRO Stage 1	Data + Optimizer	reduce-scatter + all-gather
ZeRO Stage 2	Data + Opt. + Gradients	reduce-scatter + all-gather
ZeRO Stage 3	Data + Opt. + Grad. + Weights	all-gather + all-gather + reduce-scatter

Evaluation Setup



Used Platforms

- **IB-A100:** IB interconnect 1.6 Tbps, 8x A100 per node. NVlink3 600 GB/s.
- **RoCE-A100:** RoCE interconnect 1.6 Tbps, 8x A100 per node. NVlink3 600 GB/s.
- **RoCE-H100:** RoCE interconnect 1.6 Tbps, 8x H100 per node. NVlink4 900 GB/s.



Models

Models evaluated:

Model 1

- 440m
- LLaMA-based

Model 2

- 8 Billion
- LLaMA-based
- KD from 8B teacher

Evaluation Setup



Used Platforms

- **IB-A100:** IB interconnect 1.6 Tbps, 8x A100 per node. NVlink3 600 GB/s.
- **RoCE-A100:** RoCE interconnect 1.6 Tbps, 8x A100 per node. NVlink3 600 GB/s.
- **RoCE-H100:** RoCE interconnect 1.6 Tbps, 8x H100 per node. NVlink4 900 GB/s.



Models

Models evaluated:

Model 1

- 440m

Model 2

- 8 Billion

Goal: Optimize LLM training at scale

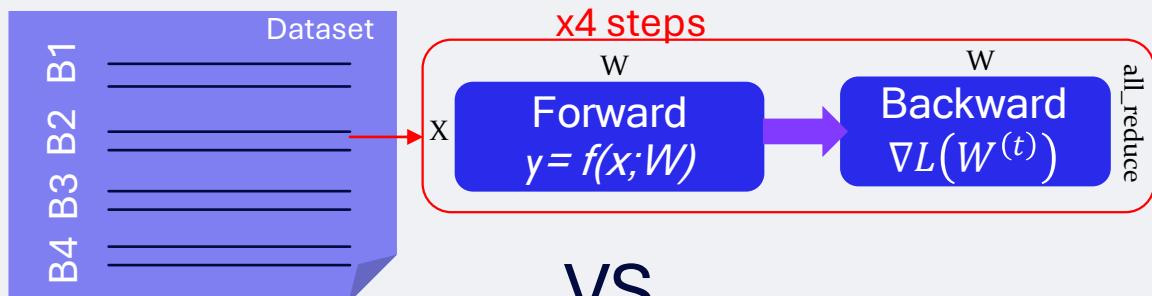
Contribution: Provide a tuning methodology that accelerates multi-node training **across different platforms**

Outline

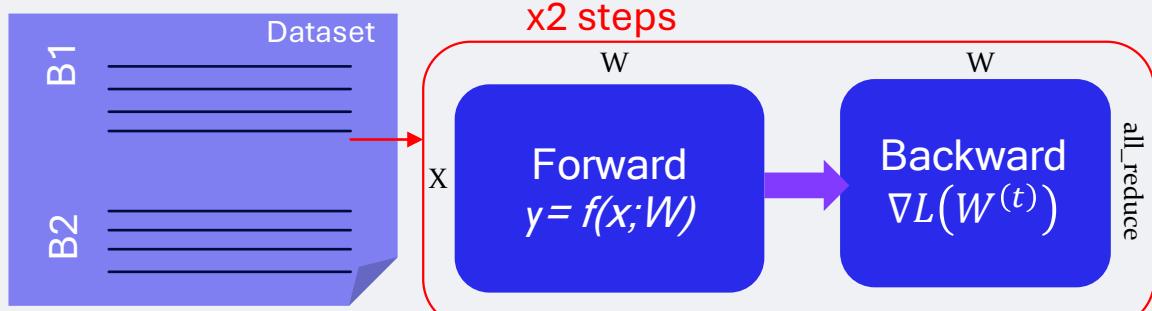
- Motivation
- LLM Training
- Used Platforms
- Performance Analysis (Model 1)
- Tuning Methodology
- Methodology Evaluation: Experimental (Model 2)
- Conclusions

Performance Analysis: Observations

Increasing batch size reduces the number of training steps (comm. collectives), but each step becomes more compute-intensive.



