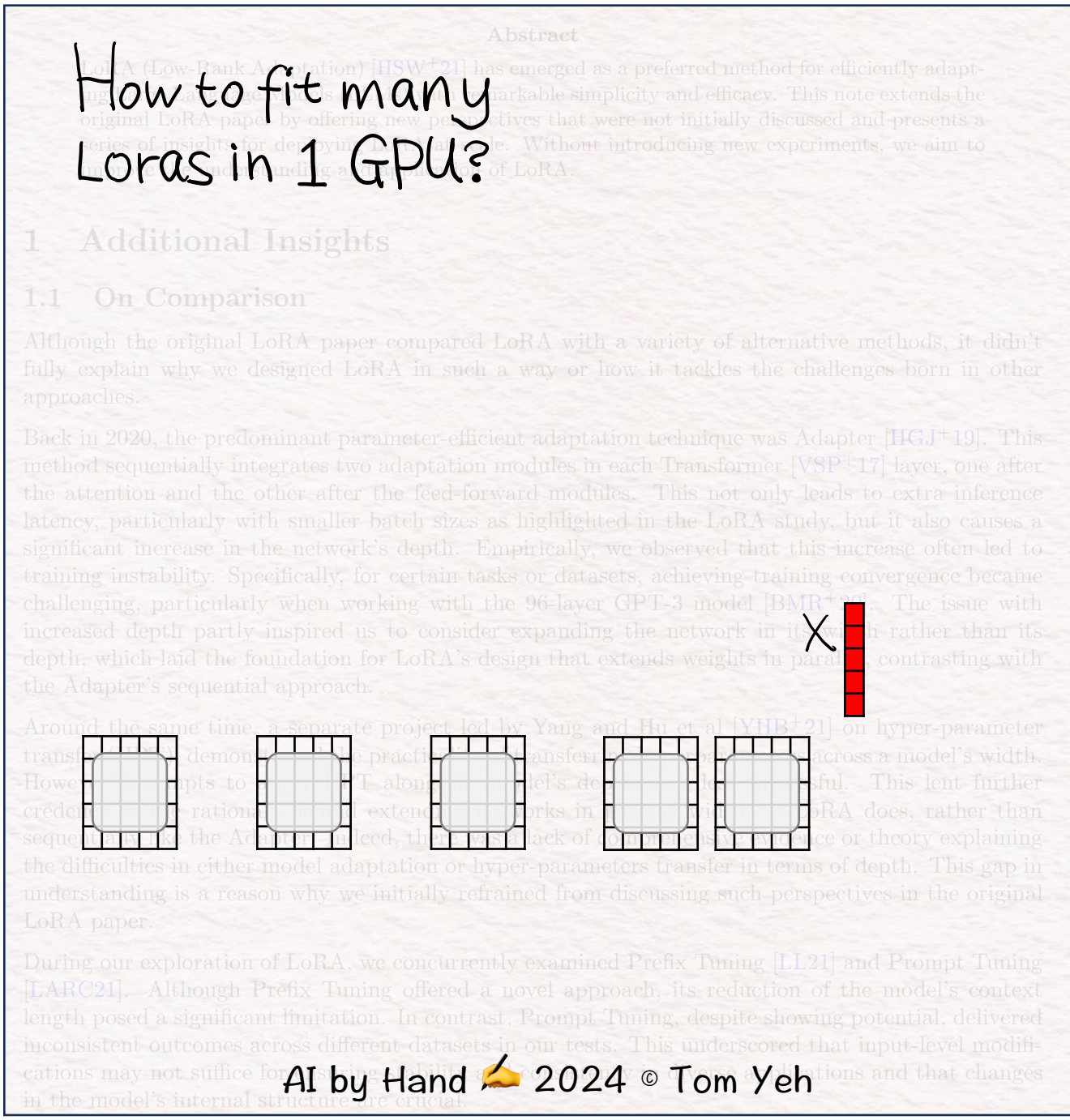
A Note on LoRA

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

Microsoft
vlfomenk@gmail.com wzchen@microsoft.com



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting both text and diffusion models, noted 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 clarify common points of confusion and enhance understanding of the underlying mechanisms 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. 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 [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to apply LoRA alone to the model's depth were unsuccessful. This lent further credence to our rationale for extending only two weights in parallel, as LoRA does, rather than sequentially as the Adapter method. Indeed, the lack of comprehensive 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 [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 reliable performance adaptations and that changes in the model's internal structure are crucial.

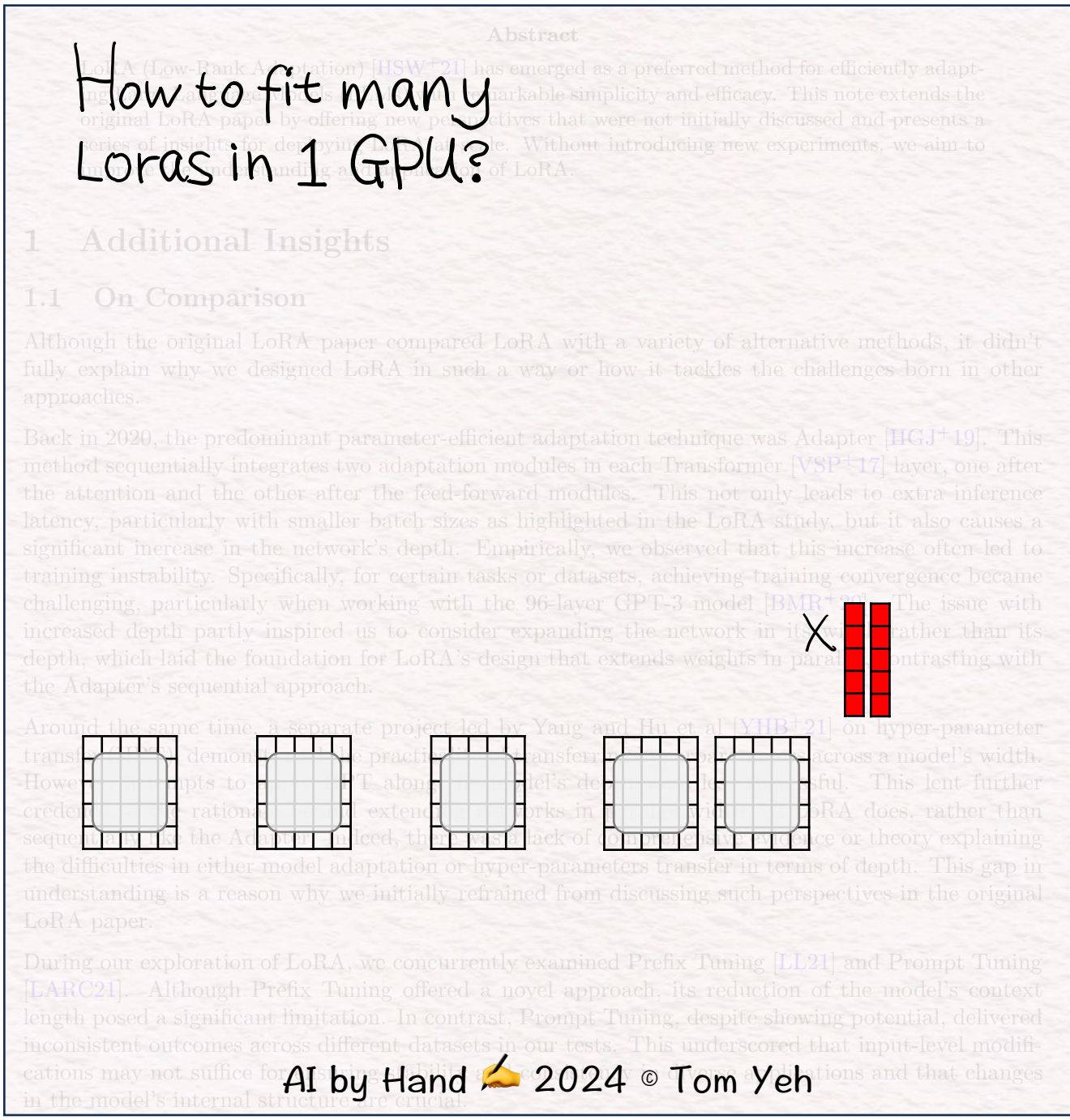
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting Large Language Models with 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 enhance the 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. 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 [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to transfer FT alone model's depth were unsuccessful. This lent further credence to the rationale behind extending weights in width, as LoRA does, rather than sequentially adding to the Adapter. Indeed, the lack of comprehensive 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 [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 [illegible] changes in activations and that changes in the model's internal structure are crucial.

