

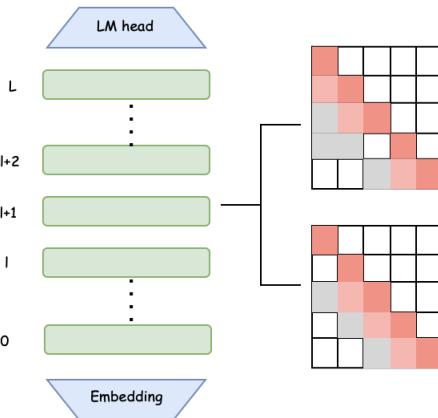
000 DENSE2MOE: UNIFYING PRUNING AND UPCYCLING FOR EFFI- 001 CIENT LARGE LANGUAGE MODELS 002

003 **Anonymous authors**

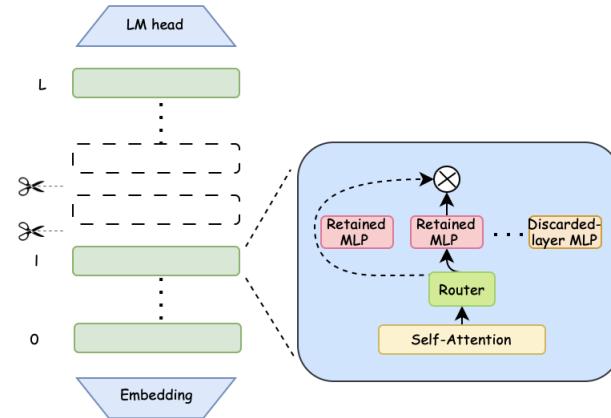
004 Paper under double-blind review

009 ABSTRACT

011 The Mixture of Experts (MoE) architecture has become a mainstream design in Large Language
012 Models (LLMs) for its ability to flexibly scale parameters while maintaining inference efficiency.
013 However, training MoE models from scratch remains prohibitively expensive due to their high com-
014 putational demands. Existing upcycling methods reduce costs by converting dense LLMs into MoEs
015 through layer duplication and fine-tuning, but introduce substantial redundancy. Layer-wise prun-
016 ing is commonly used to alleviate redundancy among the decoder layers of dense models, but it
017 inevitably incurs performance degradation. In this paper, we propose Dense2MoE, a novel approach
018 that unifies layer pruning and upcycling through a technique we term Layer-Fusion UpCycling(LF-
019 UC). Our method prunes highly redundant layers in an LLM while retaining their MLPs in the form
020 of MoE. In this way, tokens are routed through a subset of redundant MLP layers rather than all of
021 them. This design efficiently leverages open-source LLMs with low additional computational over-
022 head, enhancing model performance while reducing active parameters. Extensive experiments show
023 that Dense2MoE effectively pushes the Pareto frontier of efficiency versus performance toward a
024 more optimal region compared with the original seed models, and achieves a superior trade-off rel-
025 ative to alternative approaches.



039 (a) inter-layer similarity analysis



040 (b) prune and expand

041 **Figure 1: Overview of Dense2MoE.** (a) inter-layer similarity analysis: we analyze the output similarity of each
042 decoder layer in the LLM and the similarity of inputs to each Multi-Layer Perceptron (MLP). (b) Pruning and Layer-
043 Fusion Upcycling (LF-UC): We prune the attention modules of layers identified as highly redundant based on the
044 aforementioned similarity metrics. Instead of discarding the MLPs from these pruned layers, we fuse them into the
045 MLP layers of the retained transformer blocks.

049 1 INTRODUCTION

050 The scaling law(Kaplan et al. (2020)) of large language models (LLMs) has established that as the model size and
051 training dataset size continue to increase, the model’s capabilities improve consistently. Consequently, a growing body
052 of open-source work(Yang et al. (2025); Touvron et al. (2023); Jung et al. (2010); Tripti R et al. (2025)) has focused

on developing models with larger parameter sizes. However, the continuous expansion of model scale imposes higher requirements on hardware and poses challenges for deployment in resource-constrained hardware environments in real-world scenarios.

To more fully leverage the knowledge stored in open-source models and reduce resource waste caused by model retraining, many studies (Gromov et al. (2024); Yang et al. (2024); Ashkboos et al. (2024); Men et al. (2024); Song et al. (2024); Kim et al. (2024); Chen et al. (2024)) have opted to prune large-parameter models. These techniques remove redundant parameters or components in the model and can often ensure near-optimal performance in specific task domains. Another approach is parameter expansion, which increases the model’s parameter count by extending the number of model layers or adopting the Mixture of Experts (MoE)(Shazeer et al. (2017)) framework—typically trading a small increase in active parameters for better model performance.

Nevertheless, existing methods have inherent limitations: Unstructured model pruning requires targeted operator optimization to accelerate sparse matrix computations, while structured pruning often leads to performance degradation due to the loss of critical model parameters. Meanwhile, parameter expansion methods (Komatsuzaki et al. (2022); Sukhbaatar et al. (2024); Wu et al. (2024)) increase the model’s total parameter count, which in turn causes a multiplicative growth in redundant parameters.

To balance model performance and efficiency, we propose Dense2MoE—a novel method that jointly combines model pruning and parameter expansion. It enables open-source models to deliver better performance across diverse hardware platforms at extremely low training costs. An overview of Dense2MoE is illustrated in Fig.1: First, we analyze the output similarity between different decoder layers of the LLM and the similarity of outputs from self-attention modules. Based on this analysis, we prune highly redundant layers while retaining their MLPs to serve as experts within an MoE framework. Specifically, through our Layer-Fusion Upcycling (LF-UC) technique, we expand the MLPs of the retained layers into MoE structures, incorporating the upcycled MLPs as additional experts. Finally, we perform a small amount of finetuning to adapt the modified model structure and parameters.

