

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and application of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. The issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, contrasting with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring hyper-parameters across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to our rationale of extending the network in parallel. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of empirical evidence or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. The issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, contrasting with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring weights across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the rationale of extending LoRA in parallel. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of empirical evidence or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. The issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, contrasting with the Adapter's sequential approach.



Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring knowledge across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the limitations of extending HPT in depth. LoRA, in contrast, extends knowledge in parallel, unlike the Adapter. Indeed, this is due to the lack of depth dependency in LoRA's core mechanism or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown promising applications and that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and application of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. The issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, contrasting with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring weights across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the limitations of extending HPT in depth. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of depth dependency in LoRA or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel along with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring knowledge across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credence to the limitations of HPT. Extending HPT works in LoRA because LoRA does, rather than sequentially like the Adapter. Indeed, the LoRA's lack of depth dependency is a source or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel rather than with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al [VHB<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring hyper-parameters across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the limitations of extending HPT in depth. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of empirical evidence or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, in contrast to the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHR<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring hyper-parameters across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the limitations of extending HPT in depth. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of empirical evidence or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and application of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer GPT-3 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that extends weights in parallel, in contrast to the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YH21] on hyper-parameter transfer (HPT) demonstrated the practicality of transferring weights across a model's width. However, their attempts to extend HPT along the model's depth were unsuccessful. This lent further credibility to the rationale behind extending the network in its width. LoRA does, rather than sequentially, like the Adapter. Indeed, this is due to the lack of depth dependency in practice or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets achieving training convergence became challenging, particularly when working with the 96-layer BERT-1.35 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that crowds weights in parallel with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and al. [YH<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of parallelizing adaptations across a model's width. However, their attempts to extend HPT along the depth dimension were less successful. This lent further credibility to the rationale behind LoRA's design. In fact, the LoRA authors did not provide any theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al. 2023] has shown that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is notable for its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

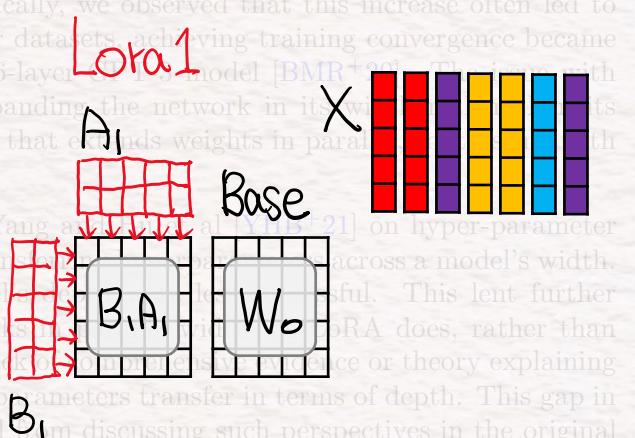
## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets achieving training convergence became challenging, particularly when working with the 96-layer BERT-1.35 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that expands weights in parallel across the Adapter's sequential approach.

Around the same time, a separate project led by Yang and al. [YH<sup>+</sup>21] on hyper-parameter transfer (HPT) demonstrated the practicality of parallelizing adaptations across a model's width. However, their attempts to extend HPT along the depth dimension were unsuccessful. This lent further credibility to the rationale behind LoRA. In fact, LoRA does, rather than sequentially, parallelize the Adapter's components. Indeed, this is a stark contrast to the Adapter's sequential approach. There is no literature or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.



During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt-based modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