VS



As many gradients as weights. Therefore, **all-reduce sends the same amount** of data independently of batch size.

Preserve factor $C = \text{batch_size} * \text{grad_acc}$ constant for accuracy.

- Model 1 has $C=1024$

$$\text{e.g. } 1024 = 256 * 4 = 512 * 2 = 128 * 8$$

Time spent **on optimizer** is independent of the batch size, since it is only determined by model parameters

Optimizer
 $W^{(t+1)} \leftarrow W^{(t)} + \alpha \nabla$

Therefore, **all-gather sends the same amount** of weights independently of batch size.

Performance Analysis: ZeRO Stage 1

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Averaging time per step every 1024 samples:

Batch_size	Grad_acc	Gradient computation
64	16	123.6
128	8	233.8
256	4	452.8

Makes sense. Doubling batch size, doubles the computation

Performance Analysis: ZeRO Stage 1

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Averaging time per step every 1024 samples:

Batch_size	Grad_acc	Gradient computation	Gradient all-reduce	Optimizer all-gather
64	16	123.6	2.58	4.5
128	8	233.8	16.78	4.51
256	4	452.8	8.07	4.41

Makes sense. Amount of weights does not depend on batch size.

Performance Analysis: ZeRO Stage 1

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Averaging time per step every 1024 samples:

Batch_size	Grad_acc	Gradient computation	Gradient all-reduce	Optimizer all-gather
64	16	123.6	2.58	4.5
128	8	233.8	16.78	4.51
256	4	452.8	8.07	4.41

This should be constant.
Number of gradients does not
depend on batch size.

Performance Analysis: ZeRO Stage 1

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Averaging time per step every 1024 samples:

Batch_size	Grad_acc	Gradient computation	Gradient all-reduce	Optimizer all-gather
64	16	123.6	2.58	4.5
128	8	233.8	16.78	4.51
256	4	452.8	8.07	4.41

all-reduce only happens in the last step (8th) of Stage 1 implementation...



Let's break down step by step:

Operations	Total Time	Computation	Communication
bwd_microstep_0	232.46	232.46	0
bwd_microstep_1	234.32	234.23	0
bwd_microstep_2	233.90	233.84	0
bwd_microstep_3	235.06	235.00	0
bwd_microstep_4	235.61	235.56	0
bwd_microstep_5	234.09	233.92	0
bwd_microstep_6	233.82	233.76	0
bwd_microstep_7	290.12	235.80	54.25

We shouldn't just maximize batch_size.

TRADE-OFF between grad_acc and batch_size !

Nodes	Batch Size	Grad ac	ZeRO	Batch Time	Estimated Hours
2	32	32	1	0.106	158.68
2	64	16	1	0.197	147.05
2	128	8	1	0.398	148.05
2	256	4	1	0.786	147.08
2	512	2	1	Out of memory	

Performance Analysis: ZeRO Stage 1

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Averaging time per step every 1024 samples:

Batch_size	Grad_acc	Gradient computation	Gradient all-reduce	Optimizer all-gather
64	16			
128	8			
256	4			

Takeaway 1:

When using gradient accumulation and ZeRO Stage 1, maximize batch size is not always the best strategy.



Let's break down the numbers...

Operations	bwd_microstep_0	bwd_microstep_1	bwd_microstep_2	bwd_microstep_3	bwd_microstep_4	bwd_microstep_5	bwd_microstep_6	bwd_microstep_7	
	232.46	232.46			0				
	234.32	234.23			0				
	233.90	233.84			0				
	235.06	235.00			0				
	235.61	235.56			0				
	234.09	233.92			0				
	233.82	233.76			0				
	290.12	235.80			54.25				

ll-reduce only happens in the last step (8th) of Stage 1 implementation...

. the larger grad_acc, the fewer all-reduces.

We shouldn't just maximize **batch_size**.

TRADE-OFF between grad_acc and batch_size !