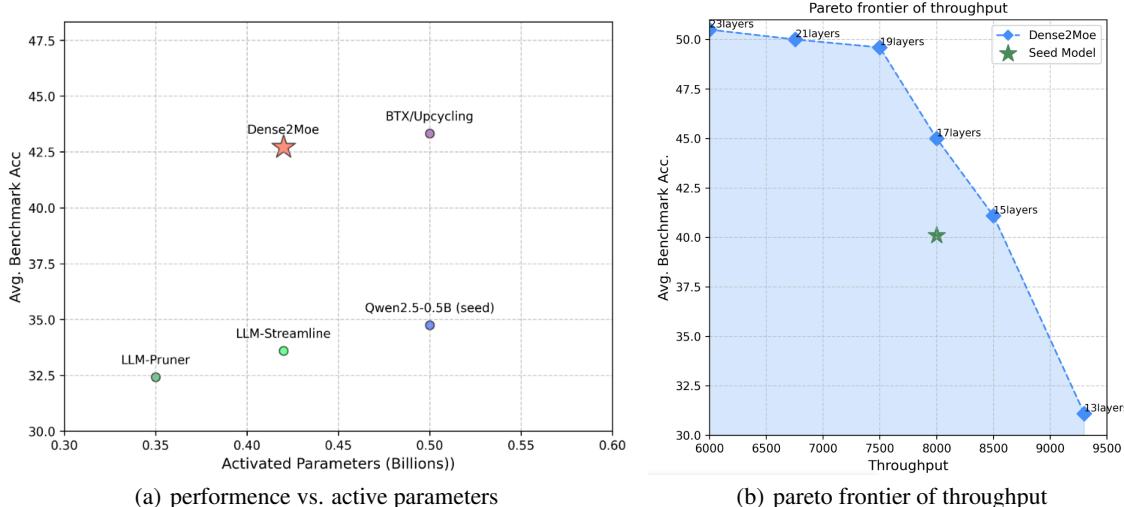


Figure 2: **Performance Comparison with the Seed Model and Alternative Methods.** (a) performance vs. active parameters mathematics, code, reasoning and general knowledge benchmarks, among different pruning methods and parameter upcycling. (b) pareto frontier of throughput

Furthermore, most existing model pruning works evaluate accuracy based on metrics such as perplexity (PPL), validation loss, or benchmarks in general knowledge. In contrast, we comprehensively evaluate our Dense2MoE across benchmarks spanning mathematics, code, and general knowledge domains. This multi-domain assessment fully validates the practical applicability of our method in real-world scenarios.

In summary, our key contributions are as follows:

- We propose a simple yet novel framework for leveraging open-source models, improving the Pareto frontier of efficiency versus accuracy when compared with seed models. Dense2MoE achieved better performance with fewer activated parameters.

108

- We conduct comprehensive evaluations across benchmarks in general knowledge, code, and mathematics

109 domains. The results demonstrate that our method achieves an optimal balance among approaches focused

110 on model structured pruning and model expansion.

111

- We validate the scalability of our method across open-source models of different scales and model families.

112

113

2 METHOD

114

115 Dense2MoE adopt a prune-and-expand approach to reuse open-source models and transform them into MoE, aiming

116 to reduce active parameters while improving performance. As illustrated in Fig.1. Our method can be summarized into

117 two key steps: (1) identify layers where both the decoder layer outputs and MLP inputs are similar; (2) retain the first

118 layer that meets the condition in (1), drop all subsequent layers, and retain their MLPs in the first layer as experts.

119 After these two steps, we perform fine-tuning on a small volume of data for the modified model.

120

121

2.1 PRELIMINARY

122

123 **Forward Propagation in LLMs.** Modern decoder-only large language models (LLMs), such as GPT, primarily stack

124 transformer blocks—composed of attention mechanisms and multi-layer perceptron (MLP) layers—along the depth

125 dimension. Specifically, the inter-layer information transmission in the transformer architecture is given by a residual

126 iteration equation

127

$$x^{(l+1)} = x^{(l)} + f(x^{(l)}, \theta^{(l)}) \quad (1)$$

128

129 Where $x^{(l)}$ denotes the hidden state input to the i -th transformer block, and $\theta^{(l)}$ represents the parameters of the i -th

130 layer. The i -th layer function f contributes a transformation of $f(x, \theta)$ to $x^{(l)}$, which consists of a multi-head attention

131 (MHA) layer and a MLP layer

132

$$f(x^{(l)}, \theta^{(l)}) = \text{MHA}^{(l)}(x^{(l)}, \theta_{mha}^{(l)}) + \text{MLP}^{(l)}(h^{(l)}, \theta_{mlp}^{(l)}) \quad (2)$$

133

134 Where, $h^{(l)}$ denotes the input to the MLP

135

$$h^{(l)} = x^{(l)} + \text{MHA}^{(l)}(x^{(l)}, \theta_{mha}^{(l)}) \quad (3)$$

136

137 **Mixture of Experts (MoE).** Mixture of Experts models expand the model parameter count by replacing the MLPs

138 in traditional Transformers with MoE layers. Owing to their sparse activation property, MoE models can typically

139 achieve better performance than dense models while activating fewer parameters.