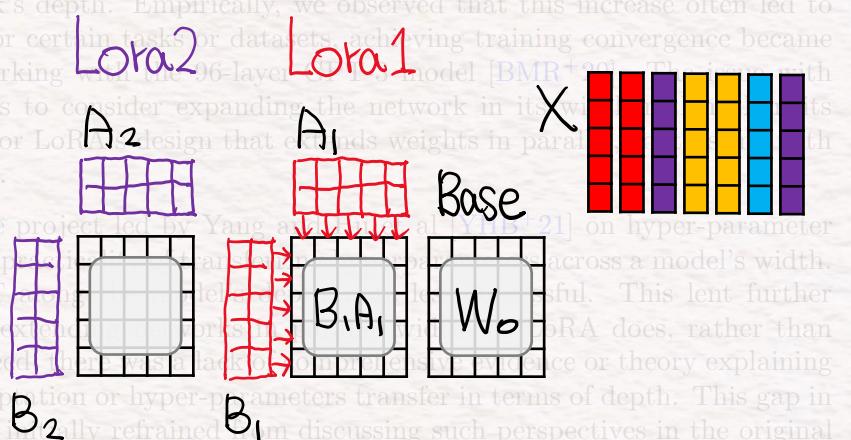
LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets achieving training convergence became challenging, particularly when working with the 96-layer BART-LM model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel with the Adapter's sequential approach.



Around the same time, a separate project of Yang and al. [YH21] on hyper-parameter transfer (HPT) demonstrated that Prefix Tuning [LL21] offers a more efficient way to adapt a model across a model's width. However, their attempts to extend HPT to LoRA were unsuccessful. This lent further credence to the Adapter's sequential nature. In contrast, LoRA does, rather than sequentially, what the Adapter does sequentially. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al. 2023] has shown that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

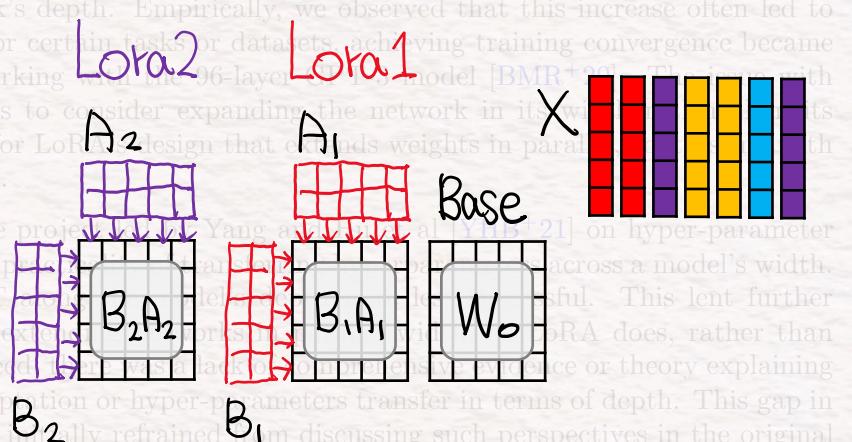
LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets achieving training convergence became challenging, particularly when working with the 96-layer BERT-15 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel with the Adapter's sequential approach.



Around the same time, a separate project [Yang et al., 2021] on hyper-parameter transfer (HPT) demonstrated Prefix Tuning [LL21] across a model's width. However, our attempts to extend HPT to LoRA were unsuccessful. This lent further credence to the Adapter's sequential nature. In contrast, LoRA does, rather than sequentially, parallelize the Adapter's task. This lack of depth is a key difference or theory explaining the difficulties in either model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Zhou et al., 2023] has shown that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer BERT-15 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel with the Adapter's sequential approach.

Around the same time, our late project transitioned from Adapter to LoRA [Yang et al., 2021] on hyper-parameter transfer [Huang et al., 2021]. However, the adapter's sequential nature posed a significant challenge. The first problem was that the adapter's sequential nature required extra layers, which increased the model's depth. This led to further instability during training. The second problem was that the adapter's sequential nature did not align with the LoRA's parallel nature. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al., 2023] has shown that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\* Han Yu Jongho Lee Stanley Hsieh Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