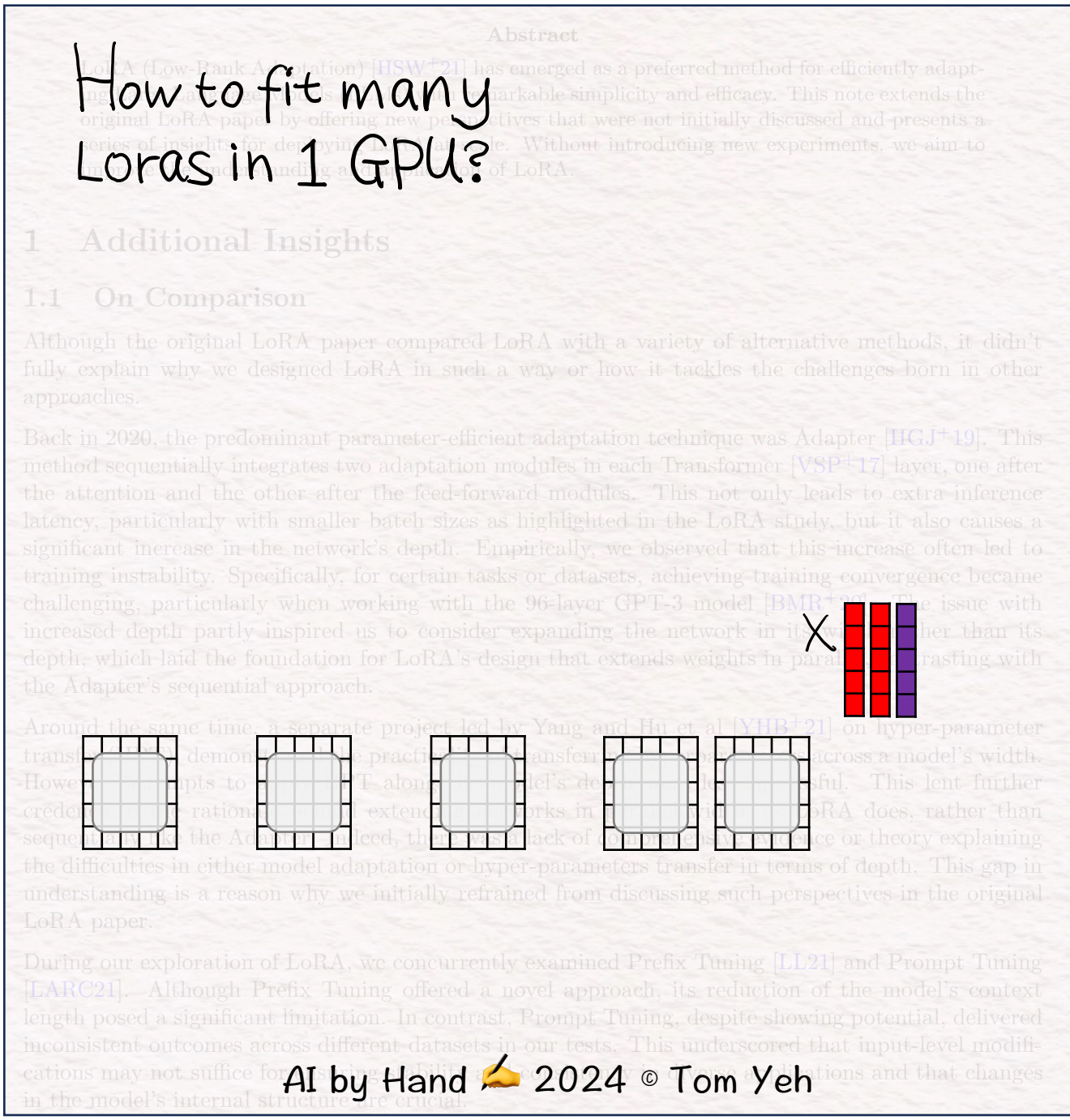
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting both text and diffusion models, recognized 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 clarify common points of confusion and enhance the understanding 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. 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 [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to transfer GPT alone model's depth were unsuccessful. This lent further credence to our rationale for extending weights in parallel, as LoRA does, rather than sequentially as the Adapter method. Indeed, the lack of comprehensive 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 [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 significant performance changes or adaptations and that changes in the model's internal structure are crucial.

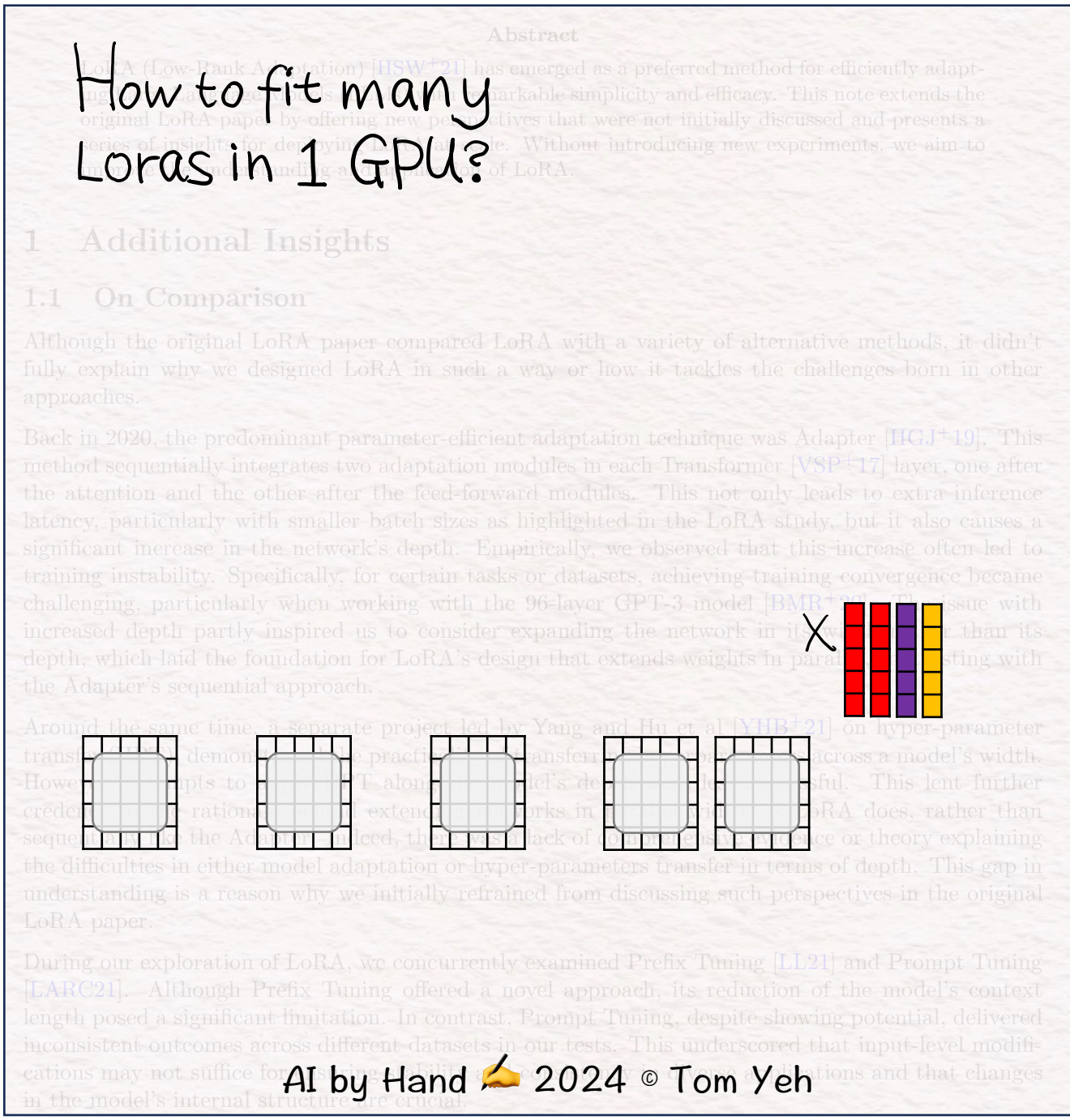
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting Large Language Models with 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 clarify common confusions and enrich the understanding 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. 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, contrasting with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to transfer GPT along the model's depth were unsuccessful. This lent further credence to our rationale for extending weights in parallel, as LoRA does, rather than sequentially as the Adapter method. Indeed, there's a lack of comprehensive 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 [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 definitive performance alterations and that changes in the model's internal structure are crucial.

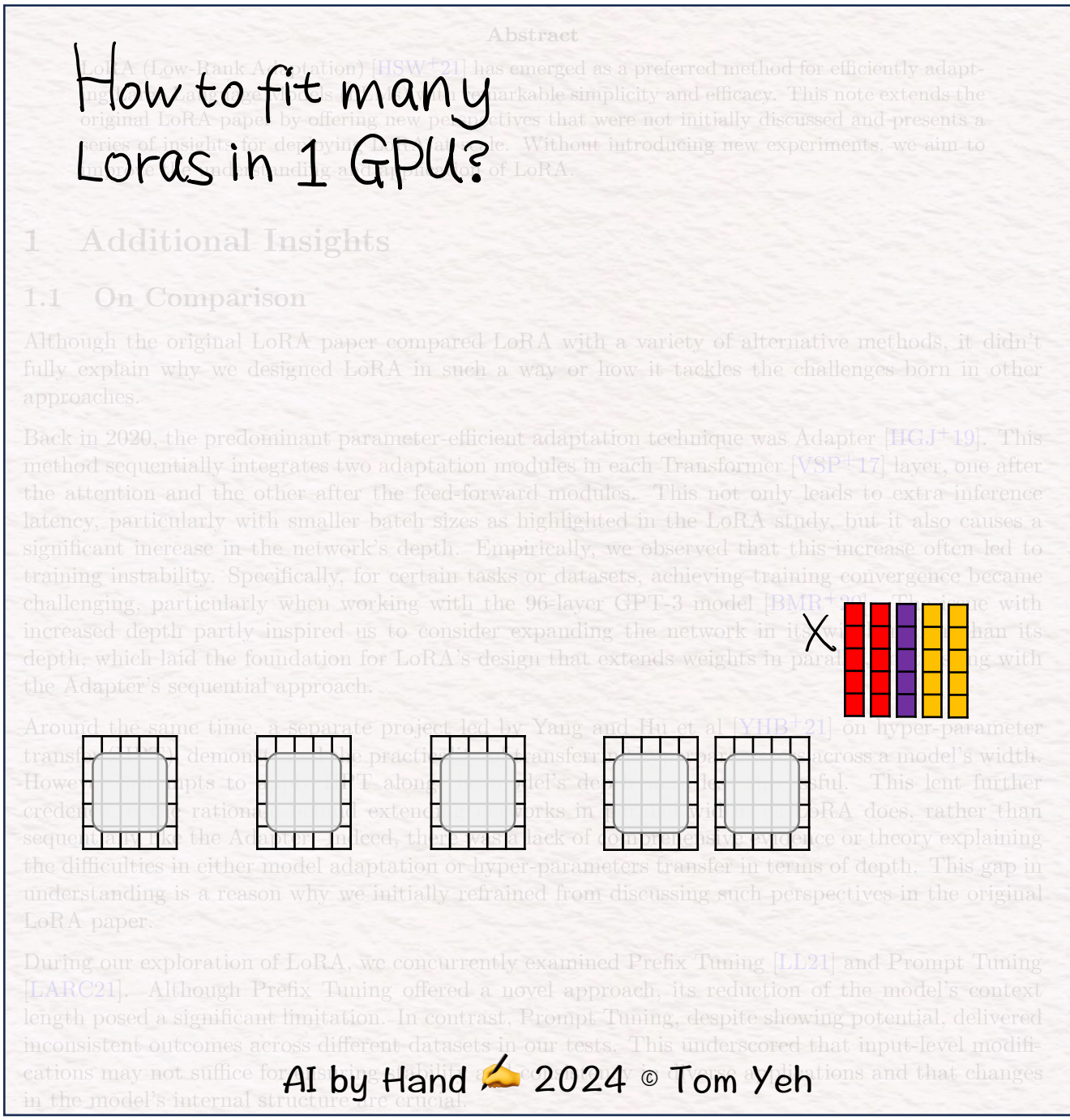
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



