# *Learning to Prune and Low-Rank Adaptation for Compact Language Model Deployment*

Authors: **Asmer Hamid Ali (aali115@asu.edu)**, Fan Zhang, Li Yang, Deliang Fan
Efficient, Secure and Intelligent Computing (ESIC) Laboratory
(https://faculty.engineering.asu.edu/dfan/)
Arizona State University

# Outline

1. **Motivation and Problem Statement**
   - Challenges in deploying large pre-trained models.
   - Limitations of existing methods.
2. **Key Contributions**
   - Overview of proposed approach and its significance.
3. **Parameter-Efficient Fine-Tuning and Model Pruning**
   - Background on PEFT techniques.
   - Importance of structured pruning for efficiency.
4. **Methodology Overview**
   - Trainable pruning masks.
   - Integration with low-rank adaptation.
5. **Efficient Pruning and Low-Rank Adaptation**
   - Detailed explanation with equations and benefits.
6. **Experimental Setup**
   - Models, datasets, and evaluation metrics.
7. **Results**
   - Performance analysis and comparison with baselines.
8. **Conclusion**
   - Summary of contributions and future directions.

# Motivation and Problem Statement

- Growing computational demands of large pre-trained models (LPMs).
- PEFT techniques address training overhead but fail to optimize inference efficiency.

- Need for a **compact and efficient** deployment-ready solution.

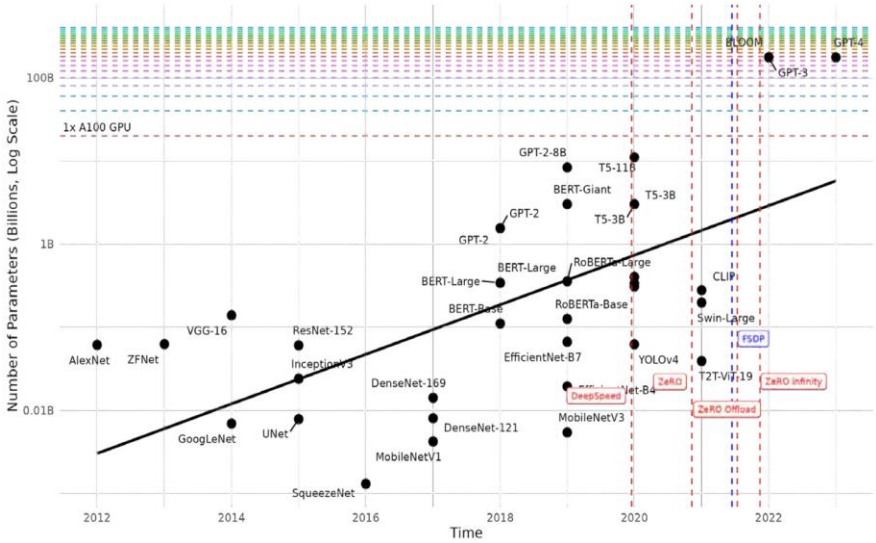


**Figure 1**: Chart showing the growth in the size of models over time with annotations on memory usage and limits of hardware *(Source: LLM: The Rise of Data)*

# Motivation and Problem Statement

- Growing computational demands of large pre-trained models (LPMs).
- PEFT techniques address training overhead but fail to optimize inference efficiency.

- Need for a **compact and efficient** deployment-ready solution.

| Model | PEFT | Params (%) | Avg. Accuracy |
|---|---|---|---|
| ChatGPT | — | — | 77.0% |
| LLaMA-7B | | | |
| | PrefT | 0.110% | 64.6% |
| | AdapterS | 0.990% | 70.8% |
| | AdapterP | 3.540% | 72.3% |
| | LoRA | 0.830% | 74.7% |
| | DoRA (half) | 0.430% | 77.5% |
| | DoRA | 0.840% | 78.1% |
| | LoReFT | 0.031% | 80.2% |
| LLaMA-13B | | | |
| | PrefT | 0.030% | 68.4% |
| | AdapterS | 0.800% | 79.5% |
| | AdapterP | 2.890% | 81.5% |
| | LoRA | 0.670% | 80.5% |
| | DoRA (half) | 0.350% | 80.8% |
| | DoRA | 0.680% | 81.5% |
| | LoReFT | 0.025% | 83.3% |