Nodes	Batch Size	Grad ac	ZeRO	Batch Time	Estimated Hours
2	32	32	1	0.106	158.68
2	64	16	1	0.197	147.05
2	128	8	1	0.398	148.05
2	256	4	1	0.786	147.08
2	512	2	1	Out of memory	

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Operations	Total Time	Computation	Communication
bwd_microstep_0	232.46	232.46	0
bwd_microstep_1	234.32	234.23	0
bwd_microstep_2	233.90	233.84	0
bwd_microstep_3	235.06	235.00	0
bwd_microstep_4	235.61	235.56	0
bwd_microstep_5	234.09	233.92	0
bwd_microstep_6	233.82	233.76	0
bwd_microstep_7	290.12	235.80	54.25

Stage 1

Operations	Total Time	Computation	Communication
bwd_microstep_0	465.04	452.29	12.67
bwd_microstep_1	471.9	453.98	17.87
bwd_microstep_2	472.84	450.67	22.07
bwd_microstep_3	470.91	450.05	20.79
bwd_microstep_4	473.14	449.91	23.18
bwd_microstep_5	484.4	449.41	24.93
bwd_microstep_6	467.27	454.16	13.05
bwd_microstep_7	471.32	449.75	21.54

Stage 2

When should we use Stage 1 and when Stage 2?

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Operations	Total Time	Computation	Communication
bwd_microstep_0	232.46	232.46	0
bwd_microstep_1	234.32	234.23	0
bwd_microstep_2	233.90	233.84	0
bwd_microstep_3	235.06	235.00	0
bwd_microstep_4	235.61	235.56	0
bwd_microstep_5	234.09	233.92	0
bwd_microstep_6	233.82	233.76	0
bwd_microstep_7	290.12	235.80	54.25

Stage 1

65 seconds every
grad_acc steps

Operations	Total Time	Computation	Communication
bwd_microstep_0	465.04	452.29	12.67
bwd_microstep_1	471.9	453.98	17.87
bwd_microstep_2	472.84	450.67	22.07
bwd_microstep_3	470.91	450.05	20.79
bwd_microstep_4	473.14	449.91	23.18
bwd_microstep_5	484.4	449.41	24.93
bwd_microstep_6	467.27	454.16	13.05
bwd_microstep_7	471.32	449.75	21.54

Stage 2

19 seconds every
step

$$19 \times \text{grad_acc} \leq 65$$

Stage 2 becomes potentially better when $\text{grad_acc} < 4$

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Stage 2 becomes potentially better when $\text{grad_acc} < 4$

Nodes	Batch Size	Grad ac	ZeRO	Batch Time	Estimated Hours
2	32	32	1	0.106	158.68
2	64	16		0.197	147.05
2	128	8		0.398	148.05
2	256	4		0.786	147.08
2	512	2		Out of memory	
2	32	32	2	0.118	176.64
2	64	16		0.211	157.93
2	128	8		0.393	150.08
2	256	4		0.752	140.71
2	512	2		Out of memory	

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Stage 2 becomes potentially better when $\text{grad_acc} < 4$