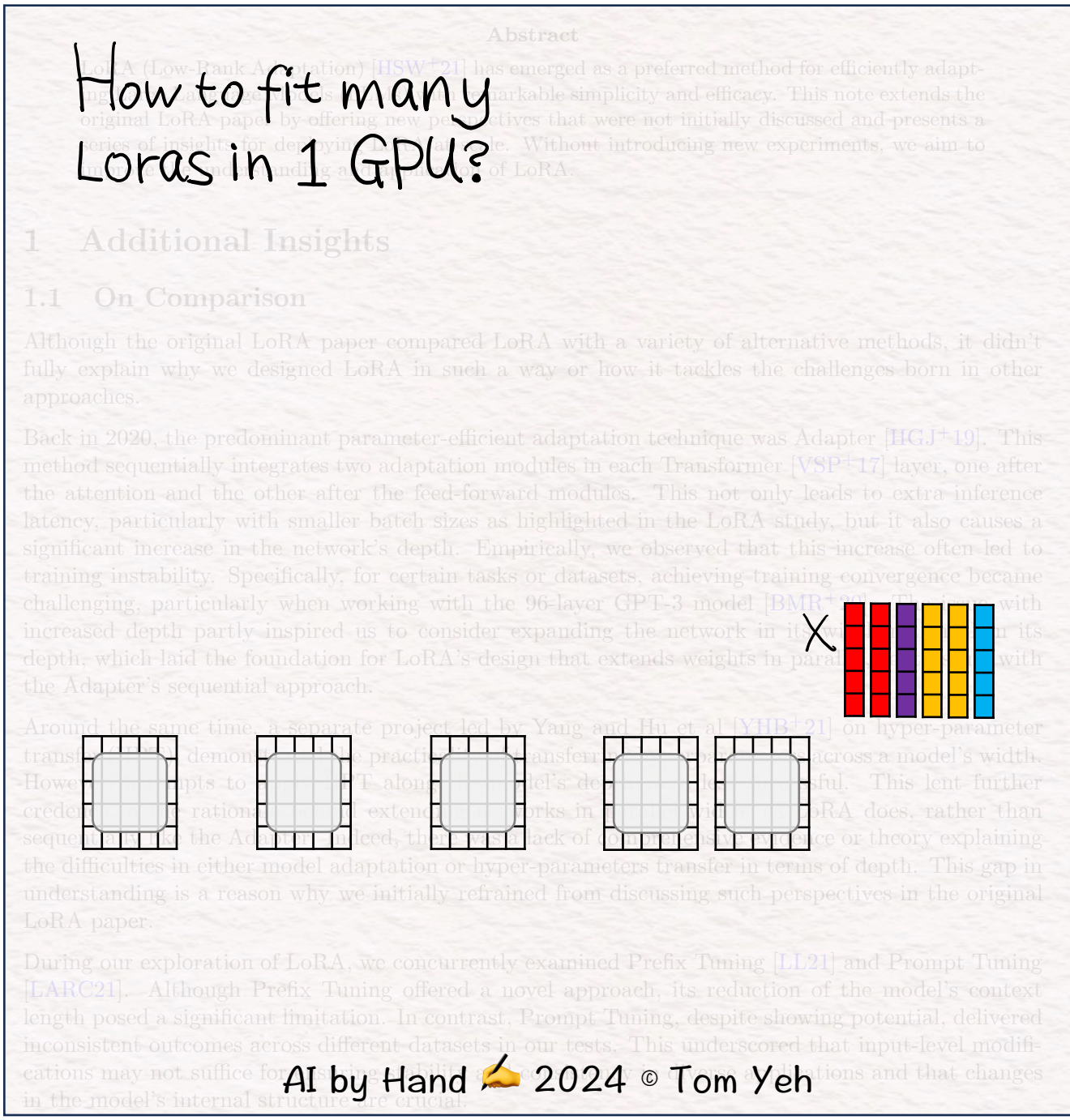
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



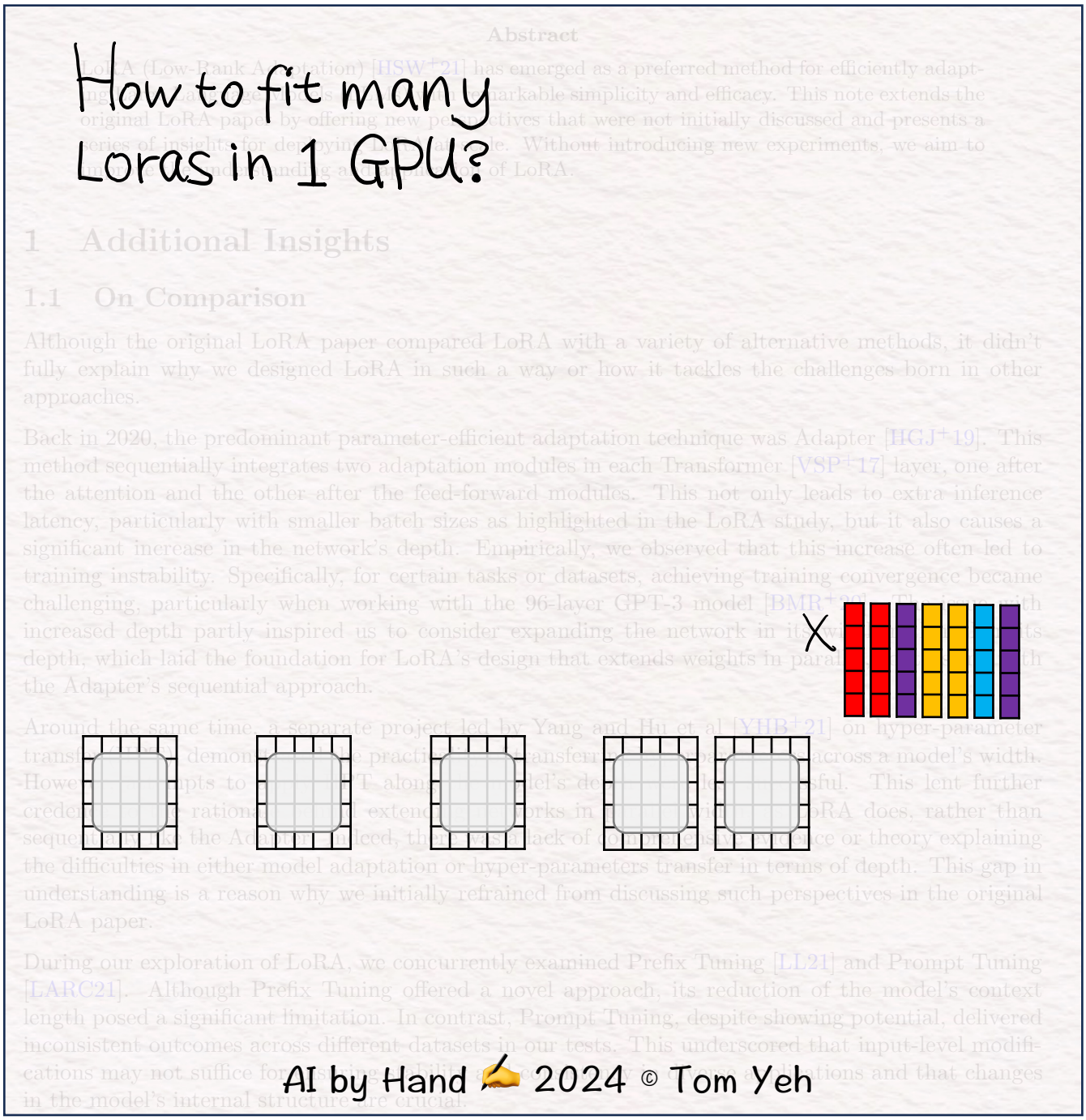
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting Large Language Models with 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 enhance the 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. This observation with increased depth partly inspired us to consider expanding the network in its width dimension instead of 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 [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to transfer GPT along the model's depth were unsuccessful. This lent further credence to our rationale to extend weights in parallel, as LoRA does, rather than sequentially as the Adapter method. Indeed, the current lack of comprehensive 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 [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 adapting billion-parameter language foundations and that changes in the model's internal structure are crucial.

