CURLoRA: Leveraging CUR Matrix Decomposition for Stable LLM Continual Fine-tuning and Catastrophic Forgetting Mitigation

Muhammad Fawi

Abstract

This paper introduces CURLoRA, a novel approach to fine-tuning large language models (LLMs) that leverages CUR matrix decomposition in the context of Low-Rank Adaptation (LoRA). Our method addresses two critical challenges in LLM fine-tuning: mitigating catastrophic forgetting during continual learning and reducing the number of trainable parameters. We propose a unique modification to the CUR decomposition process, utilizing inverted probabilities for column and row selection, which acts as an implicit regularization. Through experiments on multiple datasets, we demonstrate that CURLoRA outperforms standard LoRA in maintaining model stability and performance across tasks while significantly reducing the number of trainable parameters. Our results show that CURLoRA achieves superior accuracy and perplexity scores compared to LoRA, particularly in scenarios with limited fine-tuning data.

1 Introduction

Large Language Models (LLMs) have revolutionized natural language processing, demonstrating remarkable capabilities across a wide range of tasks [1]. However, fine-tuning these models for specific tasks often leads to catastrophic forgetting, where the model loses its ability to perform well on previously learned tasks [2]. Additionally, the computational resources required for fine-tuning LLMs are substantial, making it challenging to adapt these models efficiently, especially when working with limited datasets [3].

Low-Rank Adaptation (LoRA) [4] has emerged as an efficient fine-tuning method, decomposing pre-trained weight matrices into low-rank matrices and fine-tune those ones instead of the original matrix. Although LoRA has proven to be very promising in enabling large language model fine-tuning consuming much less computational resources, it still faces challenges in preventing catastrophic forgetting. Catastrophic forgetting occurs due to the overwriting of previously learned (pre-trained) weights during the fine-tuning process. In LoRA, this often happens as the adapted output can significantly deviate from the original:

$$y = xW + xW_{adapted} = x(W + AB) \tag{1}$$

where W is the original weight matrix, and AB is the low-rank update.

This work introduces CURLoRA, a novel approach that applies low-rank approximation (LoRA) to pre-trained weight matrices using CUR matrix decomposition [5] instead of SVD or random initiation of the low-rank A & B matrices. We propose a unique modification to the CUR decomposition process and demonstrate its effectiveness in mitigating catastrophic forgetting while also reducing the number of trainable parameters.

2 Related Work

2.1 Catastrophic Forgetting

Catastrophic forgetting is a big challenge in machine learning, particularly in the context of continual learning [2]. Various approaches have been proposed to address this issue:

- Elastic Weight Consolidation (EWC) [6] uses Fisher information to measure the importance of parameters and selectively slow down learning on important parameters.
- Progressive Neural Networks [7] propose to freeze the network trained on previous tasks and add lateral connections to new columns for new tasks.
- Memory-based approaches like Experience Replay [8] store and replay examples from previous tasks during training on new tasks.

2.2 Efficient Fine-tuning of Large Language Models

As LLMs have grown in size, efficient fine-tuning methods have become crucial:

- Adapter layers [9] introduce small trainable modules between layers of a pre-trained model.
- Low-Rank Adaptation (LoRA) [4] decomposes weight updates into low-rank matrices, significantly reducing the number of trainable parameters.
- Prefix-tuning [10] prepends trainable continuous prompts to the input, allowing for task-specific adaptations.

2.3 CUR Matrix Decomposition

CUR decomposition has been applied in various domains for its interpretability and efficiency:

- In data analysis, CUR has been used for feature selection and dimensionality reduction [5].
- In scientific computing, CUR has been applied to accelerate large-scale matrix computations [11].
- In machine learning, CUR has been explored for model compression and interpretation [12].

However, to the best of our knowledge, CUR decomposition has not been previously applied to the problem of fine-tuning large language models or addressing catastrophic forgetting in this context.

3 Background on CUR Decomposition

CUR decomposition is a matrix factorization technique that approximates a matrix A as the product of three matrices: C, U, and R. Unlike Singular Value Decomposition (SVD), CUR decomposition uses actual columns and rows of the original matrix, providing better interpretability [5].

Given a matrix $A \in \mathbb{R}^{m \times n}$, CUR decomposition approximates A as:

$$A \approx CUR \tag{2}$$

where:

- $C \in \mathbb{R}^{m \times c}$ consists of c columns of A
- $R \in \mathbb{R}^{r \times n}$ consists of r rows of A
- $U \in \mathbb{R}^{c \times r}$ is a small matrix that ensures CUR is close to A

The columns and rows are typically chosen based on their statistical leverage scores, which measure their importance in the matrix [11]

4 This Work

In this section, we present CURLoRA, our novel approach to fine-tuning large language models that leverages a modified CUR matrix decomposition to mitigate catastrophic forgetting. We provide a detailed mathematical formulation of the approach, analyze it theoretically, and explain how it addresses the challenge of catastrophic forgetting upon continual learning.