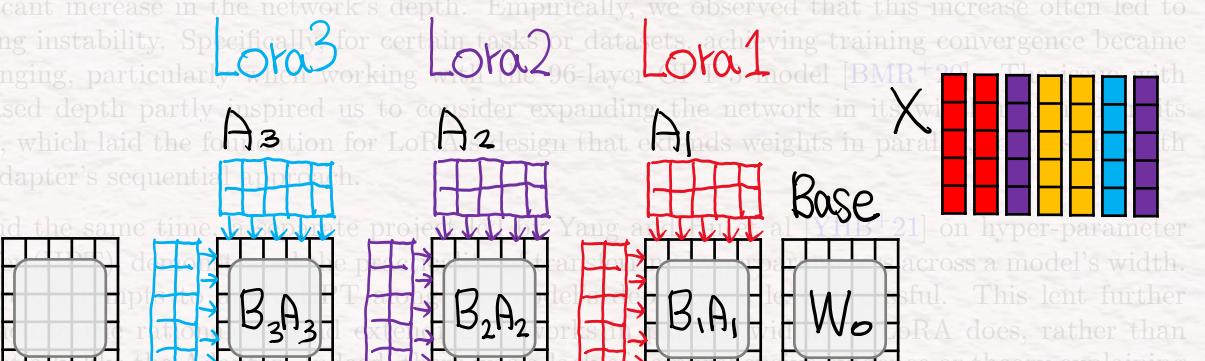
LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became challenging, particularly when working with the 96-layer BERT-1.35 model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel with the Adapter's sequential approach.



Around the same time, the project [Yang et al., 2021] on hyper-parameter transfer [HPT<sup>21</sup>] did the same thing. However, the approach taken by LoRA is more efficient and general. This led further research to focus on LoRA. LoRA does, rather than sequence the Adapter's approach, parallelize it across a model's width. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al., 2023] has shown that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became

challenging, particularly when working with the 96-layer BART model [BMR<sup>+</sup>21]. The issue with increased depth partly inspired us to consider expanding the network in its width. This laid the foundation for LoRA's design that avoids weights in parallel, the Adapter's sequential approach.

Around the same time, the project [Yang et al., 2021] on hyper-parameter tuning across a model's width. Yang and colleagues found that increasing the number of layers per block was beneficial. This lent further support to the LoRA's design choice. However, there is no evidence or theory explaining the difficulties in other model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al., 2023] shows that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

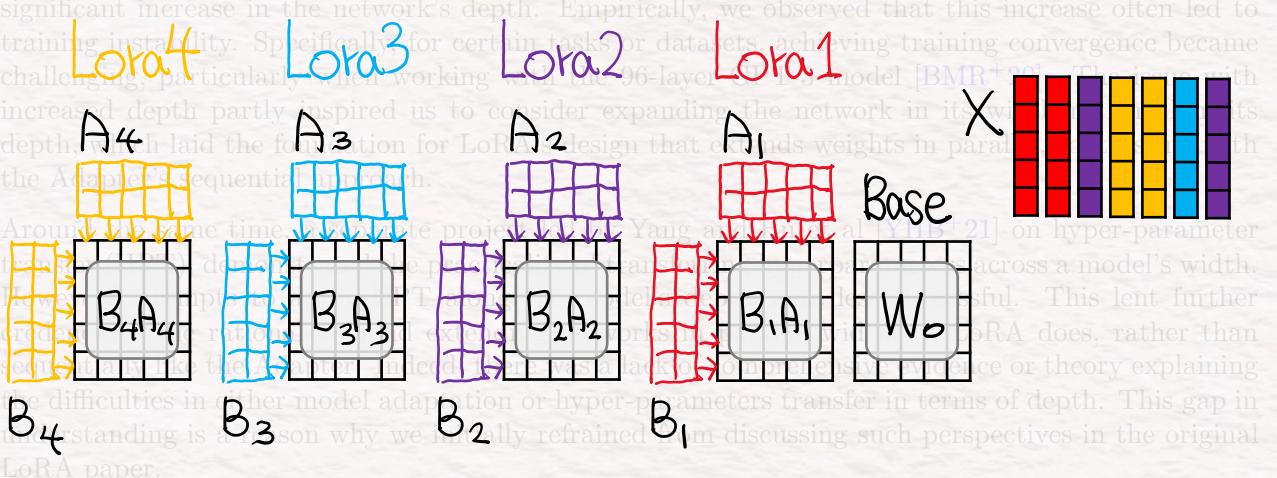
LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It is remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17] layer, one after the attention and the other after the feed-forward modules. This not only leads to extra inference latency, particularly with smaller batch sizes as highlighted in the LoRA study, but it also causes a significant increase in the network's depth. Empirically, we observed that this increase often led to training instability. Specifically, for certain tasks or datasets, achieving training convergence became