140

141 Each MoE layer comprises a router which assigns the input token representation $h^{(l)}$ to N MLP experts via a router

142 network, parameterized by $W_g \in \mathbb{R}^{d_{\text{model}} \times N}$. The output of the MoE layer is computed as the weighted sum of outputs

143 from the top- k experts with the highest probabilities assigned by the router. Denoting the set of selected top- k indices

144 as K

145

$$\text{MLP}_{\text{MoE}}^l = \sum_{i \in K} g_i(h^{(l)}) * \text{MLP}^{(i)}(h^{(l)}) g_i(h^{(l)}) = \text{softmax}^{(i)}(h^{(l)} W_g)$$

146

147

2.2 INTER-LAYER SIMILARITY ANALYSIS

148

149 Analyzing the similarity between transformer layers is crucial for our pruning-and-expansion method. We use cosine

150 similarity to measure the similarity between hidden states. The similarity between hidden state X_1 and hidden state

151 X_2 can be expressed as

152

$$s(\mathbf{X}_1, \mathbf{X}_2) = \frac{\mathbf{X}_1 \cdot \mathbf{X}_2}{\|\mathbf{X}_1\| \cdot \|\mathbf{X}_2\|} \quad (4)$$

153

154 and the distance $d(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)})$ between the MLP input $h^{(l)}$ of the l -th layer and the MLP input $h^{(l+n)}$ of the $l+n$ -th

155 layer.

156

$$s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)}) = \frac{\mathbf{h}^{(l)} \cdot \mathbf{h}^{(l+n)}}{\|\mathbf{h}^{(l)}\| \cdot \|\mathbf{h}^{(l+n)}\|} \quad (5)$$

157

158 In this way, we obtain two distance lists $S_L = \{s(\mathbf{x}^{(l)}, \mathbf{x}^{(l+n)}) \mid l \in \{0, 1, \dots, N-2\}, n \in \{1, 2, \dots, N-1\}\}$

159 and $S_M = \{s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)}) \mid l \in \{0, 1, \dots, N-2\}, n \in \{1, 2, \dots, N-1\}\}$. Similar to many previous works, we

160 hypothesize that layers with greater inter-layer output similarity $s(\mathbf{x}^{(l)}, \mathbf{x}^{(l+n)})$ exhibit higher redundancy. However,

161 instead of pruning all components of these layers as prior methods did, we retain their MLPs—the components more

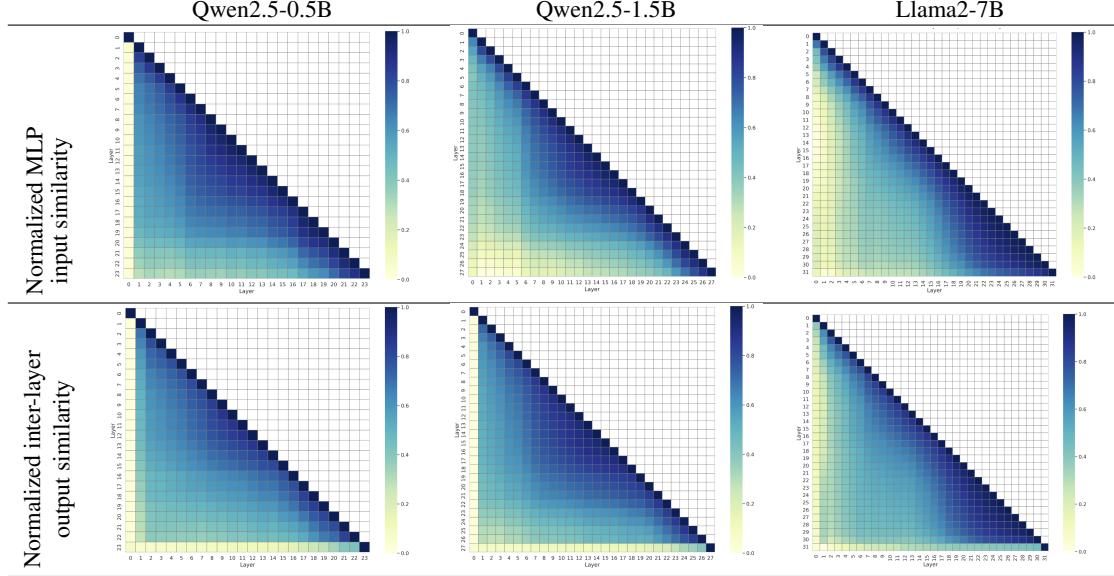


Figure 3: Normalized similarity comparison across different models.

critical to model performance. For layers with high MLP input similarity $s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)})$, we adopt a strategy of sharing attention layers (Section 2.3).

We visualize S_L and S_M across all layers for Qwen2.5-0.5B, Qwen2.5-1.5B, and LLaMA2-7B. These similarity metrics serve as prior knowledge for our pruning-and-expansion approach. In Fig.3, the color of each grid cell represents the distance between the hidden states of different layers. Layers with greater similarity exhibit relatively higher redundancy. We perform pruning-and-expansion at appropriate positions with reference to these similarities.

2.3 PRUNING AND EXPANSION

After the detailed analysis in Section 2.2, we obtain a set of output similarities S_L for each decoder layer and a set of similarities S_M for each MLP input. We consider that decoder layers with higher output similarity $s(\mathbf{x}^{(l)}, \mathbf{x}^{(l+n)})$ have higher redundancy, an idea that has also been confirmed by Gromov et al. (2024). Inspired by previous studies(He et al. (2024)), where MLP layers serve as knowledge repositories and pruning MLP layers has a significant impact on performance, while attention layers exhibit higher redundancy. We prune layers in S_L that have high values and also show approximate MLP inputs (i.e., $s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)})$). By sharing the attention mechanism from previous layers, the MLPs of redundant layers are retained as optional paths (Fig. 4).