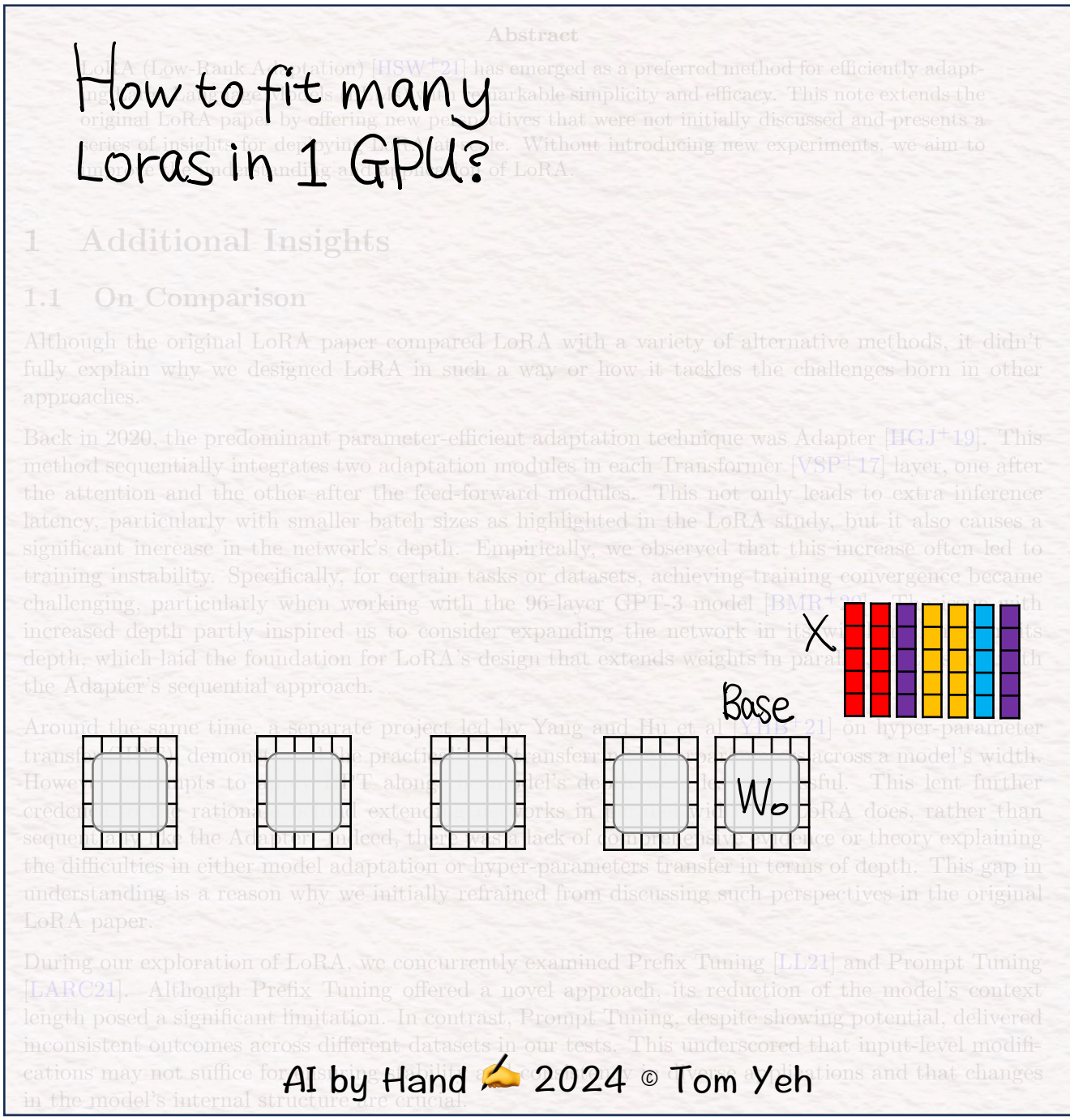
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



LoRA (Low-Rank Adaptation) [HSW+21] has emerged as a preferred method for efficiently adapting Large Language Models with 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 enhance the 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+19]. This method sequentially integrates two adaptation modules in each Transformer [VSP+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+20]. 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, contrasting with the Adapter's sequential approach.

Around the same time, a separate project led by Yang and Hu et al. [YHB+21] on hyper-parameter transfer demonstrated practical transferability across a model's width. However, attempts to extend GPT alone model's depth were unsuccessful. This lent further credence to our rationale for extending only weights in parallel, as LoRA does, rather than sequentially as the Adapter method. Indeed, the lack of comprehensive 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 [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 significant performance alterations and that changes in the model's internal structure are crucial.

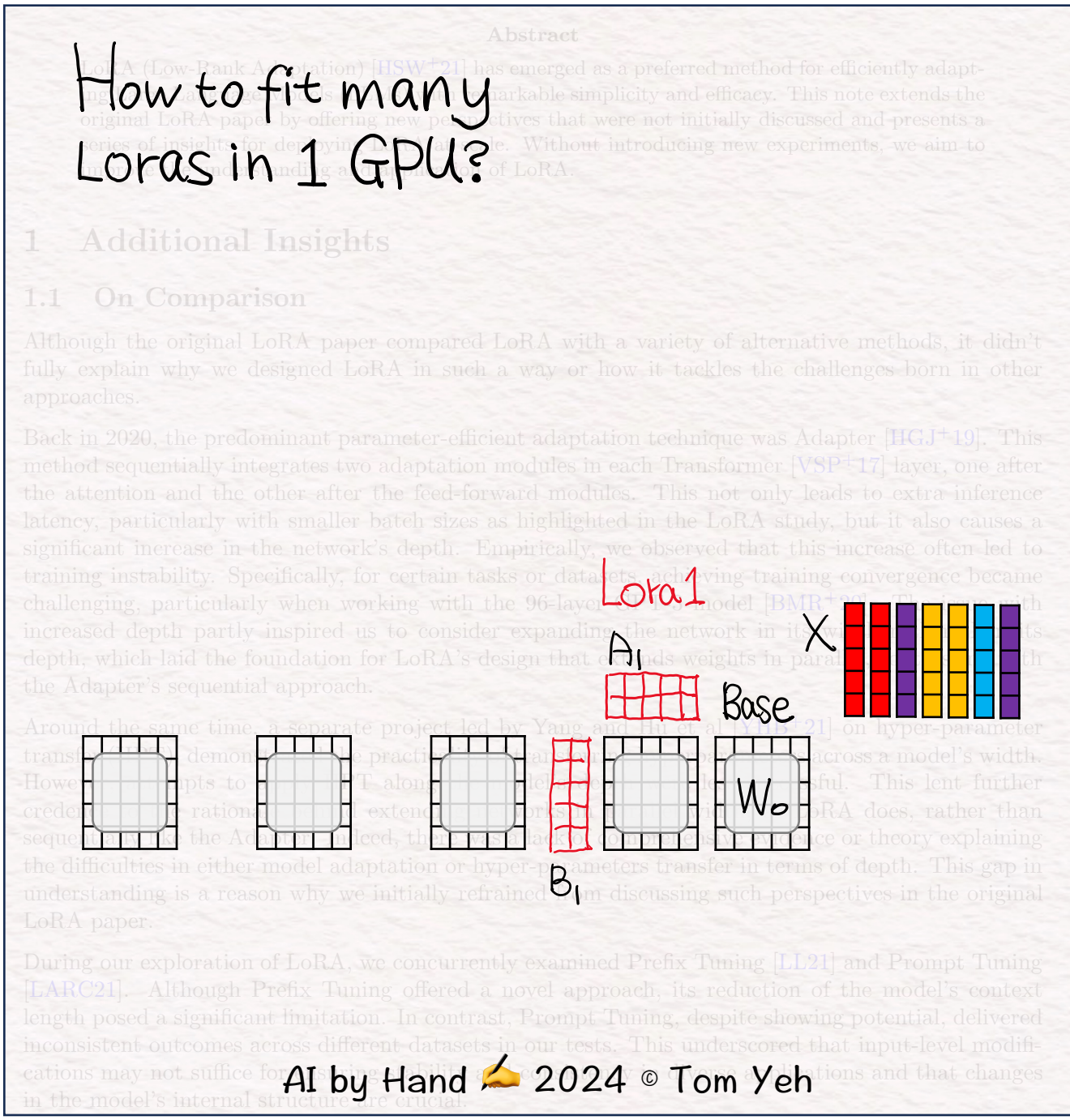
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



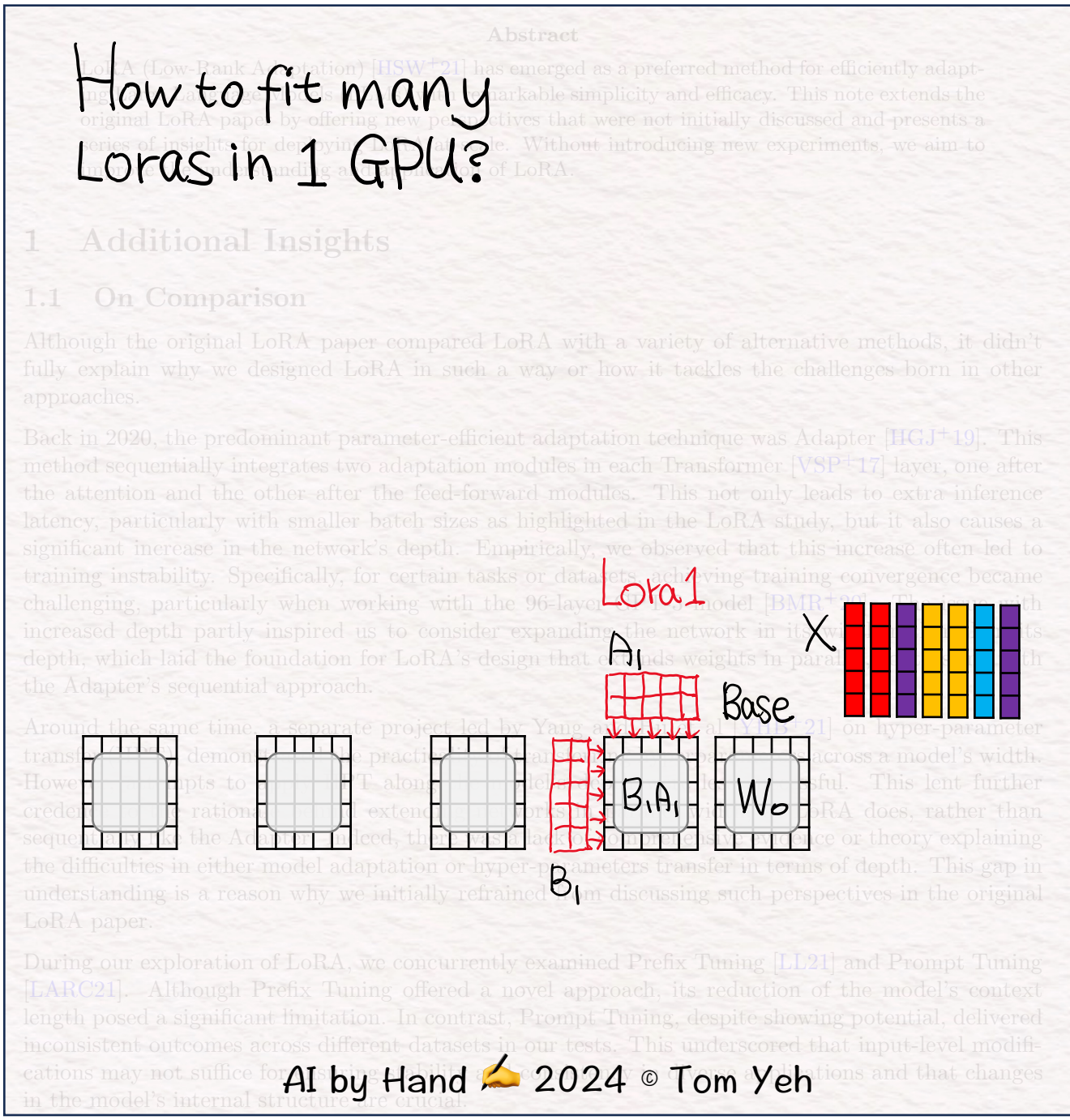
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