Nodes	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300	301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325	326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350	351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375	376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400	401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425	426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450	451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475	476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500	501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525	526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550	551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575	576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600	601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625	626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650	651	652	653	654	655	656	657	658	659	660	661	662	663	664	665	666	667	668	669	670	671	672	673	674	675	676	677	678	679	680	681	682	683	684	685	686	687	688	689	690	691	692	693	694	695	696	697	698	699	700	701	702	703	704	705	706	707	708	709	710	711	712	713	714	715	716	717	718	719	720	721	722	723	724	725	726	727	728	729	730	731	732	733	734	735	736	737	738	739	740	741	742	743	744	745	746	747	748	749	750	751	752	753	754	755	756	757	758	759	760	761	762	763	764	765	766	767	768	769	770	771	772	773	774	775	776	777	778	779	780	781	782	783	784	785	786	787	788	789	790	791	792	793	794	795	796	797	798	799	800	801	802	803	804	805	806	807	808	809	810	811	812	813	814	815	816	817	818	819	820	821	822	823	824	825	826	827	828	829	830	831	832	833	834	835	836	837	838	839	840	841	842	843	844	845	846	847	848	849	850	851	852	853	854	855	856	857	858	859	860	861	862	863	864	865	866	867	868	869	870	871	872	873	874	875	876	877	878	879	880	881	882	883	884	885	886	887	888	889	890	891	892	893	894	895	896	897	898	899	900	901	902	903	904	905	906	907	908	909	910	911	912	913	914	915	916	917	918	919	920	921	922	923	924	925	926	927	928	929	930	931	932	933	934	935	936	937	938	939	940	941	942	943	944	945	946	947	948	949	950	951	952	953	954	955	956	957	958	959	960	961	962	963	964	965	966	967	968	969	970	971	972	973	974	975	976	977	978	979	980	981	982	983	984	985	986	987	988	989	990	991	992	993	994	995	996	997	998	999	1000	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011	1012	1013	1014	1015	1016	1017	1018	1019	1020	1021	1022	1023	1024	1025	1026	1027	1028	1029	1030	1031	1032	1033	1034	1035	1036	1037	1038	1039	1040	1041	1042	1043	1044	1045	1046	1047	1048	1049	1050	1051	1052	1053	1054	1055	1056	1057	1058	1059	1060	1061	1062	1063	1064	1065	1066	1067	1068	1069	1070	1071	1072	1073	1074	1075	1076	1077	1078	1079	1080	1081	1082	1083	1084	1085	1086	1087	1088	1089	1090	1091	1092	1093	1094	1095	1096	1097	1098	1099	1100	1101	1102	1103	1104	1105	1106	1107	1108	1109	1110	1111	1112	1113	1114	1115	1116	1117	1118	1119	1120	1121	1122	1123	1124	1125	1126	1127	1128	1129	1130	1131	1132	1133	1134	1135	1136	1137	1138	1139	1140	1141	1142	1143	1144	1145	1146	1147	1148	1149	1150	1151	1152	1153	1154	1155	1156	1157	1158	1159	1160	1161	1162	1163	1164	1165	1166	1167	1168	1169	1170	1171	1172	1173	1174	1175	1176	1177	1178	1179	1180	1181	1182	1183	1184	1185	1186	1187	1188	1189	1190	1191	1192	1193	1194	1195	1196	1197	1198	1199	1200	1201	1202	1203	1204	1205	1206	1207	1208	1209	1210	1211	1212	1213	1214	1215	1216	1217	1218	1219	1220	1221	1222	1223	1224	1225	1226	1227	1228	1229	1230	1231	1232	1233	1234	1235	1236	1237	1238	1239	1240	1241	1242	1243	1244	1245	1246	1247	1248	1249	1250	1251	1252	1253	1254	1255	1256	1257	1258	1259	1260	1261	1262	1263	1264	1265	1266	1267	1268	1269	1270	1271	1272	1273	1274	1275	1276	1277	1278	1279	1280	1281	1282	1283	1284	1285	1286	1287	1288	1289	1290	1291	1292	1293	1294	1295	1296	1297	1298	1299	1300	1301	1302	1303	1304	1305	1306	1307	1308	1309	1310	1311	1312	1313	1314	1315	1316	1317	1318	1319	1320	1321	1322	1323	1324	1325	1326	1327	1328	1329	1330	1331	1332	1333	13

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Operations	Total Time	Computation	Communication
bwd_microstep_0	232.46	232.46	0
bwd_microstep_1	234.32	234.23	0
bwd_microstep_2	233.90	233.84	0
bwd_microstep_3	235.06	235.00	0
bwd_microstep_4	235.61	235.56	0
bwd_microstep_5	234.09	233.92	0
bwd_microstep_6	233.82	233.76	0
bwd_microstep_7	290.12	235.80	54.25

Stage 1

Operations	Total Time	Computation	Communication
bwd_microstep_0	465.04	452.29	12.67
bwd_microstep_1	471.9	453.98	17.87
bwd_microstep_2	472.84	450.67	22.07
bwd_microstep_3	470.91	450.05	20.79
bwd_microstep_4	473.14	449.91	23.18
bwd_microstep_5	484.4	449.41	24.93
bwd_microstep_6	467.27	454.16	13.05
bwd_microstep_7	471.32	449.75	21.54

Stage 2

Regardless number of all-reduces, why Stage 1's all-reduce is slower than Stage 2's?