4.1 CURLoRA

The core idea is to decompose the pre-trained weight matrices using a modified CUR approach and then fine-tune only the U matrix. This approach constrains the parameter space of possible adaptations keeping the fine-tuned parameters as small as possible to keep $\|W_{\text{adapted}} - W\|$ close to the original weight matrix to avoid the deviation of the adapted output.

4.2 Mathematical Formulation

Given a weight matrix W, we first compute the column probabilities:

$$p_j = \frac{\|W_{:j}\|_2^2}{\|W\|_F^2} \tag{3}$$

where $W_{:j}$ is the j-th column of W and $\|\cdot\|_F$ denotes the Frobenius norm. We then invert these probabilities:

$$\tilde{p}_j = \frac{1/p_j}{\sum_k 1/p_k} \tag{4}$$

Then, we sample columns and rows according to these inverted probabilities to construct C and R, which will always be fixed, with columns and rows with lower original probabilities. This trick plays a major role in the approach as it serves two purposes:

- It acts as a form of regularization, preventing the model from overfitting and limiting the adaptation of the U matrix stopping it from growing so big.
- It preserves the model's original behavior by focusing adaptations on less influential parts of the weight matrix.

The U matrix is initialized as a zero matrix:

$$U_{\text{init}} = 0 \tag{5}$$

During fine-tuning, we update only the U matrix, keeping C and R fixed:

$$W_{\text{adapted}} = CUR \tag{6}$$

4.3 Theoretical Analysis of Catastrophic Forgetting Mitigation

To understand how CURLoRA helps mitigate catastrophic forgetting, we analyze its properties mathematically:

4.3.1 Parameter Space Constraint

In CURLoRA, we decompose the original weight matrix W as:

$$W \approx CUR \tag{7}$$

During fine-tuning, we're optimizing:

$$W_{\text{adapted}} = C(U + \Delta U)R \tag{8}$$

where ΔU represents the changes made to U during fine-tuning. This constrains and limits the adaptation to the subspace spanned by C and R, meaning the modifications are limited, preventing drastic changes that might lead to forgetting previous knowledge.

4.3.2 Implicit Regularization

By initializing U as a zero matrix, and C and R with columns and rows of low weight values, the ones with lower probabilities, we provide an implicit regularization. This can be seen as adding a regularization term to the loss function:

$$L_{\text{CURLoRA}}(\theta) = L_{\text{task}}(\theta) + \|U\|_F^2 \tag{9}$$

where $\|U\|_F$ is the Frobenius norm of the U matrix that is being fine-tuned. This implicit regularization term encourages the model to keep the changes small i.e. $\|W_{\text{adapted}} - W\|$ is close to the original weights W. For instance, if U is initially zero, this term will push the fine-tuning process to make only necessary adjustments, preventing overfitting and excessive reliance on the fine-tuned parameters.

4.3.3 Reduced Interference

Considering the gradients of the loss with respect to the parameters, in CURLoRA we have:

$$\frac{\partial L}{\partial W} = C^T \left(\frac{\partial L}{\partial U} \right) R^T \tag{10}$$

By projecting the gradients onto the subspace defined by C and R, the updates to W are constrained. This means that changes during fine-tuning are less likely to interfere with the model's ability to perform the original task, potentially reducing interference with directions important for the original task.

4.3.4 Reduced Degree of Freedom

If $W \in \mathbb{R}^{m \times n}$ and we use a rank-k approximation, then:

- Full fine-tuning has mn degrees of freedom
- LoRA has $k(m+n)$ degrees of freedom
- CURLoRA has only k^2 degrees of freedom

This significant reduction in degrees of freedom inherently limits how far the model can stray from its original configuration.

4.3.5 Stability Analysis

We can analyze the stability of the adapted weights:

$$\|W_{\text{adapted}} - W\|_F = \|CUR - W\|_F \leq \|C\|_F \|U\|_F \|R\|_F \tag{11}$$

This equation provides an upper bound on how much the adapted weight matrix W_{adapted} can deviate from the original weight matrix W. It uses the submultiplicativity property of the Frobenius norm. Since C and R are fixed, the deviation from the original weights is directly controlled by the norm of U, which starts at zero and is the only part being updated. If C, U, and R are such that their Frobenius norms are relatively small, then the deviation will also be small. This ensures that the fine-tuning process does not drastically change the original model parameters, preserving the original model's knowledge and stability.

4.4 Theoretical Analysis of Output Shift

To understand why CURLoRA is expected to perform better than standard LoRA in terms of catastrophic forgetting, we can analyze the shift in the output distribution during fine-tuning.