**Figure 2**: Table comparing LLaMA-7B models with various PEFT methods, showing parameter reductions and accuracy trade-offs. *(Source: Charith Chandra Sai Balne et al., "Parameter Efficient Fine Tuning: A Comprehensive Analysis Across Applications,"arXiv:2404.13506, 2024)*

# Key Contribution

1. **Trainable Pruning Methodology**
   - Optimizes the structure of LPMs during fine-tuning.
   - Includes learnable binary masks for channel-wise pruning.
2. **Low-Rank Adaptation Integration**
   - Incorporates low-rank adaptation to reduce computational overhead while maintaining accuracy.
3. **Efficiency Gains**
   - Demonstrates up to **18% speed-up** in inference with real-world hardware.

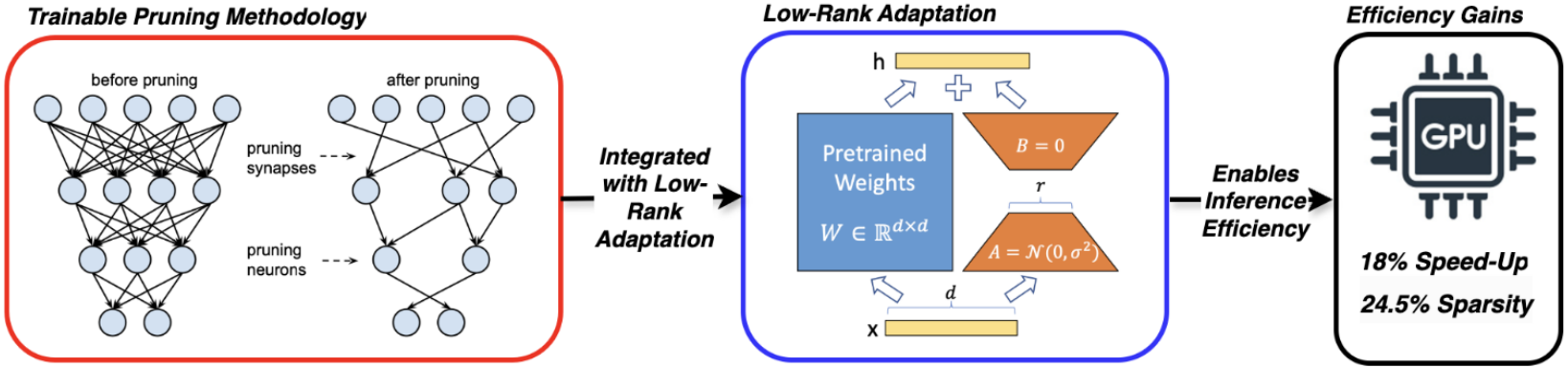


**Figure 3**: Proposed Approach

# Parameter-Efficient Fine-Tuning and Model Pruning

# Parameter-Efficient Fine-Tuning and Model Pruning

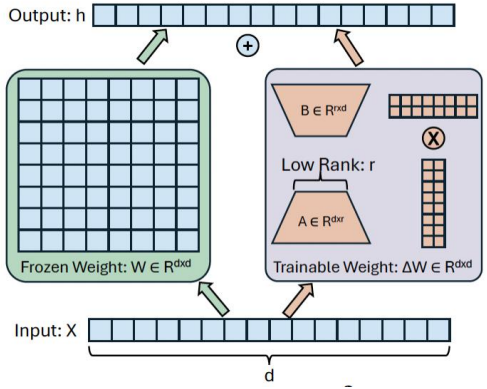


**Figure 4**: Structure of LoRA *(Source: E. J. Hu et al. Lora: Low-rank adaptation of large language models, 2021)*

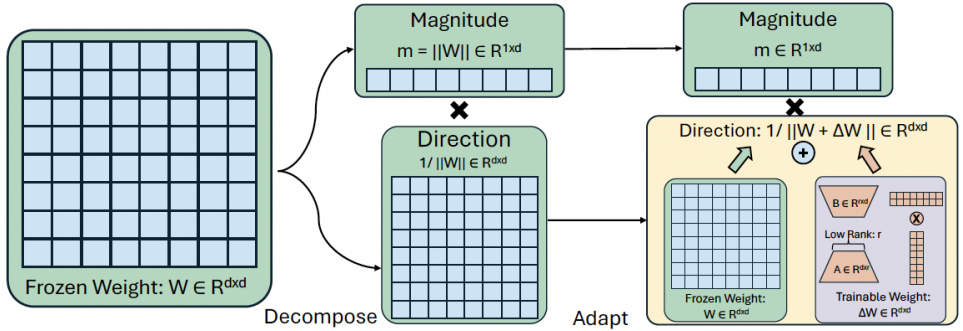


**Figure 5**: Structure of DoRA (Source: *S.-Y. Liu et al. Dora: Weight-decomposed low-rank adaptation, 2024.*

# Parameter-Efficient Fine-Tuning and Model Pruning

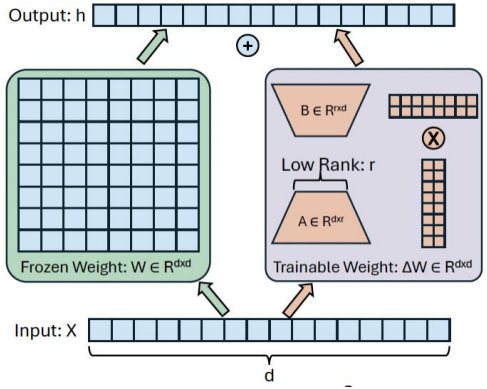