NCCL Profiler:

- (1) Stage 1 is calling two NCCL's collectives, doubling latency.
- (2) Stage 1 and Stage 2 are using NCCL's all-reduce

Performance Analysis: ZeRO Stage 2

Testing different $1024 = \text{batch_size} * \text{grad_acc}$ values for Model 1 on IB-A100

Operations	Total Time	Computation	Communication	Operations	Total Time	Computation	Communication
bwd_microstep_0	232.46	232.46	0	bwd_microstep_0	465.04	452.29	12.67
bwd_microstep_1	234.32	234.23	0	bwd_microstep_1	471.9	453.98	17.87
bwd_microstep_2					472.84	450.67	22.07
bwd_microstep_3					470.91	450.05	20.79
bwd_microstep_4					473.14	449.91	23.18
bwd_microstep_5					484.4	449.41	24.93
bwd_microstep_6					467.27	454.16	13.05
bwd_microstep_7					471.32	449.75	21.54
Stage 1							

Takeaway 3:

*Contrary to claims made in DS paper,
Stage 1 and 2 do not implement reduce-scatter for gradients but all-reduce.*

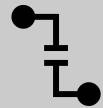
Regardless number of all-reduces, why Stage 1's all-reduce is slower than Stage 2's?

NCCL Profiler:

- (1) Stage 1 is calling two NCCL's collectives, doubling latency.
- (2) Stage 1 and Stage 2 are using NCCL's all-reduce

Tuning Methodology: Communication-aware

Collectives control overall performance fluctuation



Isolate collective performance in the system.

- Run standalone NCCL all-reduce test and tune its perf. parameters (NCCL_ALGO, NCCL_SOCKET_NTHREADS, NCCL_MIN_NCHANNELS, etc)
- Empirical search if doable; Bayesian Optimization otherwise



Explore parallelism strategies

- Evaluate different *grad_acc*, *batch_size* and ZeRO stages
- Empirical search if doable; Bayesian Optimization otherwise



Tuning of other DS communication-related parameters

- *overlap_comm*
- Hack DeepSpeed code to use *reduce-scatter**

Takeaway 4:

Tuning NCCL performance parameters yields minimal performance gains at moderate scale, but beneficial at larger scales.

Outline

- Motivation
- LLM Training
- Used Platforms
- Performance Analysis (Model 1)
- Tuning Methodology
- Methodology Evaluation: Experimental (Model 2)
- Conclusions

Methodology Evaluation

Testing methodology for training Model 2 on three platforms

$$C = 256 = \text{batch_size} \times \text{grad_acc}$$

Step 1. Tune standalone NCCL all-reduce

Category	NCCL Parameter
Algorithm Selection	NCCL_ALGO
Cross-NIC Communication	NCCL_CROSS_NIC
Threading Parameters	NCCL_NTHREADS, NCCL_SOCKET_NTHREADS
Socket/Channel Configuration	NCCL_NSOCKS_PERTHREAD, NCCL_MIN_NCHANNELS
Interface and Stack Settings	NCCL_SOCKET_IFNAME, NCCL_IB_DISABLE, NCCL_CHECK_DISABLE, NCCL_SET_STACK_SIZE

Methodology Evaluation

Testing methodology for training Model 2 on three platforms

$$C = 256 = \text{batch_size} \times \text{grad_acc}$$

Step 1. Tune standalone NCCL all-reduce

Category	NCCL Parameter
Algorithm Selection	NCCL_ALGO
Cross-NIC Communication	NCCL_CROSS_NIC
Threading Parameters	NCCL_NTHREADS, NCCL_SOCKET_NTHREADS
Socket/Channel Configuration	NCCL_NSOCKS_PERTHREAD, NCCL_MIN_NCHANNELS
Interface and Stack Settings	NCCL_SOCKET_IFNAME, NCCL_IB_DISABLE, NCCL_CHECK_DISABLE, NCCL_SET_STACK_SIZE

Step 2. Evaluate parallelism strategies

Platform	DS Stage	Batch Size	Grad Acc	Samples/s
IB-A100	1	64	4	66.78
	1	128	2	OOM
	2	64	4	62.131
	2	128	2	76.489
	2	256	1	OOM
	3	64	4	23.680
	3	128	2	39.862
	3	256	1	49.131
	1	64	4	58.51
RoCE-A100	1	128	2	OOM
	2	64	4	25.741
	2	128	2	72.891
	2	256	1	OOM
	3	64	4	18.89
	3	128	2	35.33
	3	256	1	47.89
RoCE-H100	1	64	4	92.35
	1	128	2	OOM
	2	64	4	84.07
	2	128	2	134.00
	2	256	1	OOM
	3	64	4	25.12
	3	128	2	44.76
	3	256	1	68.53

In all platforms, grad_acc = 2 and Stage 2 got best results.