Specifically, we sort S_L in descending order and search for l^* and n^* in S_L that satisfy $s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)}) < \delta$, where δ is a hyperparameter. We expand the parameters in the form of MoE: we replicate the MLP of layer l^* N times to serve as N experts, and replicate the MLPs from layer l^* to $l^* + n^*$ M times each, incorporating these $n^* * M$ experts into layer l^* . That is,

$$x^{(l^*+n^*)} = x^{l^*-1} + \text{MHA}^{(l^*)}(x^{(l^*-1)}) + \sum_{i=1}^K \alpha_i * \text{MLP}^{(l^*)}(h^{(l^*)}) + \sum_{k=1}^{n^*} \sum_{j=1}^M \beta_j * \text{MLP}^{(l^*+k)}(h^{(l^*)}) \quad (6)$$

Evidently, our method prunes the attention layers from layer $l^* + 1$ to $l + n^*$ and shares the hidden state $h^{(l^*)}$ of the MLP input at layer l^* . Due to the multi-head and redundant nature of the attention mechanism, pruning attention layers with high similarity does not result in significant performance degradationHe et al. (2024). Since we ensure the condition $s(\mathbf{h}^{(l)}, \mathbf{h}^{(l+n)}) < \delta$ during merging, the MLPs from layer $l^* + 1$ to $l + n^*$, when serving as experts, can restore the previous outputs as much as possible.

Our method decouples various structures within the main path of the LLM, enabling the model to learn and select network structures via residual paths. This approach mitigates redundancy arising from the serial arrangement of model layers and significantly enhances the scalability of the model's parameter count through replication.

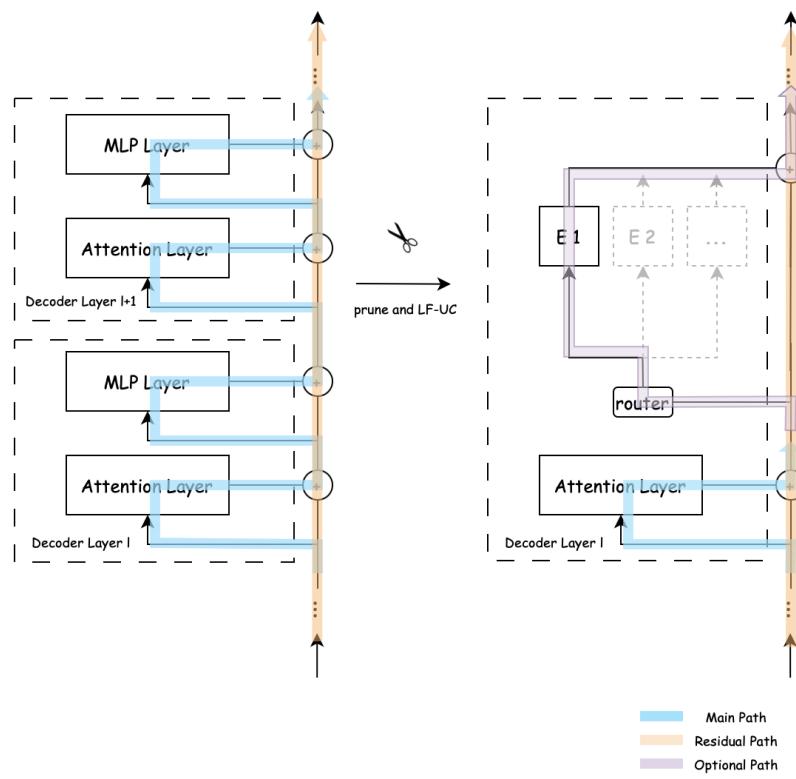


Figure 4: Tokens skip redundant layers through path selection.

3 EXPERIMENTS

We validated the Dense2MoE method on models of different scales across two open-source model families (Qwen2.5Team (2024) and Llama2Touvron et al. (2023)), and compared it with prevalent model layer pruning and upcycling approaches. For fairness, all methods were fine-tuned on a small-scale dataset of equal size, and their performance was validated on benchmarks spanning multiple domains (mathematics, code, reasoning and general knowledge).

3.1 SETUP

Considering the application scenarios of edge-side models, we conducted our experiments on Qwen2.5-0.5B(Team (2024)). To verify the generality of our method, we further extended it to models of larger scales and different families (Qwen2.5-1.5B(Team (2024)), Llama2-7B(Touvron et al. (2023))). Unless otherwise specified, our experimental settings are as follows: the modified model was trained with Continual Pre-Training on a dataset of 180B tokens, using a batch size corresponding to 40 million tokens. For the learning rate schedule, we implemented a warm-up phase from 0 to 1e-4 over 360 million tokens, followed by a cosine decay to 1e-5 after reaching the peak value.

In addition, we constructed a 225B-token dataset with the same composition to compare model performance under the same training resource consumption as the upcycling approach.

We compared our model with several mainstream works on model layer pruning and parameter expansion across multiple benchmarks. Specifically, for the general knowledge domain, we validated the superiority of our method on C-Eval(Huang et al. (2023)), CMMU(Li et al. (2023)), and MMLU(Hendrycks et al. (2020)). To compare the mathematical and coding capabilities of different methods, we conducted evaluations on GSM8K(Liu et al. (2023)), CMath(Wei et al. (2023)) (mathematics), HumanEval(Chen et al. (2021)) and MBPP(Austin et al. (2021)) (coding). For assessing the model's reasoning ability, we adopted BBH(Suzgun et al. (2022)) and ARC-Challenge(Clark et al. (2018)).

270
271 Table 1: Domain data sources and composition.
272
273
274
275
276
277
278
279
280
281

Domain	Dataset	Sampling ratio (%)
Math	OpenWebMath	12.0
	Arxiv	14.0
	Github	2.3
Code	Code	16.2
	Synthetic Data	3.8
General Knowledge	Wikipedia	50.6
	COIG	6.8
	C4	2.6

282
283
284 3.2 DATASET DETAILS
285
286
287
288
289
290
291

Our dataset is primarily assembled from publicly available, open-source sources, which we meticulously cleaned, normalized, and deduplicated to ensure high data quality. After processing, the dataset comprises approximately 180 billion tokens, which serve as the foundation for fine-tuning our models. To better align the Dense2MoE-modified architecture with diverse downstream tasks, we carefully adjusted the proportions of data from different domains, thereby enabling targeted re-adaptation of the model parameters. This approach ensures that each domain is adequately represented during fine-tuning, improving both generalization and task-specific performance. A detailed breakdown of the dataset composition, including the domain-specific proportions and token counts, is provided in Table 1.