**Figure 4**: Structure of LoRA *(Source: E. J. Hu et al. Lora: Low-rank adaptation of large language models, 2021)*

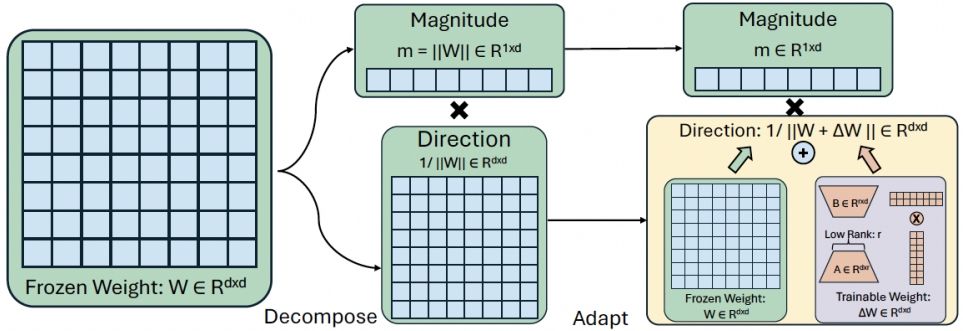


**Figure 5**: Structure of DoRA (Source: *S.-Y. Liu et al. Dora: Weight-decomposed low-rank adaptation, 2024.*

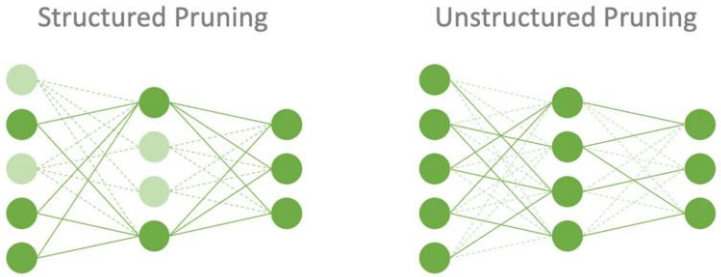


**Figure 6**: Pruning Techniques

# Methodology Overview

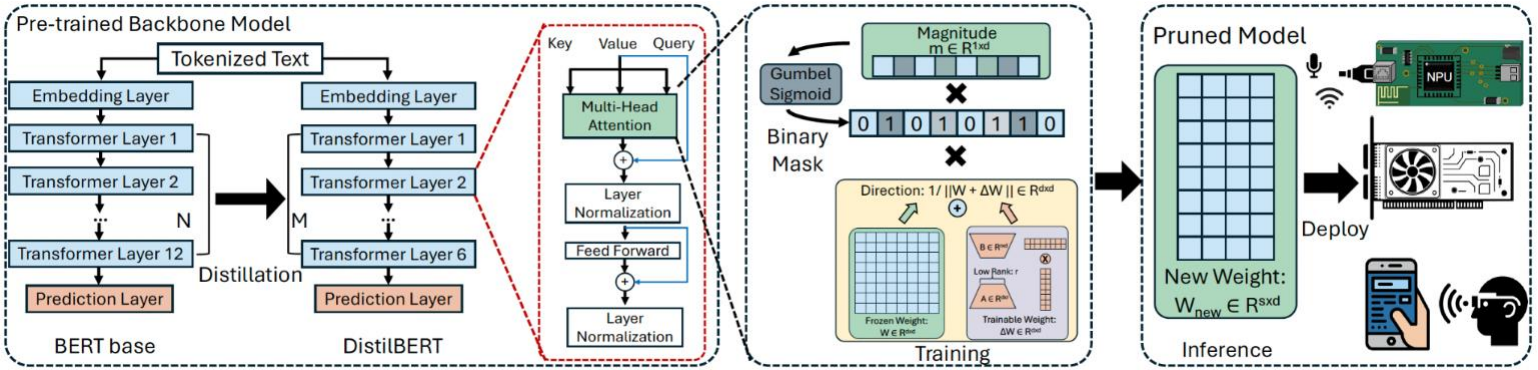


**Figure 5**: Overview of the proposed approach

# Methodology Overview

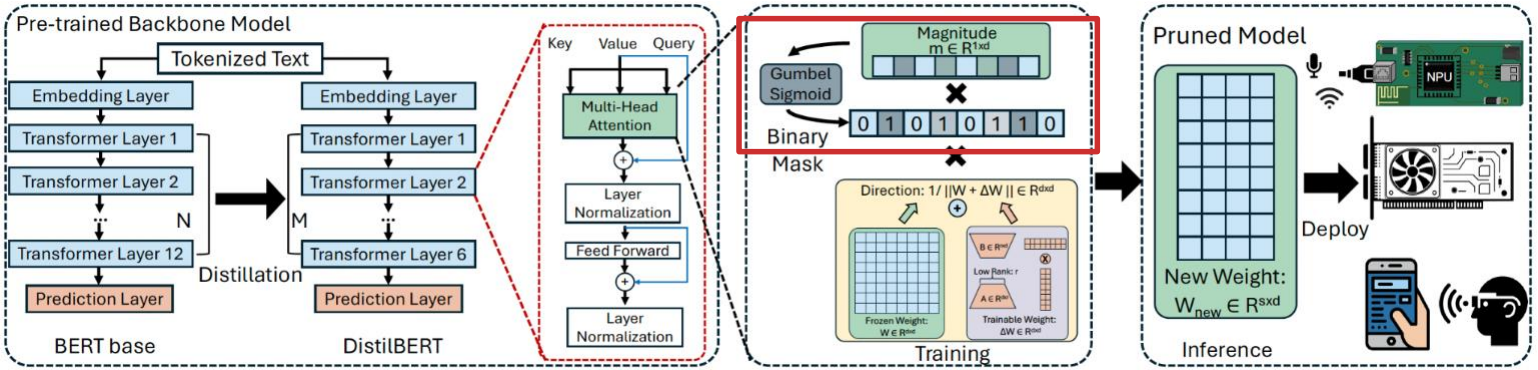


**Figure 5**: Overview of the proposed approach

1. **Trainable Pruning Masks:** Introduce binary masks to prune unimportant weights in both frozen and trainable components.