During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown promising applications and that changes in the model's internal structure are crucial,

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. It achieves remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it didn't fully explain why we designed LoRA in such a way or how it tackles the challenges born in other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [HGJ<sup>+</sup>19]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>17], layer one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training convergence becomes challenging, particularly when working with the 96-layer LLaMA-13B model [BMR<sup>+</sup>23]. This issue laid the foundation for LoRA's design that avoids weights in parallel across the Adapter's sequential modules.

Around the same time, the project [Yang et al., 2021] on hyper-parameter tuning across a model's width. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [Liu et al., 2023] has shown that changes in the model's internal structure are crucial,

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

## A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen<sup>†</sup>

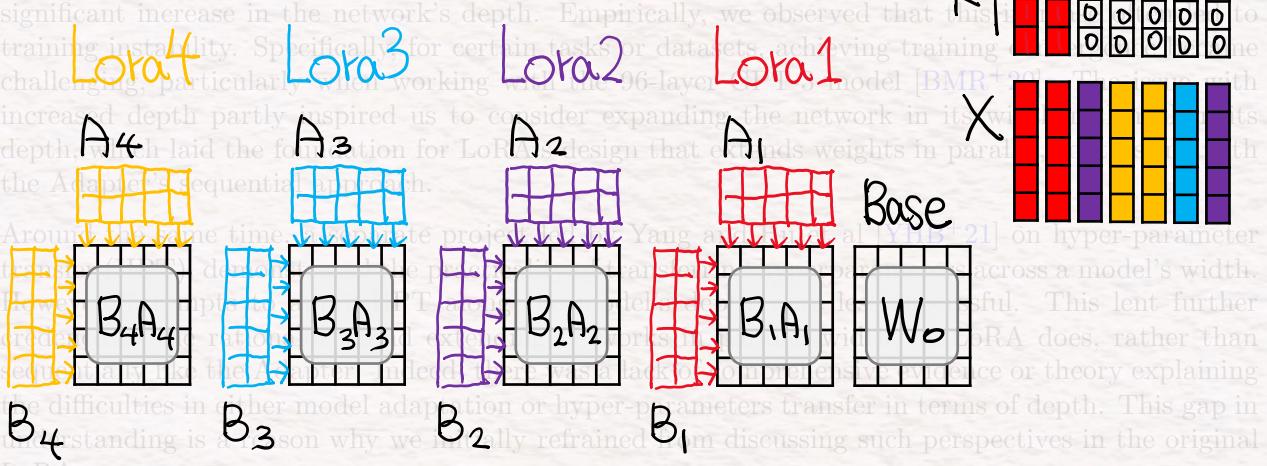
Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

**How to fit many Loras in 1 GPU?**

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of these other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [17]. This method sequentially integrates two adaptation modules in each Transformer [VSP-17]. One module is placed after the attention and the other after the feed-forward modules. This not only leads to better performance but also reduces the number of parameters.



During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [LARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for achieving full readability and consistency across all applications and that changes in the model's internal structure are often necessary.

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

<sup>†</sup>Weizhu completed the majority of the manuscript. Vlad edited the manuscript and drafted section 2.1.

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficiently adapting large models. Its remarkable simplicity and efficacy. This note extends the original LoRA paper by offering new perspectives that were not initially discussed and presents a series of insights for deploying LoRA at scale. Without introducing new experiments, we aim to improve the general understanding and adoption of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP<sup>+</sup>20]. It sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>20], one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 96-layer BART-L15 model [BMR<sup>+</sup>22].

This challenge, along with the need to parallelize the adapter modules across a model's width, inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel across the Adapter's sequential modules.

Around the same time, the project team at Alibaba Cloud [YH<sup>+</sup>21] on hyper-parameter optimization for LoRA [LH<sup>+</sup>21] on hyper-parameter transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work [ZL<sup>+</sup>24] has shown promising applications and that changes in the model's internal structure are crucial,

AI by Hand  2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

How to fit many Loras in 1 GPU?