For a given input x, the original output is $y = xW$. After fine-tuning:

For LoRA: $y_{\text{adapted}} = x(W + AB)$

For CURLoRA: $y_{\text{adapted}} = x(W + CUR)$

We can quantify the shift using the Frobenius norm of the difference:

$$\|y_{\text{adapted}} - y\|_F = \|x(W_{\text{adapted}} - W)\|_F \tag{12}$$

For LoRA: $\|x(AB)\|_F$

For CURLoRA: $\|x(CUR)\|_F$

This equation measures the shift in the model's output after fine-tuning. y is the original output, and y_{adapted} is the output after fine-tuning. After fine-tuning for a different task, the adapted output y_{adapted} might shift. We use the Frobenius norm to quantify this shift. If the shift is small, it means that the model's predictions haven't changed much, indicating that the model has retained its original knowledge.

Theoretically, CURLoRA should result in a smaller shift because:

1. The C and R matrices are directly sampled from W, maintaining some structure of the original matrix.
2. The C and R matrices are sampled from columns and rows with lower values.
3. Only U is trained, which is constrained by C and R.
4. The initialization of U as a zero matrix.

This constrained adaptation in CURLoRA is expected to lead to better preservation of the model's original knowledge, thereby reducing catastrophic forgetting.

4.5 Memory Efficiency

CURLoRA offers significant memory savings compared to full fine-tuning and even LoRA. For a weight matrix $W \in \mathbb{R}^{m \times n}$, the number of trainable parameters for each method is:

- Full fine-tuning: mn
- LoRA (rank r): $mr + nr$
- CURLoRA (rank r): r^2

The memory savings can be substantial, especially for large matrices. In our experiment, with rank 16, the trainable parameters were:

- Full fine-tuning: 7,248,023,552 parameters
- LoRA: 9,437,184 parameters
- CURLoRA: 24,576 parameters

This reduction in trainable parameters not only saves memory but also potentially leads to faster training and inference times.

In conclusion, CURLoRA provides multiple mathematical mechanisms that can help mitigate catastrophic forgetting:

- It constrains the parameter space of possible adaptations.
- It provides implicit regularization towards the original weights.
- It preserves important directions from the original weight matrix.
- It reduces the degrees of freedom in adaptation, limiting potential deviation.
- It allows for direct control and analysis of weight stability through the U matrix.

These properties suggest that CURLoRA can indeed help in reducing catastrophic forgetting while still allowing for meaningful and good adaptation to new tasks. The effectiveness of these theoretical mechanisms are validated through our experiments on various tasks and datasets, as detailed in the following sections.

5 Methodology

5.1 CURLoRA Algorithm

Our CURLoRA algorithm consists of the following steps:

1. **Decomposition**: For each weight matrix W in the attention layers (query, key, value), we perform the following:
    - Compute column probabilities: $p_j = \frac{\|W_{:j}\|_2^2}{\|W\|_F^2}$
    - Invert probabilities: $\tilde{p}_j = \frac{1/p_j}{\sum_k 1/p_k}$
    - Sample columns and rows according to $\tilde{p}_j$ to construct C and R
    - Initialize U as a zero matrix
2. **Fine-tuning**: Update only the U matrix during training, keeping C and R fixed.
3. **Inference**: Use the adapted weight matrix $W_{\text{adapted}} = CUR$ for forward passes along with the original W matrix i.e. $x(W + CUR)$.

6 Experiment Setup

6.1 Datasets

We used the following datasets for our experiments:

- GLUE-MRPC: Microsoft Research Paraphrase Corpus for paraphrase detection [13]
- GLUE-SST-2: Stanford Sentiment Treebank for binary sentiment classification [14]
  These datasets are part of the General Language Understanding Evaluation (GLUE) benchmark [15], which includes a diverse set of tasks for evaluating natural language understanding systems.
- Sentiment140: A large-scale sentiment analysis dataset [16]
- WikiText-2: A dataset that we use to measure language model perplexity [17]

For each fine-tuning task, we limited the training data to 1000 records to simulate scenarios with limited data availability.

6.2 Model and Hyperparameters

We used the Mistral 7B model [18] as our base model. For both LoRA and CURLoRA, we used the following hyperparameters:

- Rank: 16
- Alpha: 1
- Optimizer: AdamW
- Learning rate: 2.5e-4
- Scheduler: Cosine with 500 warmup steps
- Training epochs: 3

We did not use dropout to disable explicit regularization, allowing us to observe the implicit regularization effects of CURLoRA.

6.3 Evaluation Metrics

We used the following metrics for evaluation:

- Accuracy: For classification tasks (MRPC, SST-2, Sentiment140)
- Perplexity: For language modeling capability (WikiText-2)

6.4 Experimental Procedure

Our experimental procedure was as follows:

1. Measure initial perplexity of the base model on WikiText-2.

2. Fine-tune on MRPC and evaluate.
3. Fine-tune on SST-2 and evaluate, then re-evaluate on MRPC.
4. Fine-tune on Sentiment140 and evaluate, then re-evaluate on MRPC and SST-2.
5. Re-calculate perplexity on WikiText-2.