# Methodology Overview

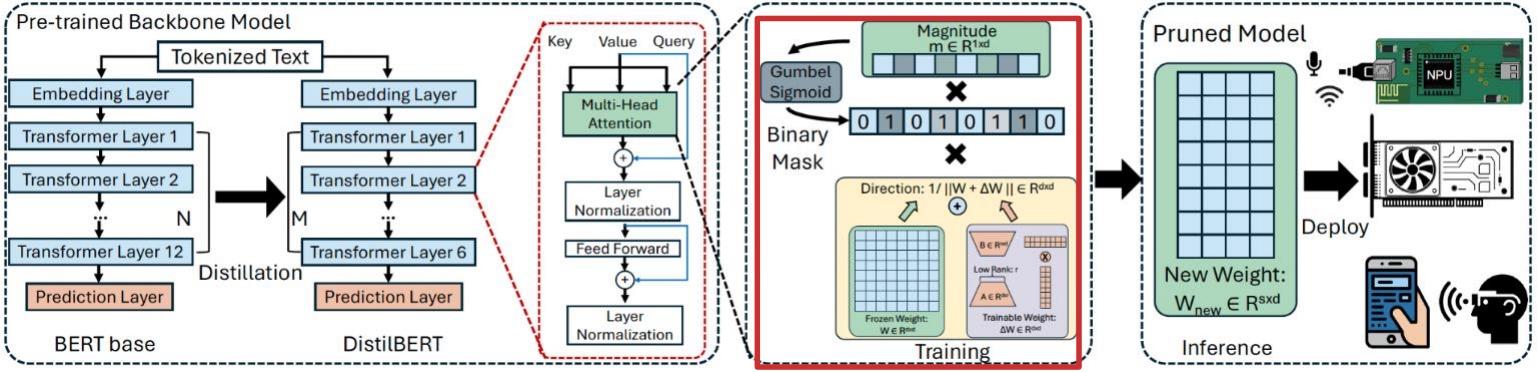


**Figure 5**: Overview of the proposed approach

1. **Trainable Pruning Masks:** Introduce binary masks to prune unimportant weights in both frozen and trainable components.
2. **Integration with Low-Rank Adaptation**: Decompose weights into magnitude and direction using low-rank adaptation (based on DoRA). Optimize the pruning process by focusing only on magnitude vectors, minimizing training overhead.