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP20]. This method sequentially integrates two adaptation modules in each Transformer [VSP20]: one after the attention and the other after the feed-forward modules. This not only leads to significant increases in latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this sequential design leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 96-layer BERT-1.3 model [BMR+21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that avoids weights in parallel. The Adapter's sequential approach, on the other hand, is less efficient and less memory-friendly. Around the same time, the project team at Alibaba Cloud [YHJ+21] focused on hyper-parameter tuning to adapt the LoRA design to the Adapter's sequential approach. Yang et al. [YHJ+21] also highlighted the difficulties in understanding the impact of LoRA across a model's width. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LL21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its reduction of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt modifications and that changes in the model's internal structure are crucial.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

# How to fit many Loras in 1 GPU?

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP+20]. This method sequentially integrates two adaptation modules in each Transformer [VSP+20]: one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach can lead to training instability. Specifically, for certain tasks or datasets, achieving training convergence becomes challenging, particularly when working with the 96-layer BART-L13 model [BMB+21]. This issue with increased depth partly inspired us to consider expanding the network in width rather than in depth, which laid the foundation for LoRA's design that avoids weights in parallel. This approach, known as the Adapter Sequential method, is illustrated in Figure 1. Around the same time, the project team at Alibaba Cloud [YHJ+21] on hyper-parameter tuning proposed the Prefix Tuning approach. This approach, similar to LoRA, focuses on adapting the model's context length. Yang et al. [YHJ+21] found that Prefix Tuning is less effective than LoRA across a model's width. This led further research into LoRA's design choices. We believe that LoRA does, rather than follows, what makes sense from a practical perspective or theory explaining the difficulties in other model adaptation or hyper-parameters transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [YHJ+21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel approach, its requirement of adapting the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown promising applications and that changes in the model's internal structure are crucial,

W<sub>b</sub>X

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft  
vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

How to fit many Loras in 1 GPU?

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP20]. This method sequentially integrates two adaptation modules in each Transformer [VSP20]: one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 96-layer Adapter model [BMR+21]. This issue with increased depth partly inspired us to consider expanding the network in its width rather than its depth, which laid the foundation for LoRA's design that encodes weights in parallel. This is in contrast to the Adapter's sequential approach.

Around the same time, the project team at Alibaba Cloud [YH21] on hyper-parameter tuning [HPS21] and Yang et al. [YH21] on hyper-parameter transfer in terms of depth. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

During our exploration of LoRA, we concurrently examined Prefix Tuning [LARC21] and Prompt Tuning [EARC21]. Although Prefix Tuning offered a novel way to control the model's context length, it posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown promising applications and that changes in the model's internal structure are crucial.

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

## A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen<sup>†</sup>

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

# How to fit many Loras in 1 GPU?

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it didn't fully explain why we designed LoRA in such a way or how it tackles the following challenges.

During our exploration of LoRA, we recently examined Prefix Tuning [D21] and Prompt Tuning [EARC21]. Although Prefix Tuning is a novel approach, the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for addressing LoRA's limitations and that changes in the model's internal structure are necessary.