292
293 3.3 MAIN RESULTS
294
295
296
297
298
299

Compared with the seed model, Dense2MoE advances the Pareto frontier of performance versus throughput. We measured throughput on a single NVIDIA A800 GPU (80GB HBM2e VRAM) with a batch size of 32, a prompt length of 1023, and a maximum generation length of 256 tokens. As shown in Fig. 2(b), under the same settings, Dense2MoE achieves a superior efficiency-performance trade-off after pruning different numbers of decoder layers. At comparable throughput, Dense2MoE delivers a 12.5% performance improvement (40.1 → 45).

300
301 Table 2: Performance comparison of different methods across various benchmarks
302
303

Method	Activated Parameters	Benchmarks								Avg.
		CEVAL	CMMLU	MMLU	GSM8K	CMath	HUMANEVAL	MBPP	BBH	
Qwen2.5-0.5B	0.5B	53.77	52.06	47.6	38.59	30.66	28.66	29.6	29.85	26.96
UIDL	0.39B	43.63	45.28	36.1	26.54	30.5	27.44	21.8	27.5	23.29
LLM-Pruner	0.4B	42.87	44.79	37.2	34.7	34.59	24.13	22.6	26.43	24.5
LLM-Streamline	0.4B	45.75	45.94	47.62	34.17	28.51	25.63	20.76	29.2	26.34
Dense2MoE	0.4B	59.68	57.8	44.7	43.37	43	36.59	41	33.79	24.57