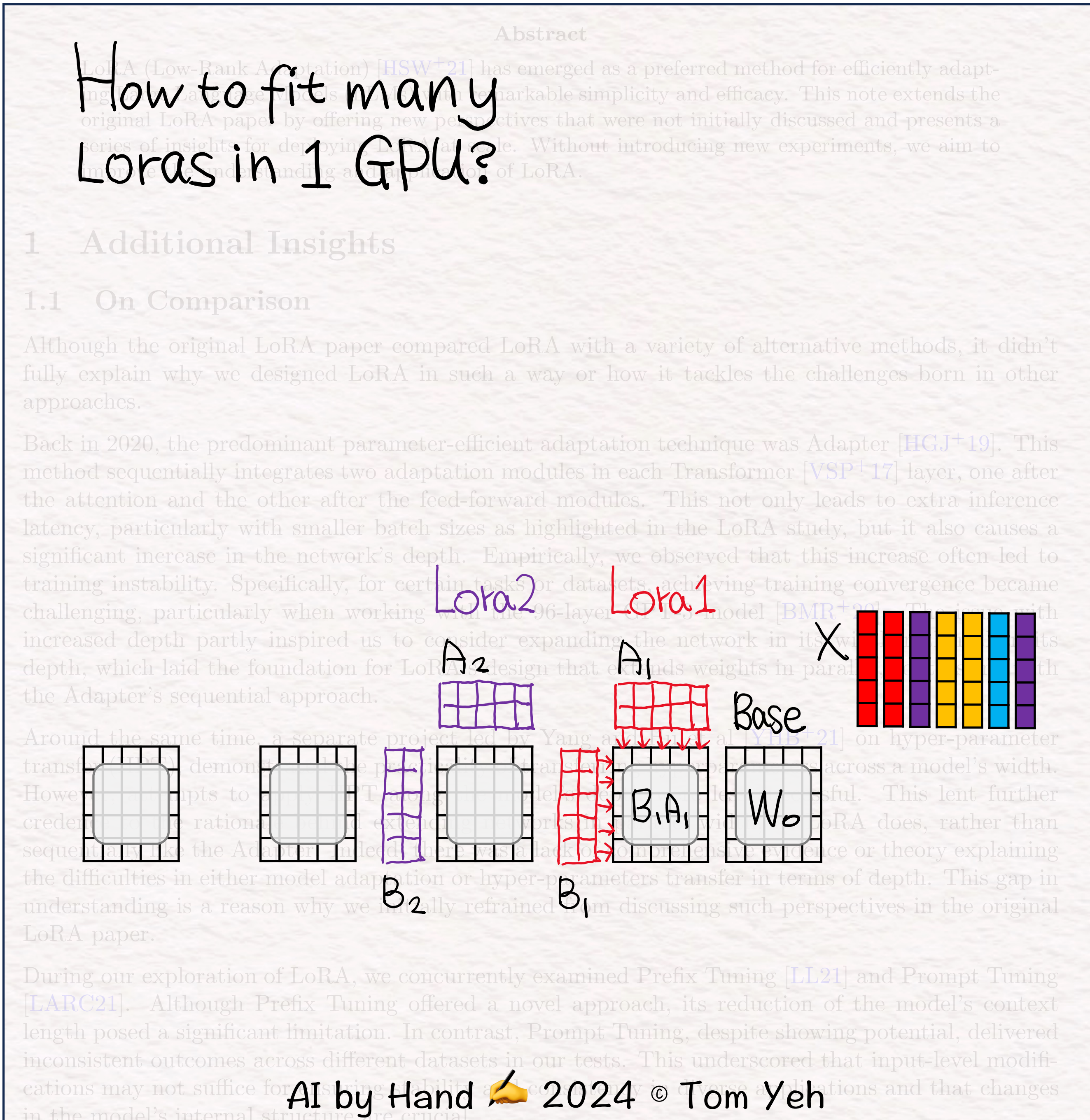
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