This procedure was carried out for both LoRA and CURLoRA independently.

7 Results and Discussion

Table 1 presents the results of our experiments comparing LoRA and CURLoRA across multiple tasks and evaluation metrics.

Table 1: Experimental Results: LoRA vs CURLoRA

Metric	LoRA	CURLoRA
Initial WikiText-2 Perplexity	5.44	5.44
MRPC Accuracy (After MRPC)	0.6495	0.6642
SST-2 Accuracy (After SST-2)	0.5092	0.8578
MRPC Accuracy (After SST-2)	0.3162	0.6642
Sentiment140 Accuracy	1.0000	0.9424
MRPC Accuracy (After Sentiment140)	0.3162	0.6642
SST-2 Accuracy (After Sentiment140)	0.4908	0.8578
Final WikiText-2 Perplexity	65055.02	5.44

7.1 Performance Analysis

7.1.1 Task-Specific Performance

CURLoRA consistently outperformed LoRA on the MRPC and SST-2 tasks, showing higher accuracy even after fine-tuning on subsequent tasks. This suggests that CURLoRA is more effective at preserving task-specific knowledge.

7.1.2 Catastrophic Forgetting

The stability of CURLoRA's performance across tasks is particularly noteworthy. While LoRA's accuracy on MRPC dropped from 0.6495 to 0.3162 after fine-tuning on other tasks, CURLoRA maintained its accuracy at 0.6642. This demonstrates CURLoRA's superior ability to mitigate catastrophic forgetting.

7.1.3 General Language Modeling Capability

The final perplexity scores on WikiText-2 provide strong evidence for CURLoRA's effectiveness in preserving general language modeling capabilities. While LoRA's perplexity increased dramatically from 5.44 to 65055.02, CURLoRA maintained the original perplexity of 5.44, indicating no degradation in general language understanding.

7.2 Theoretical Insights

The experimental results align with our theoretical analysis:

- **Parameter Space Constraint**: The stability of CURLoRA's performance across tasks supports our hypothesis that constraining adaptations to the subspace spanned by C and R helps preserve original knowledge.
- **Implicit Regularization**: The maintained perplexity on WikiText-2 suggests that CURLoRA's implicit regularization effectively prevents overfitting to specific tasks.
- **Reduced Interference**: The consistent performance across tasks indicates that CURLoRA successfully reduces interference between task-specific adaptations.

7.3 Limitations and Future Work

While CURLoRA shows promising results, there are several areas for future research:

- **Optimal Rank Selection**: Investigating methods for automatically selecting the optimal rank for CUR decomposition could further improve performance.
- **Combination with Other Techniques**: Exploring the integration of CURLoRA with other continual learning techniques could yield even better results.

8 Conclusion

This paper introduced CURLoRA, a novel approach to fine-tuning large language models that leverages CUR matrix decomposition to mitigate catastrophic forgetting and improve computational efficiency. Through theoretical analysis and empirical experiments, we demonstrated that CURLoRA outperforms standard LoRA in maintaining model stability and performance across tasks while significantly reducing the number of trainable parameters.

Key contributions of this work include:

- A novel modification to CUR decomposition using inverted probabilities for column and row selection and initiating U matrix as zeros.
- Theoretical analysis of how CURLoRA addresses catastrophic forgetting.
- Empirical evidence of CURLoRA's effectiveness across multiple tasks and evaluation metrics.

Our results suggest that CURLoRA is a promising approach for efficient and stable fine-tuning of large language models, particularly in scenarios with limited fine-tuning data.

References

[1] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[2] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of learning and motivation*, 24:109–165, 1989.

[3] Xiang Lisa Li and Percy Liang. Efficient few-shot learning without prompts. *arXiv preprint arXiv:2111.10952*, 2021.

[4] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2021.

[5] Michael W Mahoney and Petros Drineas. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[6] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[7] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 8154–8162, 2016.

[8] David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[9] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.

[10] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[11] Petros Drineas, Michael W Mahoney, and S Muthukrishnan. Relative-error cur matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.

[12] Shusen Wang, Zhihua Zhang, and Guiying Guo. Efficient and effective knn ensemble classification with efficient and effective knn ensemble classification with homogeneous or heterogeneous learners. *IEEE transactions on cybernetics*, 48(6):1757–1770, 2016.

[13] William B Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005.

[14] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642, 2013.

[15] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, 2018.

[16] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. In *CS224N project report, Stanford*, volume 1, page 2009, 2009.

[17] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

[18] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.