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
5510
5511
5512
5513
5514
5515
5516
5517
5518
5519
5520
5521
5522
5523
5524
5525
5526
5527
5528
5529
5530
5531
5532
5533
5534
5535
5536
5537
5538
5539
55310
55311
55312
55313
55314
55315
55316
55317
55318
55319
55320
55321
55322
55323
55324
55325
55326
55327
55328
55329
55330
55331
55332
55333
55334
55335
55336
55337
55338
55339
55340
55341
55342
55343
55344
55345
55346
55347
55348
55349
55350
55351
55352
55353
55354
55355
55356
55357
55358
55359
55360
55361
55362
55363
55364
55365
55366
55367
55368
55369
55370
55371
55372
55373
55374
55375
55376
55377
55378
55379
55380
55381
55382
55383
55384
55385
55386
55387
55388
55389
55390
55391
55392
55393
55394
55395
55396
55397
55398
55399
553100
553101
553102
553103
553104
553105
553106
553107
553108
553109
553110
553111
553112
553113
553114
553115
553116
553117
553118
553119
553120
553121
553122
553123
553124
553125
553126
553127
553128
553129
553130
553131
553132
553133
553134
553135
553136
553137
553138
553139
553140
553141
553142
553143
553144
553145
553146
553147
553148
553149
553150
553151
553152
553153
553154
553155
553156
553157
553158
553159
553160
553161
553162
553163
553164
553165
553166
553167
553168
553169
553170
553171
553172
553173
553174
553175
553176
553177
553178
553179
553180
553181
553182
553183
553184
553185
553186
553187
553188
553189
553190
553191
553192
553193
553194
553195
553196
553197
553198
553199
553200
553201
553202
553203
553204
553205
553206
553207
553208
553209
553210
553211
553212
553213
553214
553215
553216
553217
553218
553219
553220
553221
553222
553223
553224
553225
553226
553227
553228
553229
553230
553231
553232
553233
553234
553235
553236
553237
553238
553239
553240
553241
553242
553243
553244
553245
553246
553247
553248
553249
553250
553251
553252
553253
553254
553255
553256
553257
553258
553259
553260
553261
553262
553263
553264
553265
553266
553267
553268
553269
553270
553271
553272
553273
553274
553275
553276
553277
553278
553279
553280
553281
553282
553283
553284
553285
553286
553287
553288
553289
553290
553291
553292
553293
553294
553295
553296
553297
553298
553299
553300
553301
553302
553303
553304
553305
553306
553307
553308
553309
553310
553311
553312
553313
553314
553315
553316
553317
553318
553319
553320
553321
553322
553323
553324
553325
553326
553327
553328
553329
553330
553331
553332
553333
553334
553335
553336
553337
553338
553339
553340
553341
553342
553343
553344
553345
553346
553347
553348
553349
553350
553351
553352
553353
553354
553355
553356
553357
553358
553359
553360
553361
553362
553363
553364
553365
553366
553367
553368
553369
553370
553371
553372
553373
553374
553375
553376
553377
553378
553379
553380
553381
553382
553383
553384
553385
553386
553387
553388
553389
553390
553391
553392
553393
553394
553395
553396
553397
553398
553399
553400
553401
553402
553403
553404
553405
553406
553407
553408
553409
553410
553411
553412
553413
553414
553415
553416
553417
553418
553419
553420
553421
553422
553423
553424
553425
553426
553427
553428
553429
553430
553431
553432
553433
553434
553435
553436
553437
553438
553439
553440
553441
553442
553443
553444
553445
553446
553447
553448
553449
553450
553451
553452
553453
553454
553455
553456
553457
553458
553459
553460
553461
553462
553463
553464
553465
553466
553467
553468
553469
553470
553471
553472
553473
553474
553475
553476
553477
553478
553479
553480
553481
553482
553483
553484
553485
553486
553487
553488
553489
553490
553491
553492
553493
553494
553495
553496
553497
553498
553499
553500
553501
553502
553503
553504
553505
553506
553507
553508
553509
553510
553511
553512
553513
553514
553515
553516
553517
553518
553519
553520
553521
553522
553523
553524
553525
553526
553527
553528
553529
553530
553531
553532
553533
553534
553535
553536
553537
553538
553539
553540
553541
553542
553543
553544
553545
553546
553547
553548
553549
553550
553551
553552
553553
553554
553555
553556
553557
553558
553559
553560
553561
553562
553563
553564
553565
553566
553567
553568
553569
553570
553571
553572
553573
553574
553575
553576
553577
553578
553579
553580
553581
553582
553583
553584
553585
553586
553587
553588
553589
553590
553591
553592
553593
553594
553595
553596
553597
553598
553599
553600
553601
553602
553603
553604
553605
553606
553607
553608
553609
553610
553611
553612
553613
553614
553615
553616
553617
553618
553619
553620
553621
553622
553623
553624
553625
553626
553627
553628
553629
553630
553631
553632
553633
553634
553635
553636
553637
553638
553639
553640
553641
553642
553643
553644
553645
553646
553647
553648
553649
553650
553651
553652
553653
553654
553655
553656
553657
553658
553659
553660
553661
553662
553663
553664
553665
553666
553667
553668
553669
553670
553671
553672
553673
553674
553675
553676
553677
553678
553679
553680
553681
553682
553683
553684
553685
553686
553687
553688
553689
553690
553691
553692
553693
553694
553695
553696
553697
553698
553699
553700
553701
553702
553703
553704
553705
553706
553707
553708
553709
553710
553711
553712
553713
553714
553715
553716
553717
553718
553719
553720
553721
553722
553723
553724
553725
553726
553727
553728
553729
553730
553731
553732
553733
553734
553735
553736
553737
553738
553739
5537340
5537341
5537342
5537343
5537344
5537345
5537346
5537347
5537348
5537349
5537350
5537351
5537352
5537353
5537354
5537355
5537356
5537357
5537358
5537359
5537360
5537361
5537362
5537363
5537364
5537365
5537366
5537367
5537368
5537369
5537370
5537371
5537372
5537373
5537374
5537375
5537376
5537377
5537378
5537379
5537380
5537381
5537382
5537383
5537384
5537385
5537386
5537387
5537388
5537389
5537390
5537391
5537392
5537393
5537394
5537395
5537396
5537397
5537398
5537399
5537400
5537401
5537402
5537403
5537404
5537405
5537406
5537407
5537408
5537409
5537410
5537411
5537412
5537413
5537414
5537415
5537416
5537417
5537418
5537419
5537420
5537421
5537422
5537423
5537424
5537425
5537426
5537427
5537428
5537429
5537430
5537431
5537432
5537433
5537434
5537435
5537436
5537437
5537438
5537439
5537440
5537441
5537442
5537443
5537444
5537445
5537446
5537447
5537448
5537449
5537450
5537451
5537452
5537453
5537454
5537455
5537456
5537457
5537458
5537459
5537460
5537461
5537462
5537463
5537464
5537465
5537466
5537467
5537468
5537469
5537470
5537471
5537472
5537473
5537474
5537475
5537476
5537477
5537478
5537479
5537480
5537481
5537482
5537483
5537484
5537485
5537486
5537487
5537488
5537489
5537490
5537491
5537492
5537493
5537494
5537495
5537496
5537497
5537498
5537499
5537500
5537501
5537502
5537503
5537504
5537505
5537506
5537507
5537508
5537509
5537510
5537511
5537512
5537513
5537514
5537515
5537516
5537517
5537518
5537519
5537520
5537521
5537522
5537523
5537524
5537525
5537526
5537527
5537528
5537529
5537530
5537531
5537532
5537533
5537534
5537535
5537536
5537537
5537538
5537539
55375340
55375341
55375342

324 presented in Table 4. After fine-tuning Dense2MoE using only 180B tokens, it outperforms Qwen2.5-0.5B by 3.04
 325 percentage points and LLaMA2-7B by 11.16 percentage points, demonstrating that our model remains effective as
 326 model scale increases and can be extended to other open-source model families.
 327

328
329 Table 4: Performance comparison of different methods across various benchmarks

330 LLM	331 Model	332 Activated Parameters	333 Benchmarks								334 Avg.	
			335 CEVAL	336 CMMLU	337 MMLU	338 GSM8K	339 CMath	340 HUMANEVAL	341 MBPP	342 BBH	343 ARC-C	
332 Qwen2.5	Dense	1.5B	335 68.72	336 67.82	337 61.1	338 49	339 63.99	340 35.98	341 46	342 39.73	343 64.42	334 55.19
	Ours	1.2B	335 72.51	336 70.5	337 55.5	338 51.67	339 63.31	340 52.44	341 53.2	342 42.73	343 62.2	334 58.23
332 Llama2	Dense	7B	335 30.99	336 32.75	337 45.8	338 22.83	339 16.6	340 12.8	341 21.6	342 39.36	343 27.8	334 27.84
	Ours	5.7B	335 59.43	336 51.2	337 39.8	338 34.82	339 24.8	340 32.5	341 29.94	342 36.43	343 42.1	334 39

336
337
338 3.4 ABLATION STUDIES

339 **Layer-Fusion Upcycling (LF-UC) outperforms the naive combination of redundant layer pruning and upcycling.** In Table 5, we compare our LF-UC with a baseline approach that prunes layers based on inter-layer similarity as
 340 the redundancy criterion, then simply duplicates and expands the retained layers’ MLPs into MoEs through upcycling.
 341 Experimental results demonstrate that our method achieves superior performance, confirming that Dense2MoE better
 342 preserves LLM’s original capabilities while eliminating redundancy.
 343