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

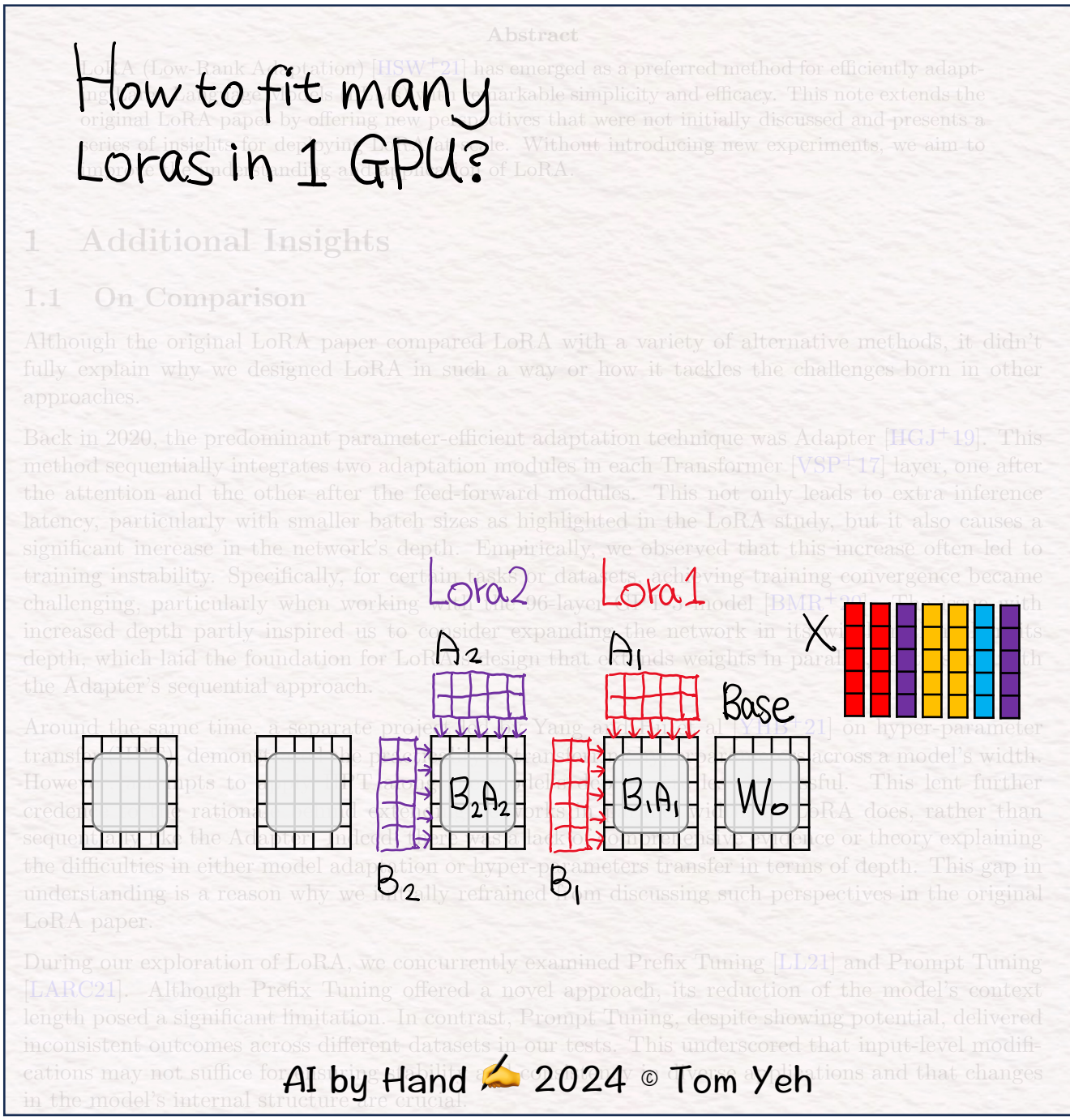
*work done while at Microsoft

†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



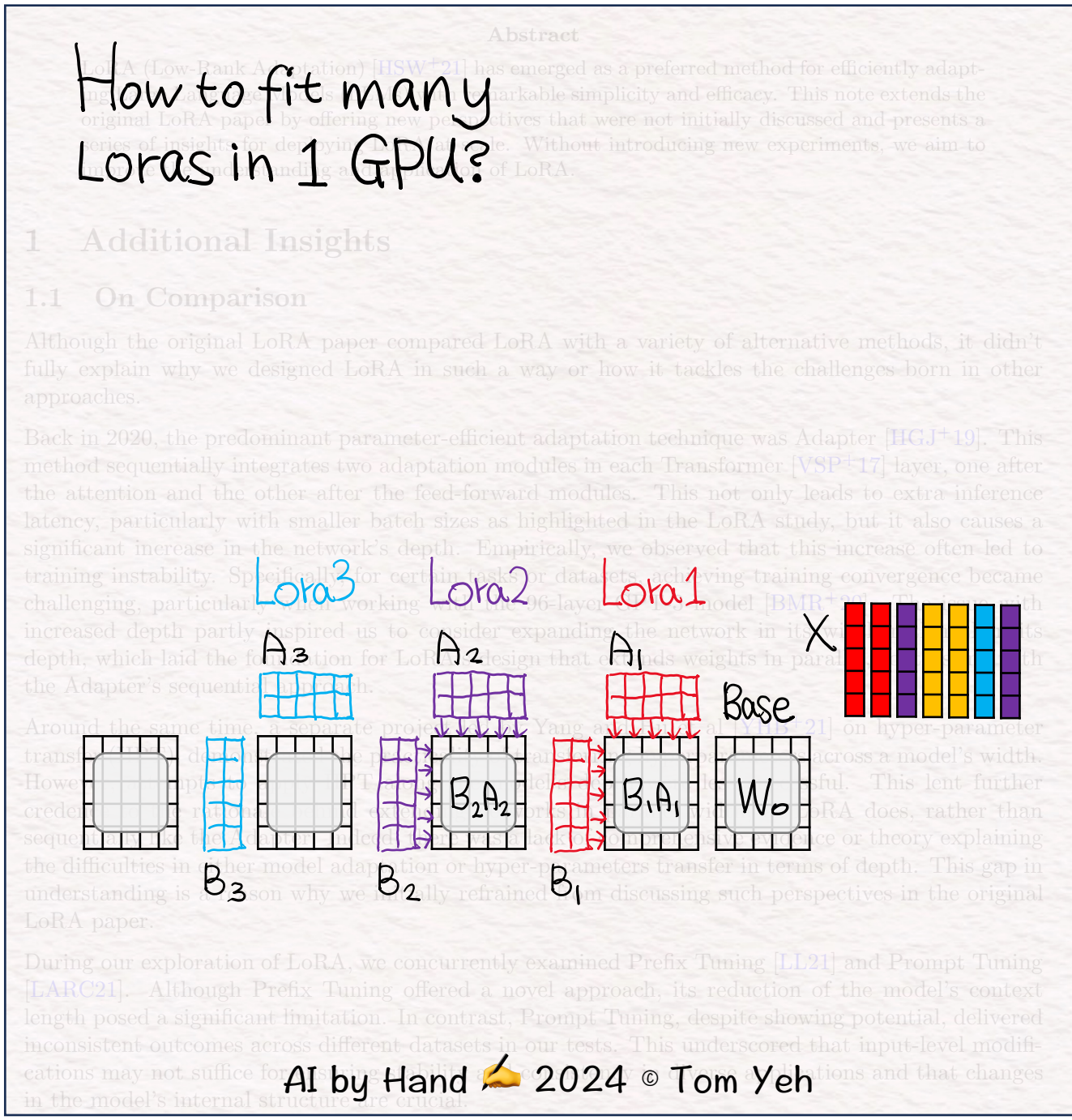
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



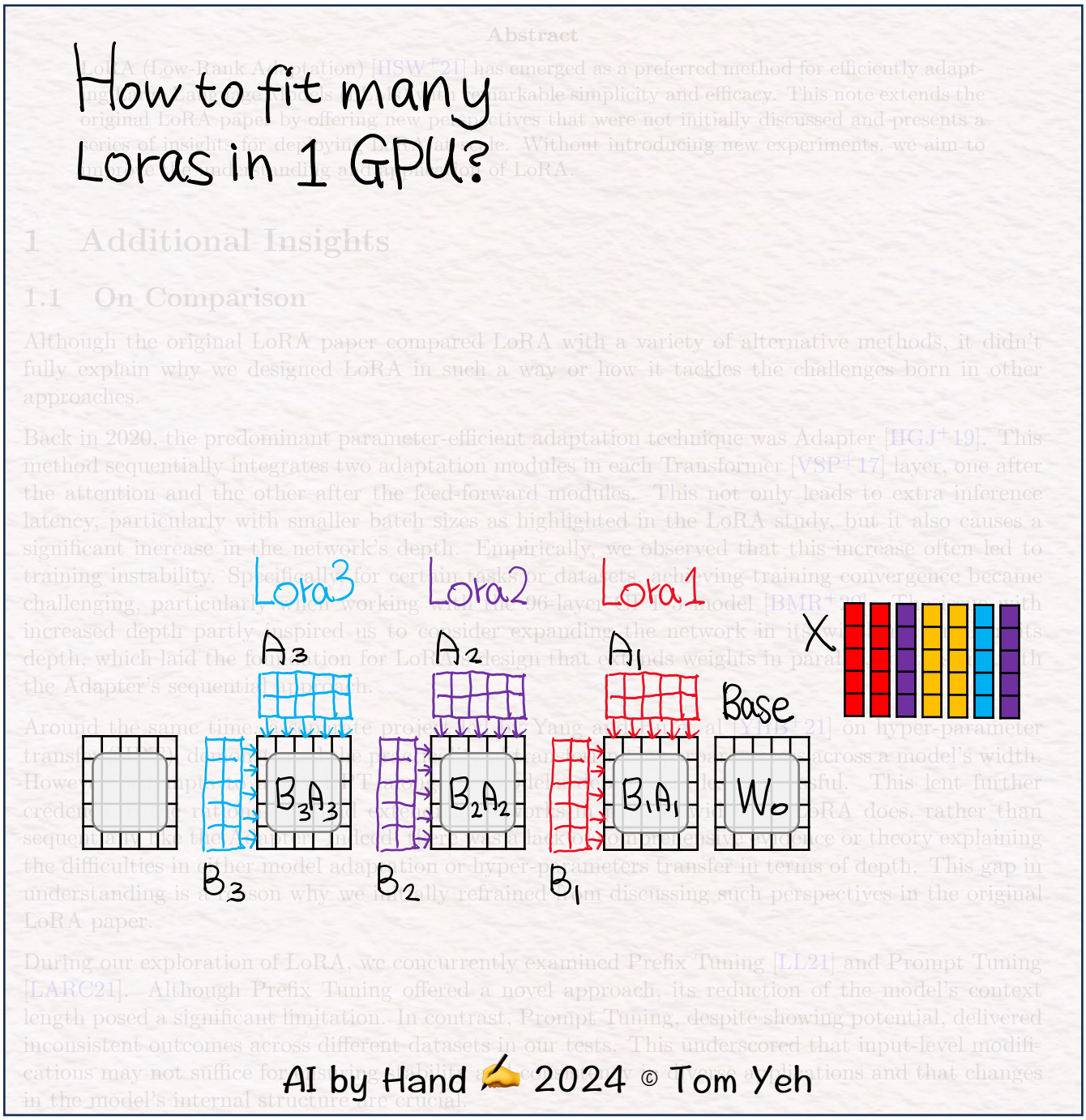
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



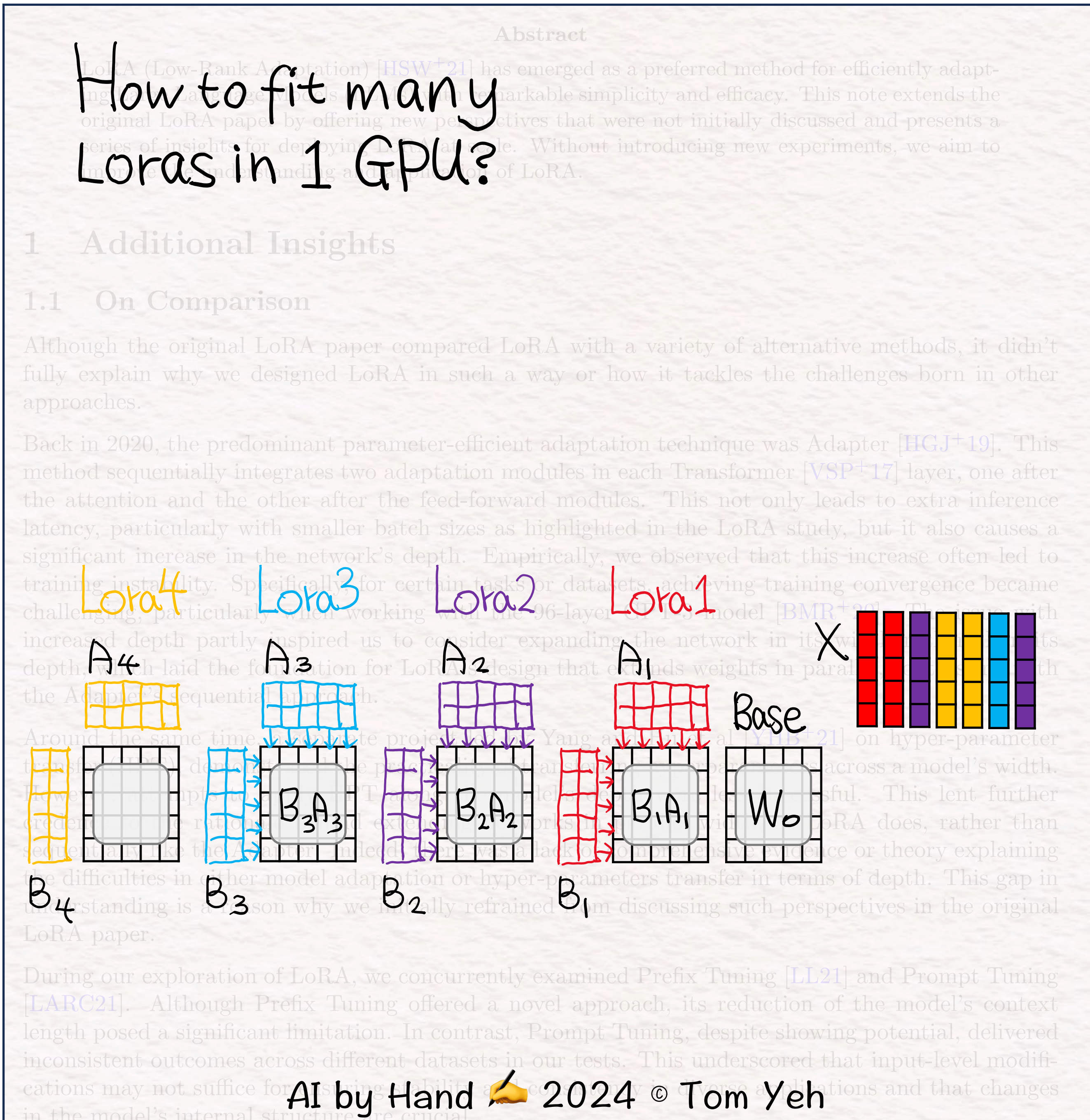
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