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

<sup>†</sup>Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

## A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen<sup>†</sup>

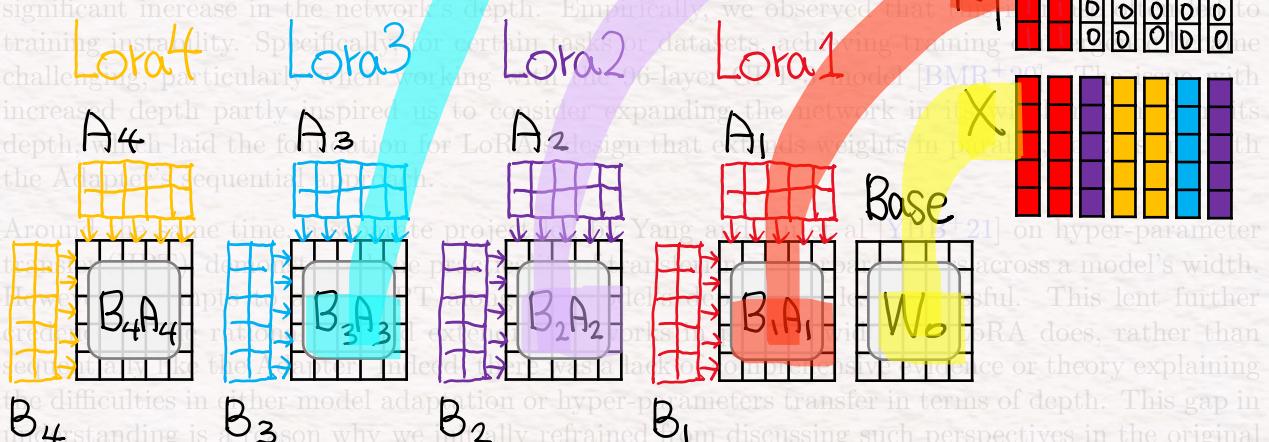
Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

How to fit many Loras in 1 GPU? R4

Although the original LoRA paper compared LoRA with a variety of alternative approaches, it did not fully explain why we designed LoRA in such a way or how it tackles the following challenges.

Back in 2020, the predominant parameter-efficient adaptation technique was AdaP [VSP+20], which sequentially integrates two adaptation modules in each Transformer layer. This not only leads to significant latency increase, particularly with smaller batch sizes as highlighted in the LoRA paper [BMR+22]. Empirically, we observed that training instability. Specifically, for certain flash or datasets, achieving training stability becomes challenging, particularly when working with larger 6-layer models [BMR+22]. This laid the foundation for LoRA's design that eliminates weights in parallel layers. The diagram illustrates the sequential integration of four adaptation modules (A<sub>1</sub>, A<sub>2</sub>, A<sub>3</sub>, A<sub>4</sub>) across four layers (Lora1, Lora2, Lora3, Lora4). Each module consists of a weight matrix (red) and a bias vector (blue). The sequential nature of AdaP is shown by the vertical stack of modules, where each layer receives the output of the previous layer as input.



During our exploration of LoRA, we concurrently examined Prefix Tuning [K21] and Prompt Tuning [EARC21]. Although Prefix Tuning introduced novel mechanisms to manage the model's context length, it posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for achieving full generality in large language models and that changes in the model's internal structure are often necessary.

AI by Hand 🤝 2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

<sup>†</sup>Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficient model adaptation. This note aims to highlight the remarkable simplicity and efficacy of the original LoRA paper by offering new perspectives that were not initially discussed. We present a series of insights for deploying LoRA at scale. Without introducing new overheads, we show how to fit many Loras in one GPU, and how to better understand and utilize the power of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP<sup>+</sup>20]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>20], one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 16-layer Adapter model [BMR<sup>+</sup>21].

Increased depth partly inspired us to consider expanding the network in width rather than depth, which laid the foundation for LoRA's design that encodes weights in parallel. This is similar to the Adapter's sequential integration, but it is more efficient and stable. Specifically, we found that training stability is significantly improved when weights are shared across a model's width. This is because the Adapter's sequential integration leads to redundant calculations, while LoRA does, rather than adds, weight matrices. This is a key difference between LoRA and Adapter, despite both being based on the same underlying principle of low-rank approximation.

Around the same time, we also explored the projective adapter [YH<sup>+</sup>21] on hyper-parameter tuning. The projective adapter is a variant of the Adapter that uses a linear projection layer to map the input features to the output features. This approach is more efficient than the Adapter, but it still suffers from the same issues of increased depth and training instability. We found that the projective adapter's performance is heavily dependent on the choice of hyper-parameters, and it is difficult to find a set of hyper-parameters that work well for all tasks and datasets. This lack of generalization is a significant limitation of the projective adapter. In contrast, LoRA is more robust and generalizable, as it can handle different tasks and datasets without requiring significant changes to its hyper-parameters. This is because LoRA's design is based on a more principled and data-driven approach, which takes into account the specific characteristics of the task and dataset being adapted.