# Methodology Overview

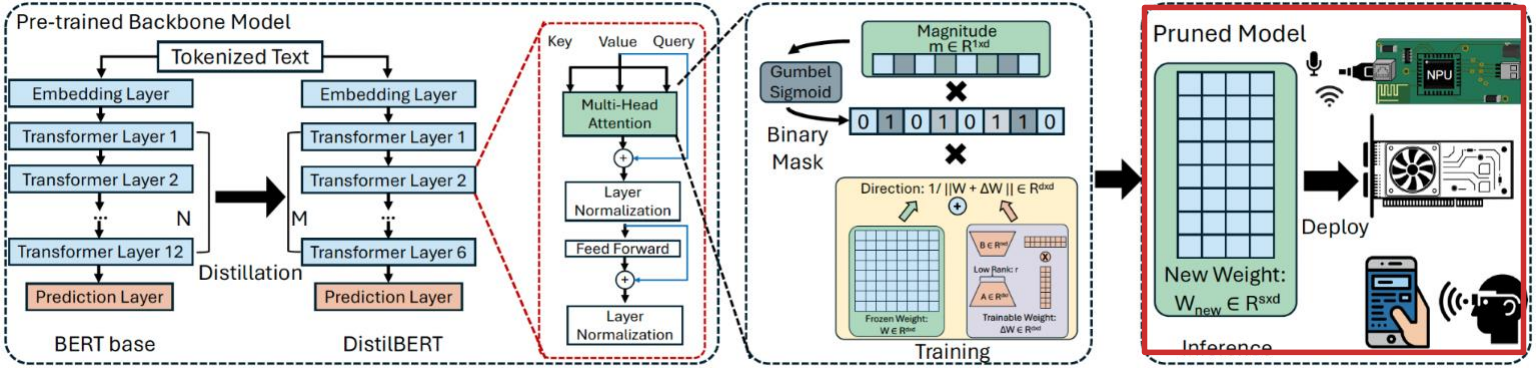


**Figure 5**: Overview of the proposed approach

1. **Trainable Pruning Masks:** Introduce binary masks to prune unimportant weights in both frozen and trainable components.
2. **Integration with Low-Rank Adaptation**: Decompose weights into magnitude and direction using low-rank adaptation (based on DoRA). Optimize the pruning process by focusing only on magnitude vectors, minimizing training overhead.
3. **Hardware-Compatible Compact Model**: The final pruned model retains its compact structure. Achieves significant inference speed-up on commercial GPUs and CPUs.

# Efficient Pruning and Low-Rank Adaptation

# Efficient Pruning and Low-Rank Adaptation

1. **Integration of Pruning with Low-Rank Adaptation**:
   - Use a trainable binary mask ($m_b$) to optimize the magnitude vector in the DoRA framework.
   - Low-rank adaptation ensures computational efficiency while maintaining accuracy.
2. **Low-Rank Adaptation**:

$$W' = m\frac{V + \Delta V}{||V + \Delta V||} = m\frac{V + BA}{||V + BA||}$$

# Efficient Pruning and Low-Rank Adaptation

1. **Integration of Pruning with Low-Rank Adaptation**:
    - Use a trainable binary mask ($m_b$) to optimize the magnitude vector in the DoRA framework.
    - Low-rank adaptation ensures computational efficiency while maintaining accuracy.
2. **Low-Rank Adaptation**:

$$W' = m\frac{V + \Delta V}{||V + \Delta V||} = m\frac{V + BA}{||V + BA||}$$

- **Pruned Weight Update**:

$$W'_p = m_b W' = m_b(m\frac{V + \Delta V}{||V + \Delta V||}) = (m_b \odot m)\frac{V + \Delta V}{||V + \Delta V||}$$

# Experimental Setup

1. **Model Used:** DistilBERT, RoBERTa ($RoB_{base}$), and LLaMA-7B
2. **Dataset:**
    a. GLUE Benchmark for DistilBERT and RoBERTa.
    b. Commonsense reasoning datasets (e.g., BoolQ, PIQA, ARC) for LLaMA-7B.
3. **Hardware and Training Details**:
    a. GPU: NVIDIA A5000
    b. Batch size: 8, mixed precision for efficiency.
    c. Optimizer: Adam, learning rate fine-tuned during stages (e.g., 5e-5 → 1e-5).

# Results for LLaMA-7B

Table 1: Performance comparison of different LLaMA-7B models on various datasets.

| Model | # TP | BoolQ | PIQA | HellaSwag | WinoGrande | ARC-e | ARC-c | OBQA | Avg. |
|---|---|---|---|---|---|---|---|---|---|
| LLaMA-7B (Baseline) | 6.74B | 76.5 | 79.8 | 76.1 | 70.1 | 72.8 | 47.6 | 57.2 | 68.59 |
| LLaMA-7B (LLM-Pruner) | 5.42B | 69.54 | 76.44 | 68.11 | 65.11 | 63.43 | 37.88 | 40 | 60.07 |
| LLaMA-7B (LoRAPrune) | - | 65.62 | 79.31 | 70 | 62.76 | 65.87 | 37.69 | 39.14 | 60.05 |
| LLaMA-7B (LoRAShear) | - | 72.78 | 76.36 | 69.49 | 67.63 | 69.02 | 39.47 | 40.78 | 62.22 |
| Pruned-LLaMA-7B (Ours) | 5.09B | 72.12 | 79.54 | 71.65 | 67.93 | 67.98 | 38.84 | 41.32 | 62.77 |

1. **Accuracy Gains**: *Pruned-LLaMA-7B (Ours)* achieves a competitive **average accuracy (62.77%)**, outperforming most models like LLM-Pruner and LoRAPrune.
2. **Model Size Efficiency**: Reduces the number of trainable parameters to **5.09B**, compared to the baseline's **6.74B**, while maintaining comparable performance.
3. **Task-Specific Highlights**:
    a. Excels in **PIQA (79.54%)**, outperforming all baselines.
    b. Strong performance on **BoolQ (72.12%)** and **WinoGrande (67.93%)** compared to LLM-Pruner and LoRAPrune.