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

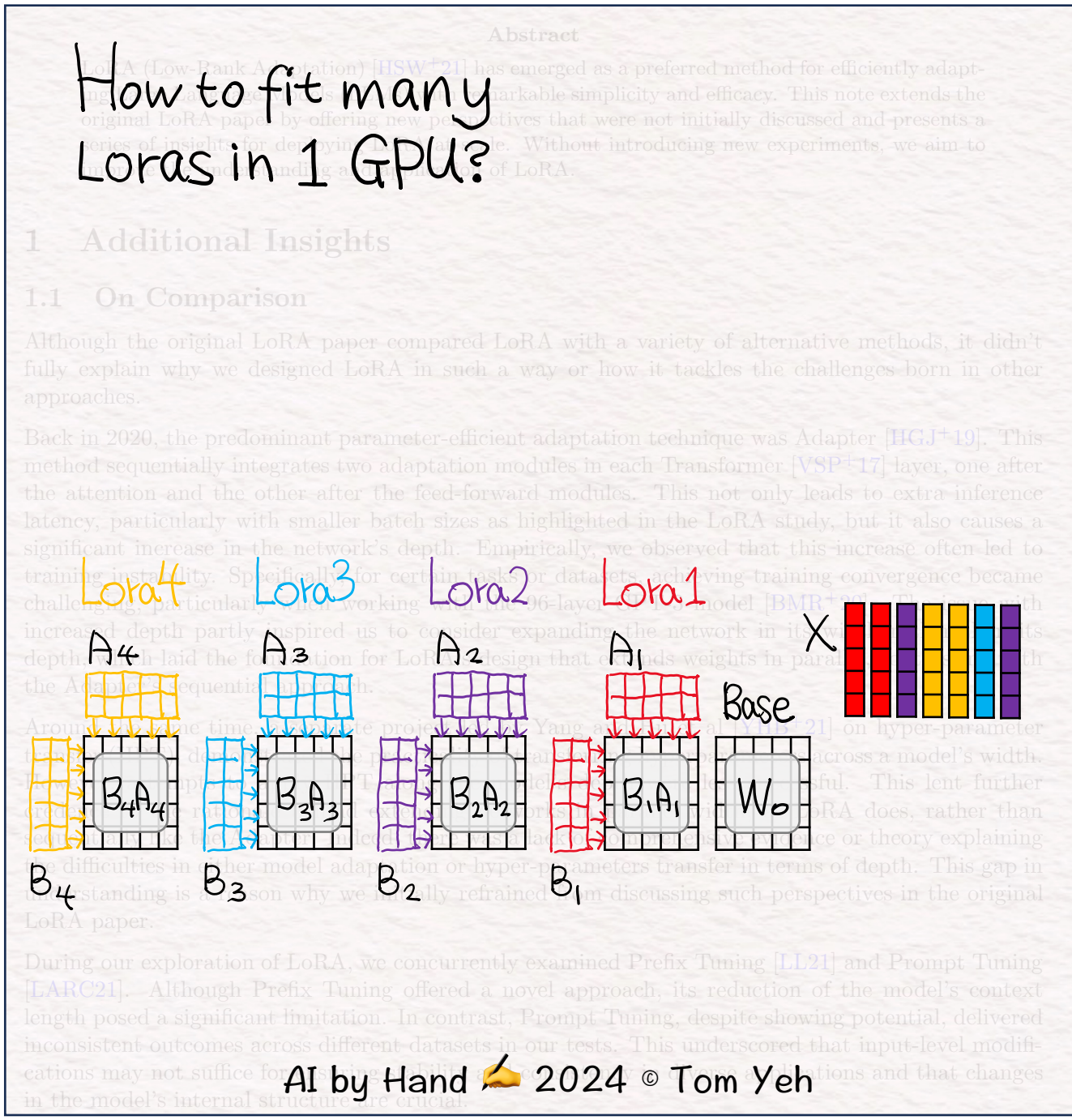
*work done while at Microsoft

†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



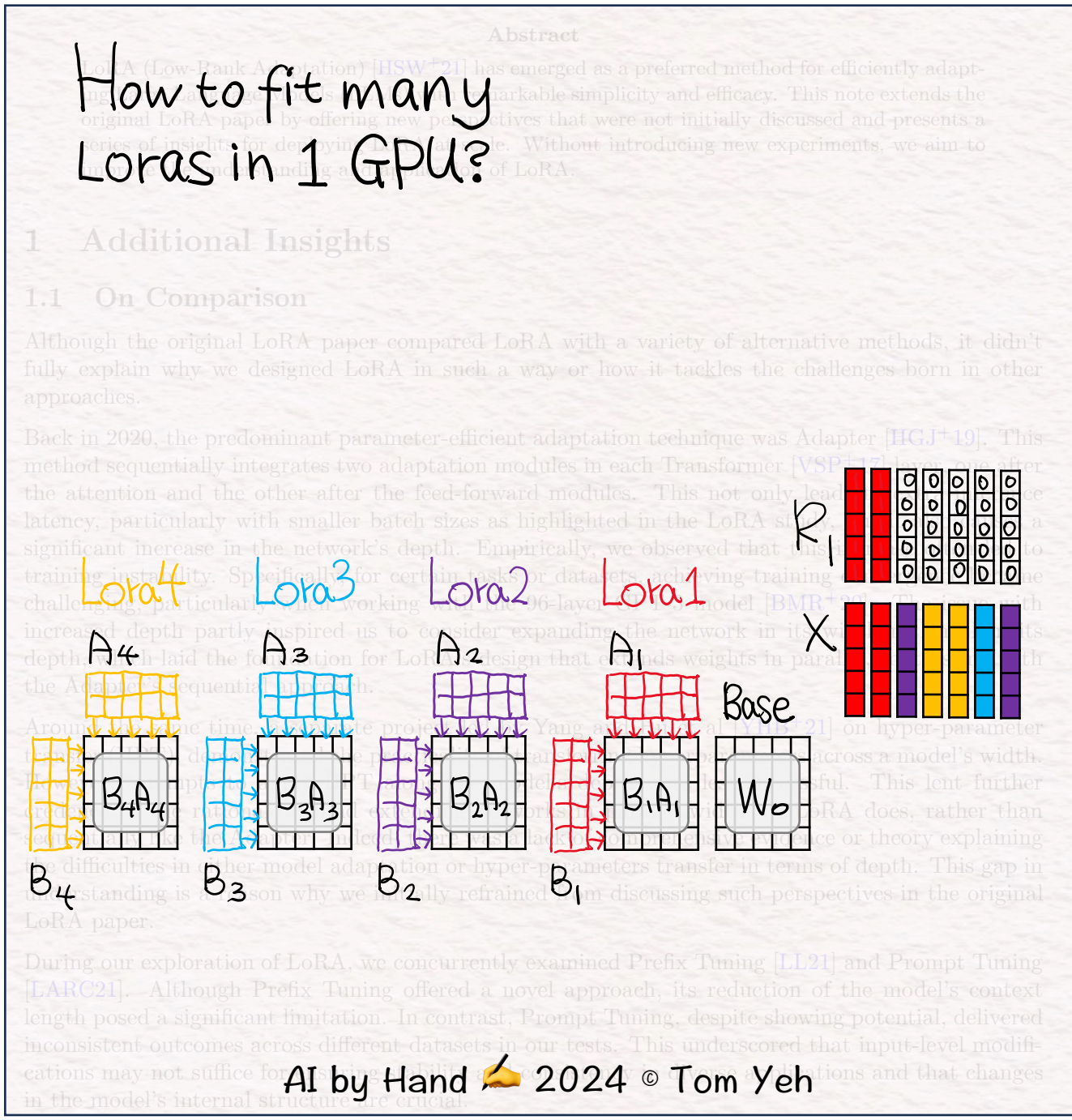
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