344
345 Table 5: Performance comparison of different methods across various benchmarks

346 Method	347 Benchmarks								348 Avg.	
	349 CEVAL	350 CMMLU	351 MMLU	352 GSM8K	353 CMath	354 HUMANEVAL	355 MBPP	356 BBH		
350 Prune+Upcycling	351 53.72	352 54.92	353 41.8	354 39.67	355 38.06	356 36.58	357 37.8	358 26.73	359 27.05	360 39.59
351 Prune+Layer-Fusion Upcycling	352 59.68	353 57.8	354 44.7	355 43.37	356 43	357 36.59	358 41	359 33.79	360 24.57	361 42.72

362
363 4 ANALYSIS364
365 4.1 SCALING ANALYSIS OF THE NUMBER OF PRUNED LAYERS

366 We empirically searched for the impact of the number of pruned layers on model performance, as shown in Figure 5.
 367 We pruned the top- k redundant layers ($k = 1, 3, 5, 7, 9, \dots$) and expanded the parameters via Layer-Fusion Upcycling
 368 (LF-UC). The results demonstrate that Dense2MoE effectively mitigates pruning-induced performance degradation
 369 when the number of pruned layers is within a certain range ($k \leq 8$). However, when the number of pruned layers
 370 exceeds a threshold, the performance of Dense2MoE degrades rapidly, as the negative impact of layer-wise pruning
 371 becomes the dominant factor.

372
373 4.2 SCALING ANALYSIS OF THE NUMBER OF EXPERTS

374 We scaled the total number of experts per layer. Under the top-1 gating setting, model performance improves sharply
 375 as the number of experts increases, but the improvement slows when the number of experts reaches a certain point,
 376 limited by the scale of training data and active parameters. Thus, for Dense2MoE at the Qwen2.5-0.5B scale, we chose
 377 a configuration with a total of $N=6$ experts and pruning the top-5 redundant layers.

378
379 5 RELATED WORK

380 The Mixture of Experts (MoE) framework has emerged as a dominant paradigm for scaling large language models
 381 (LLMs) while balancing computational efficiency, addressing the limitations of dense models where parameter growth
 382 linearly increases inference costs. By decomposing the model into specialized “expert” sub-modules and a routing
 383 mechanism that activates only a subset of experts per token, MoE achieves parameter sparsity without sacrificing
 384 capacity.

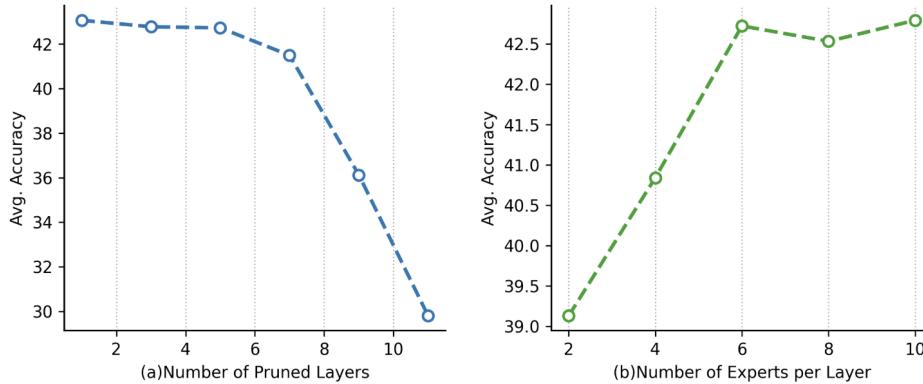


Figure 5: Scaling Analysis of Layers and Experts

However, the computational cost of training MoE models from scratch remains prohibitive, prompting research into efficiently leveraging open-source models for resource reuse. Model pruning is a straightforward approach that calculates similarity between weights, considers dimensionality reduction, and evaluates parameter activation frequency to eliminate redundancy. Unstructured pruning maintains performance while sparsifying the model but creates irregular parameter patterns that often require operator-level customization and optimization. In contrast, structured pruning ensures parameter regularity and better compatibility but risks significant performance degradation by removing entire neurons or layers, as it deletes larger, potentially more critical components.

Upcycling methods extend dense model parameters by reusing attention mechanisms and replicating MLPs, cost-effectively improving performance under equivalent activated parameter constraints. However, this approach increases the redundancy already present in LLMs.

Our Dense2MoE integrates layer-wise pruning with upcycling through our proposed Layer-Fusion Upcycling (LF-UC) technique, removing redundant LLM layers while preserving their more critical MLP components. This approach enables Dense2MoE to achieve a superior balance between performance and efficiency compared to standalone pruning or upcycling methods.

6 CONCLUSION

In this work, we propose Dense2MoE, a novel framework that jointly combines layer-wise pruning and parameter upcycling to enhance both efficiency and performance of large language models. By leveraging the Layer-Fusion Upcycling (LF-UC) technique, Dense2MoE identifies and removes redundant layers while repurposing valuable MLP parameters as additional experts in an MoE structure, achieving a balance between parameter efficiency and model capacity. Comprehensive evaluations across multiple domains, including mathematics, code, and general knowledge, demonstrate that Dense2MoE consistently outperforms approaches relying solely on pruning or parameter expansion. Moreover, the method generalizes effectively across different model families and scales, enabling practical deployment on diverse hardware with minimal training costs.

Dense2MoE highlights a synergistic strategy for optimizing open-source models, paving the way for further exploration of expert activation, attention upcycling, and integration with other compression techniques to maximize efficiency without compromising performance.

432 REFERENCES

433

434 Saleh Ashkboos, Maximilian L Croci, Marcelo Gennari do Nascimento, Torsten Hoefer, and James Hensman.
435 Slicecpt: Compress large language models by deleting rows and columns. *arXiv preprint arXiv:2401.15024*, 2024.