# Results for $RoB_{base}$

Table 2: Performance comparison of different RoBERTa models on various datasets.

| Model | # TP | Model Size (MB) | MRPC (Acc) | RTE (Acc) | SST-2 (Acc) | MNLI (Acc) | CoLA (Mat.) | QNLI (Acc) | QQP (Acc) | STS-B (Pearson) | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $RoB_{base}$ (Baseline) | 125M | 476.84 | 90.2 | 78.7 | 94.8 | 87.6 | 63.3 | 92.8 | 91.9 | 91.2 | 86.4 |
| $RoB_{base}$ (LoRA) | 0.3M | 478.1 | 89.7 | 86.6 | 95.1 | 87.5 | 63.4 | 93.3 | 90.8 | 91.5 | 87.2 |
| $RoB_{base}$ (DyLoRA) | 0.887M | - | 91.38 | 77.55 | 94.36 | 86.76 | 59.51 | 93 | 89.91 | 91.05 | 85.44 |
| $RoB_{base}$ (LoRA-drop) | 0.15M | - | 89.5 | 81.4 | 94.5 | 87.3 | 62.9 | 93.1 | 90.1 | 91 | 86.2 |
| Pruned-$RoB_{base}$ (Ours) | 0.331M | 429.3** | 91.5 | 83.8 | 95.3 | 88 | 64.2 | 93.28 | 91 | 92.62 | 87.46 |

** Refers to the average model size. # TP refers to number of trainable parameters.

1. **Accuracy Gains**: *Pruned-$RoB_{base}$ (Ours) achieves the highest **average accuracy (87.46%)**, outperforming the baseline and LoRA methods.*
2. **Model Size Efficiency**: Pruned-$RoB_{base}$ reduces model size to **429.3 MB**, significantly smaller than the baseline $RoB_{base}$ (476.84 MB).
3. **Task-Specific Highlights**: Achieves the best accuracy on **SST-2 (95.3%)** and **RTE (83.8%)** while maintaining competitive performance on other tasks.

# Results for DistilBERT

Table 3: Performance comparison of different DistilBERT models on various datasets.

| Model | # TP* | Model Size (MB) | MRPC (Acc) | RTE (Acc) | SST-2 (Acc) | MNLI (Acc) | CoLA (Matthew) | QNLI (Acc) | QQP (Acc) | STS-B (Pea.) | Avg. |
|---|---|---|---|---|---|---|---|---|---|---|---|
| DistilBERT* (Baseline) | 66M | 255.75 | 87.5 | 59.9 | 91.3 | 82.2 | 51.3 | 89.2 | 88.5 | 86.9 | 79.6 |
| DistilBERT-LoRA | 0.147M | 258.32 | 87.2 | 63.7 | 91.8 | 82.9 | 55.4 | 90.1 | 89.2 | 87.3 | 81 |
| DistilBERT-DoRA | 0.156M | 268.5 | 89.6 | 79.6 | 93.2 | 83.9 | 61.2 | 92.4 | 91.2 | 89.4 | 85.1 |
| Uniformly Pruned DistilBERT-DoRA | 0.156M | 209.6 | 84.7 | 58.1 | 89.6 | 80.8 | 50 | 83.1 | 84.2 | 85 | 77 |
| Pruned-DistilBERT (Ours) | 0.165M | 222.85** | 88.6 | 65.2 | 91.4 | 83.4 | 57 | 91.3 | 90.7 | 88 | 82 |

* Refers to the results directly from their original paper[15]. # TP refers to number of trainable parameters. ** Refers to the average model size.