Methodology Evaluation

Testing methodology for training Model 2 on three platforms

$$C = 256 = \text{batch_size} \times \text{grad_acc}$$

Step 3. Tune DS other communication-related parameters.

Platform	Baseline samples/s	Tuning samples/s	Overlap_comm?	Best Collective
IB-A100	65.78	90.71	Disabled	Reduce-Scatter
RoCE-A100	58.51	79.9	Disabled	Reduce-Scatter
RoCE-H100	92.35	146.37	Enabled	Reduce-Scatter

* Baseline config. Uses ZeRO Stage 1 with the largest batch size possible, the default all-reduce collective and overlap_comm enable

Methodology Evaluation

Testing methodology for training Model 2 on three platforms

$$C = 256 = \text{batch_size} \times \text{grad_acc}$$

Step 3. Tune DS other communication-related parameters.

Platform	Baseline sample
IB-A100	65.78
RoCE-A100	58.51
RoCE-H100	92.35

Takeaway 5:

Performance tuning is necessary, and each model and platform require separated tuning searches.

Methodology Evaluation

Testing methodology for training Model 2 on three platforms

$$C = 256 = \text{batch_size} \times \text{grad_acc}$$

Step 3. Tune DS other communication-related parameters.

Platform	Baseline samples/s	Tuning samples/s	Overlap_comm?	Best Collective
IB-A100	65.78	90.71	Disabled	Reduce-Scatter
RoCE-A100	58.51	79.9	Disabled	Reduce-Scatter
RoCE-H100	92.35	146.37	Enabled	Reduce-Scatter

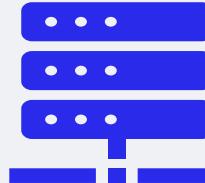
- 1) H100 platform trained $1.85x$ faster than A100s
- 2) Tuning communication-related parameters boost training throughput
- 3) Default *all-reduce* does not provide best results in any platform.
- 4) Optimal configuration varies by platform.
- 5) IB-A100 platform slightly outperforms RoCE-A100.
- 6) Disabling RDMA and using TCP led to avg. $12x$ slowdown for backward pass.

Conclusions

Our findings challenge many common assumptions in LLM training.



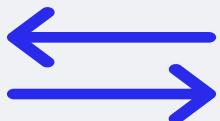
Maximize batch size is
not always the best
strategy



RoCE-based clusters can achieve
similar performance to IB-based. TCP
is a perf. killer

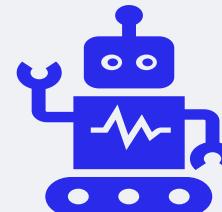


Achieving full compute and
communication overlapping
remains a fundamental challenge

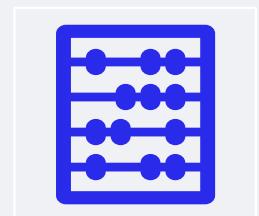


Communication collectives
are responsible for most
overall performance
fluctuation

Default DeepSpeed
implementation uses *all-*
reduce for gradient
reduction



Separated performance
tuning is required per
model and platform



Our 3-step methodology
delivered 1.6x over
baseline

Thank you

Nothing in these materials is an offer to sell any of the components or devices referenced herein.

© Qualcomm Technologies, Inc. and/or its affiliated companies. All Rights Reserved.

Qualcomm and Snapdragon are trademarks or registered trademarks of Qualcomm Incorporated.

Other products and brand names may be trademarks or registered trademarks of their respective owners.

References in this presentation to “Qualcomm” may mean Qualcomm Incorporated, Qualcomm Technologies, Inc., and/or other subsidiaries or business units within the Qualcomm corporate structure, as applicable. Qualcomm Incorporated includes our licensing business, QTL, and the vast majority of our patent portfolio. Qualcomm Technologies, Inc., a subsidiary of Qualcomm Incorporated, operates, along with its subsidiaries, substantially all of our engineering, research and development functions, and substantially all of our products and services businesses, including our QCT semiconductor business.

Snapdragon and Qualcomm branded products are products of Qualcomm Technologies, Inc. and/or its subsidiaries. Qualcomm patents are licensed by Qualcomm Incorporated.

Follow us on: [in](#) [X](#) [@](#) [YouTube](#) [f](#)

For more information, visit us at qualcomm.com & qualcomm.com/blog