436 Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang,
437 Carrie Cai, Michael Terry, Quoc Le, et al. Program synthesis with large language models. *arXiv preprint
438 arXiv:2108.07732*, 2021.

439

440 Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Ed-
441 wards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code.
442 *arXiv preprint arXiv:2107.03374*, 2021.

443 Xiaodong Chen, Yuxuan Hu, Jing Zhang, Yanling Wang, Cuiping Li, and Hong Chen. Streamlining redundant layers
444 to compress large language models. *arXiv preprint arXiv:2403.19135*, 2024.

445

446 Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord.
447 Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*,
448 2018.

449 Andrey Gromov, Kushal Tirumala, Hassan Shapourian, Paolo Glorioso, and Daniel A Roberts. The unreasonable
450 ineffectiveness of the deeper layers. *arXiv preprint arXiv:2403.17887*, 2024.

451

452 Shuai He, Guoheng Sun, Zheyu Shen, and Ang Li. What matters in transformers? not all attention is needed. *arXiv
453 preprint arXiv:2406.15786*, 2024.

454

455 Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Mea-
456 suring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

457

458 Yuzhen Huang, Yuzhuo Bai, Zhihao Zhu, Junlei Zhang, Jinghan Zhang, Tangjun Su, Junteng Liu, Chuancheng Lv,
459 Yikai Zhang, Yao Fu, et al. C-eval: A multi-level multi-discipline chinese evaluation suite for foundation models.
Advances in Neural Information Processing Systems, 36:62991–63010, 2023.

460

461 Gueyoung Jung, Matti A Hiltunen, Kaustubh R Joshi, Richard D Schlichting, and Calton Pu. Mistral: Dynamically
462 managing power, performance, and adaptation cost in cloud infrastructures. In *2010 IEEE 30th International Con-
463 ference on Distributed Computing Systems*, pp. 62–73. IEEE, 2010.

464

465 Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Rad-
466 ford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*,
467 2020.

468

469 Bo-Kyeong Kim, Geonmin Kim, Tae-Ho Kim, Thibault Castells, Shinkook Choi, Junho Shin, and Hyoung-Kyu Song.
470 Shortened llama: A simple depth pruning for large language models. *arXiv preprint arXiv:2402.02834*, 11:1, 2024.

471

472 Aran Komatsuzaki, Joan Puigcerver, James Lee-Thorp, Carlos Riquelme Ruiz, Basil Mustafa, Joshua Ainslie, Yi Tay,
473 Mostafa Dehghani, and Neil Houlsby. Sparse upcycling: Training mixture-of-experts from dense checkpoints. *arXiv
474 preprint arXiv:2212.05055*, 2022.

475

476 Haonan Li, Yixuan Zhang, Fajri Koto, Yifei Yang, Hai Zhao, Yeyun Gong, Nan Duan, and Timothy Baldwin. Cmmlu:
477 Measuring massive multitask language understanding in chinese. *arXiv preprint arXiv:2306.09212*, 2023.

478

479 Bingbin Liu, Sebastien Bubeck, Ronen Eldan, Janardhan Kulkarni, Yuanzhi Li, Anh Nguyen, Rachel Ward, and
480 Yi Zhang. Tinygsm: achieving 80% on gsm8k with small language models. *arXiv preprint arXiv:2312.09241*,
481 2023.

482

483 Xin Men, Mingyu Xu, Qingyu Zhang, Bingning Wang, Hongyu Lin, Yaojie Lu, Xianpei Han, and Weipeng Chen.
484 Shortgpt: Layers in large language models are more redundant than you expect. *arXiv preprint arXiv:2403.03853*,
485 2024.

486

487 N Shazeer, A Mirhoseini, K Maziarz, A Davis, Q Le, G Hinton, and J Dean. The sparsely-gated mixture-of-experts
488 layer. *Outrageously large neural networks*, 2, 2017.

489

490 Jiwon Song, Kyungseok Oh, Taesu Kim, Hyungjun Kim, Yulhwa Kim, and Jae-Joon Kim. Sleb: Streamlining llms
491 through redundancy verification and elimination of transformer blocks. *arXiv preprint arXiv:2402.09025*, 2024.

486 Sainbayar Sukhbaatar, Olga Golovneva, Vasu Sharma, Hu Xu, Xi Victoria Lin, Baptiste Rozière, Jacob Kahn, Daniel
 487 Li, Wen-tau Yih, Jason Weston, et al. Branch-train-mix: Mixing expert llms into a mixture-of-experts llm. *arXiv*
 488 *preprint arXiv:2403.07816*, 2024.

489 Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha
 490 Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought
 491 can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

492 Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

493 Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov,
 494 Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models.
 495 *arXiv preprint arXiv:2307.09288*, 2023.

496 Kulkarni Tripti R, P Chethan, Singh Arun Vikas, and S Monisha S. Deepseek open-source ai. *International Journal*
 497 *of Trend in Scientific Research and Development*, 9(3):555–560, 2025.

498 Tianwen Wei, Jian Luan, Wei Liu, Shuang Dong, and Bin Wang. Cmath: Can your language model pass chinese
 499 elementary school math test? *arXiv preprint arXiv:2306.16636*, 2023.

500 Chengyue Wu, Yukang Gan, Yixiao Ge, Zeyu Lu, Jiahao Wang, Ye Feng, Ying Shan, and Ping Luo. Llama pro:
 501 Progressive llama with block expansion. *arXiv preprint arXiv:2401.02415*, 2024.

502 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengan
 503 Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

504 Yifei Yang, Zouying Cao, and Hai Zhao. Laco: Large language model pruning via layer collapse. *arXiv preprint*
 505 *arXiv:2402.11187*, 2024.

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