1. **Accuracy Gains**: *Pruned-DistilBERT (Ours)* achieves the highest **average accuracy (82%)**, outperforming other methods. Consistently performs better across tasks like **RTE (65.2%)**, **SST-2 (91.4%)**, and **MNLI (83.4%)**.
2. **Model Size Efficiency**: *Pruned-DistilBERT* achieves an efficient model size of **222.85 MB**, smaller than most other methods like LoRA (258.32 MB) and DoRA (268.5 MB).
3. **Improved Performance vs. Baseline**: Outperforms DistilBERT Baseline by **6.4%** in average accuracy (82% vs. 79.6%).

# Analysis of Sparsity and Inference Gains

Table 4: Inference model size and time reduction with our pruning method on different models and datasets.

| Dataset | Pruned-DistilBERT-DoRA Reduction (Ours) | | Pruned-RoB$_{base}$-DoRA Reduction (Ours) | |
|---|---|---|---|---|
| | Model Size | Inference Time | Model Size | Inference Time |
| MRPC | 18% | 15% | 25% | 18.5% |
| RTE | 21% | 16.2% | 28% | 20.3% |
| SST-2 | 21% | 14.2% | 27% | 18.7% |
| MNLI | 17% | 11.9% | 23% | 17.8% |
| CoLA | 15% | 11.6% | 24% | 18.1% |
| QNLI | 13% | 9.6% | 22% | 16.6% |
| QQP | 15% | 10.9% | 23% | 17.2% |
| STS-B | 16% | 15% | 24% | 17.4% |

Table 5: Model size and inference time reduction of LLaMa model with our pruning method on various datasets.

| Dataset | Model Size | Inference Time |
|---|---|---|
| BoolQ | 22% | 16.6% |
| PIQA | 27% | 20% |
| HellaSwag | 25% | 18.3% |
| WinoGrande | 24% | 17.6% |
| ARC-e | 23% | 17% |
| ARC-c | 24% | 18.2% |
| OBQA | 26% | 19.6% |

# Conclusion

**Summary of Contributions**:

- Introduced a novel trainable pruning methodology for structured pruning.
- Integrated pruning with low-rank adaptation to reduce computational costs while maintaining accuracy.

**Key Results**:

- Up to **24.5% sparsity** across layers.
- Up to **18% inference speed-up** on real-world hardware.

**Broader Impact**:

- Enables practical deployment of large pre-trained models in resource-constrained environments.
- Balances performance, efficiency, and deployment feasibility.

# References


- *E. J. Hu et al. Lora: Low-rank adaptation of large language models, 2021.*
- S.-Y. Liu et al. Dora: Weight-decomposed low-rank adaptation, 2024.
- V. Sanh et al. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.
- Y. Liu et al. Roberta: A robustly optimized bert pretraining approach, 2019.
- H. Touvron et al. Llama: Open and efficient foundation language models, 2023.
- X. Ma et al. Llm-pruner: On the structural pruning of large language models, 2023.
- M. Zhang et al. Loraprune: Pruning meets low-rank parameter-efficient finetuning, 2023
- M. Xia et al. Sheared llama: Accelerating language model pre-training via structured pruning, 2024
- E. Jang et al. Categorical reparameterization with gumbel-softmax, 2017.
- T. Chen et al. Lorashear: Efficient large language model structured pruning and knowledge recovery, 2023.
- H. Zhou et al. Lora-drop: Efficient lora parameter pruning based on output evaluation, 2024.
- M. Valipour et al. Dylora: Parameter efficient tuning of pre-trained models using dynamic search-free low-rank adaptation, 2023.

# Thank You!

# Any Questions?


## *Acknowledgement*

*This work is supported in part by the National Science Foundation under Grant No.2314591, No.2505326, No.2452573, No.2452657, No.2503906, and No.2505209.*



*Contact information:*

*Asmer Hamid Ali: aali115@asu.edu*

*Deliang Fan: dfan@asu.edu*