Bringing the exploration of LoRA, we concurrently examined LoRAfix Tuning [LW<sup>+</sup>21] and Prompt Tuning [LW<sup>+</sup>21]. Although LoRAfix Tuning is a novel approach to adapt models to longer contexts, the length of the model's context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While LoRAfix Tuning is promising for certain applications and that changes in the model's internal structure are crucial.

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficient model adaptation. This note provides a markable simplicity and efficacy of the original LoRA paper by offering new perspectives that were not initially discussed. It also presents a series of insights for deploying LoRA at scale. Without introducing new overhead, we show how to fit more Loras in one GPU, and how to better understand and utilize the power of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP<sup>+</sup>20]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>20], one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 16-layer Adapter model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in parallel instead of sequentially. Around the same time, the project [Yang et al., 2021] proposed LoRA, which laid the foundation for LoRA's design that conducts weight in parallel across the Adapter's sequential modules.

Around the same time, the project [Yang et al., 2021] proposed LoRA, which laid the foundation for LoRA's design that conducts weight in parallel across the Adapter's sequential modules. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

Bringing the exploration of LoRA, we concurrently examined LoFix Tuning [KZL<sup>+</sup>21] and Prompt Tuning [AHL<sup>+</sup>21]. Although LoFix Tuning introduced a novel way of adapting models to their context, its length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for the model's internal structure. This note also highlights that changes in the model's internal structure are crucial.

AI by Hand  2024 © Tom Yeh

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1

# A Note on LoRA

Vlad Fomenko\*

Han Yu

Jongho Lee

Stanley Hsieh

Weizhu Chen†

Microsoft

vlfomenk@gmail.com wzchen@microsoft.com

## Abstract

LoRA (Low-Rank Adaptation) [HSW<sup>+</sup>21] has emerged as a preferred method for efficient model adaptation. This note aims to highlight the remarkable simplicity and efficacy of the original LoRA paper by offering new perspectives that were not initially discussed. We present a series of insights for deploying LoRA at scale. Without introducing new overheads, we show how to fit many Loras in one GPU, and how to better understand and utilize the power of LoRA.

## 1 Additional Insights

### 1.1 On Comparison

Although the original LoRA paper compared LoRA with a variety of alternative methods, it did not fully explain why we designed LoRA in such a way or how it tackles the challenges of other approaches.

Back in 2020, the predominant parameter-efficient adaptation technique was Adapter [VSP<sup>+</sup>20]. This method sequentially integrates two adaptation modules in each Transformer [VSP<sup>+</sup>20], one after the attention and the other after the feed-forward modules. This not only leads to significant latency, particularly with smaller batch sizes as highlighted in the LoRA paper, but also to a significant increase in the network's depth. Empirically, we observed that this approach leads to training instability. Specifically, for certain tasks or datasets, achieving training stability becomes challenging, particularly when working with the 16-layer Adapter model [BMR<sup>+</sup>21]. This issue with increased depth partly inspired us to consider expanding the network in parallel instead of sequentially. Around the same time, the project [Yang et al., 2021] proposed LoRA, which laid the foundation for LoRA's design that encodes weights in parallel across the Adapter's sequential modules.

Around the same time, the project [Yang et al., 2021] proposed LoRA, which laid the foundation for LoRA's design that encodes weights in parallel across the Adapter's sequential modules. This gap in understanding is a reason why we initially refrained from discussing such perspectives in the original LoRA paper.

Bringing the exploration of LoRA, we concurrently expanded Prefix Tuning [KZK<sup>+</sup>21] and Prompt Tuning [Liu et al., 2021]. Although Prefix Tuning initially showed promise, its requirement of maintaining context length posed a significant limitation. In contrast, Prompt Tuning, despite showing potential, delivered inconsistent outcomes across different datasets in our tests. This underscored that input-level modifications may not suffice for some tasks. While recent work has shown that prompt additions and that changes in the model's internal structure are crucial,

LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

\*work done while at Microsoft

†Weizhu completed the majority of the manuscript, Vlad edited the manuscript and drafted section 2.1