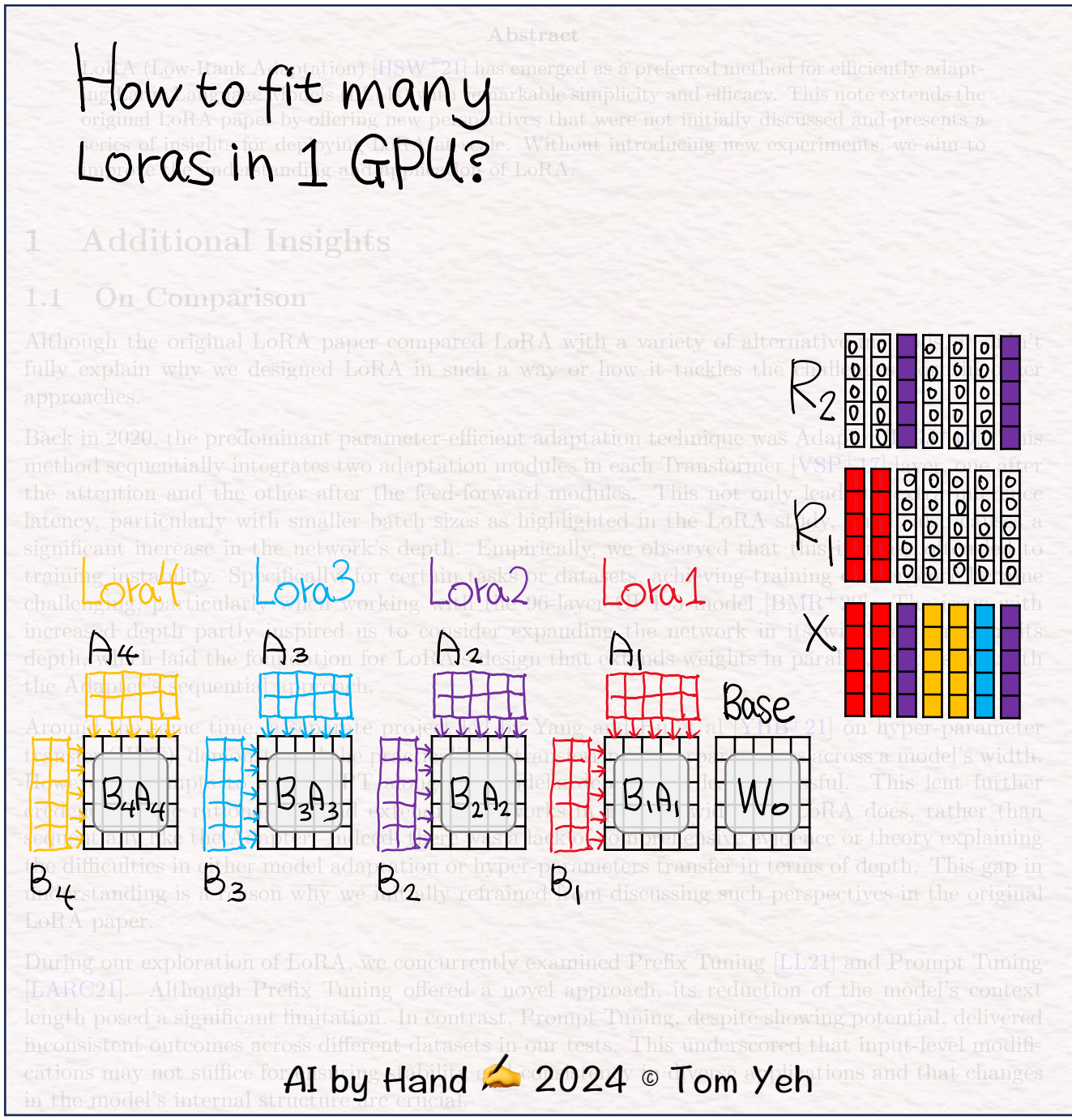
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



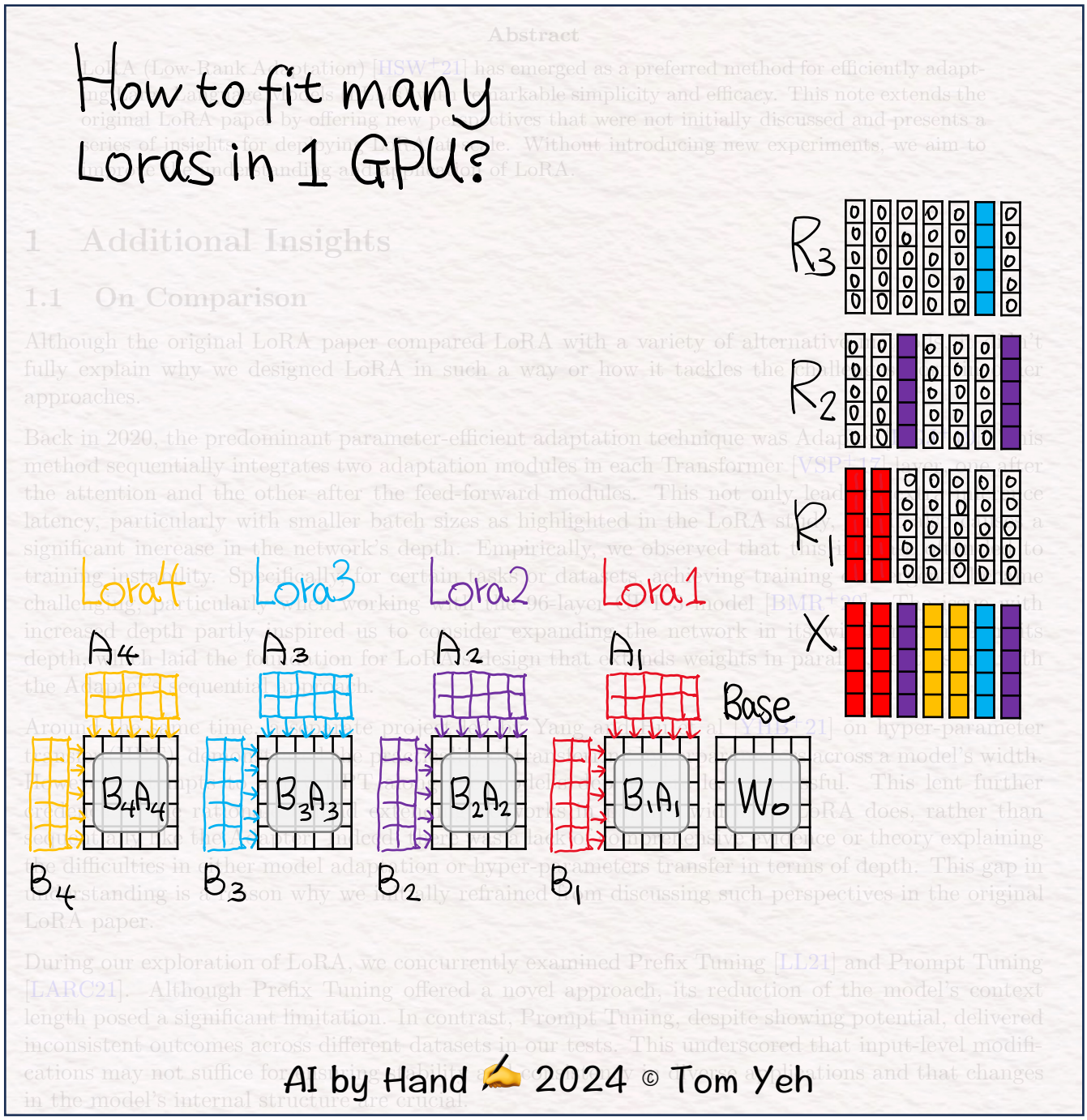
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



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

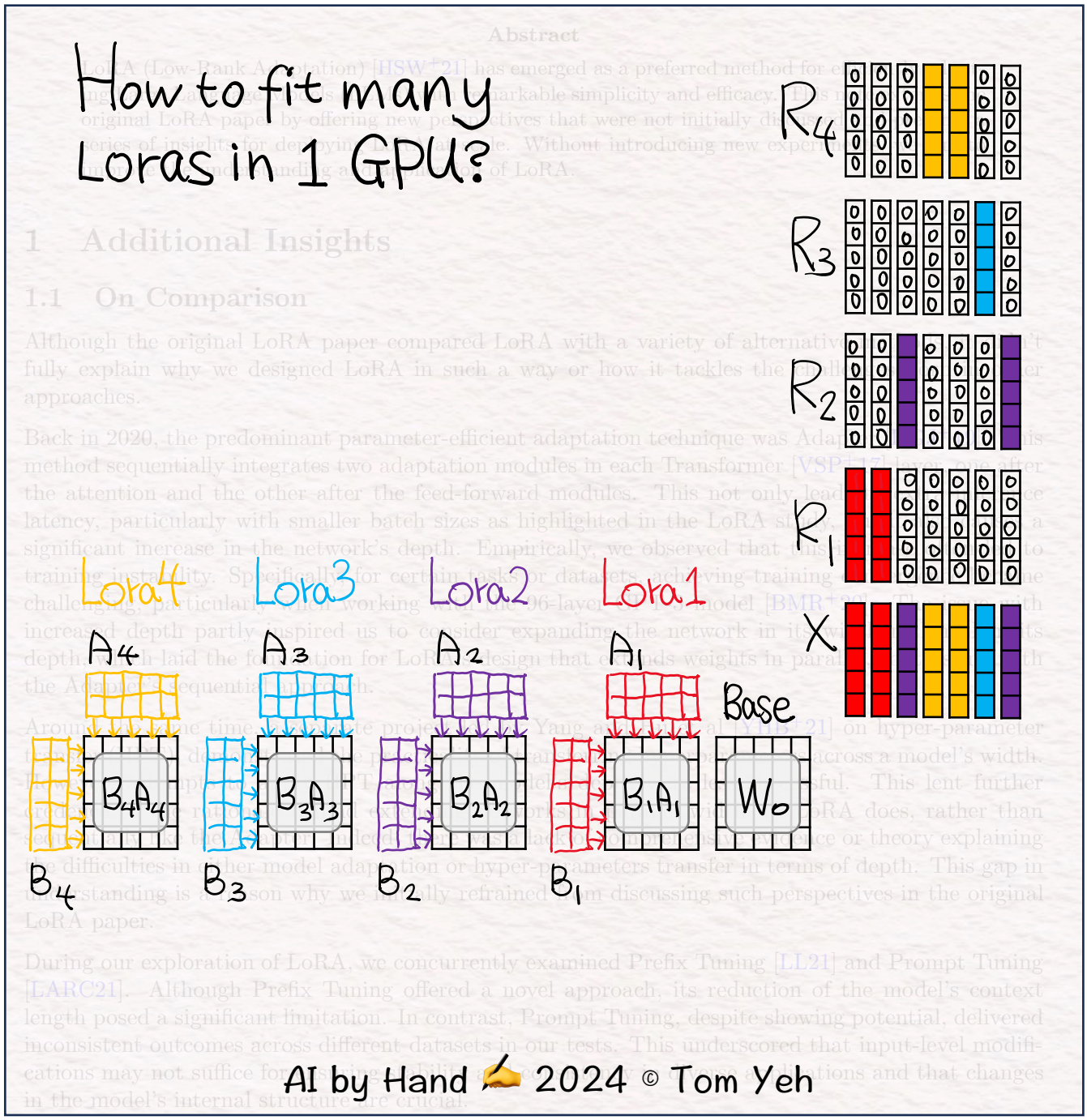
*work done while at Microsoft

†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



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

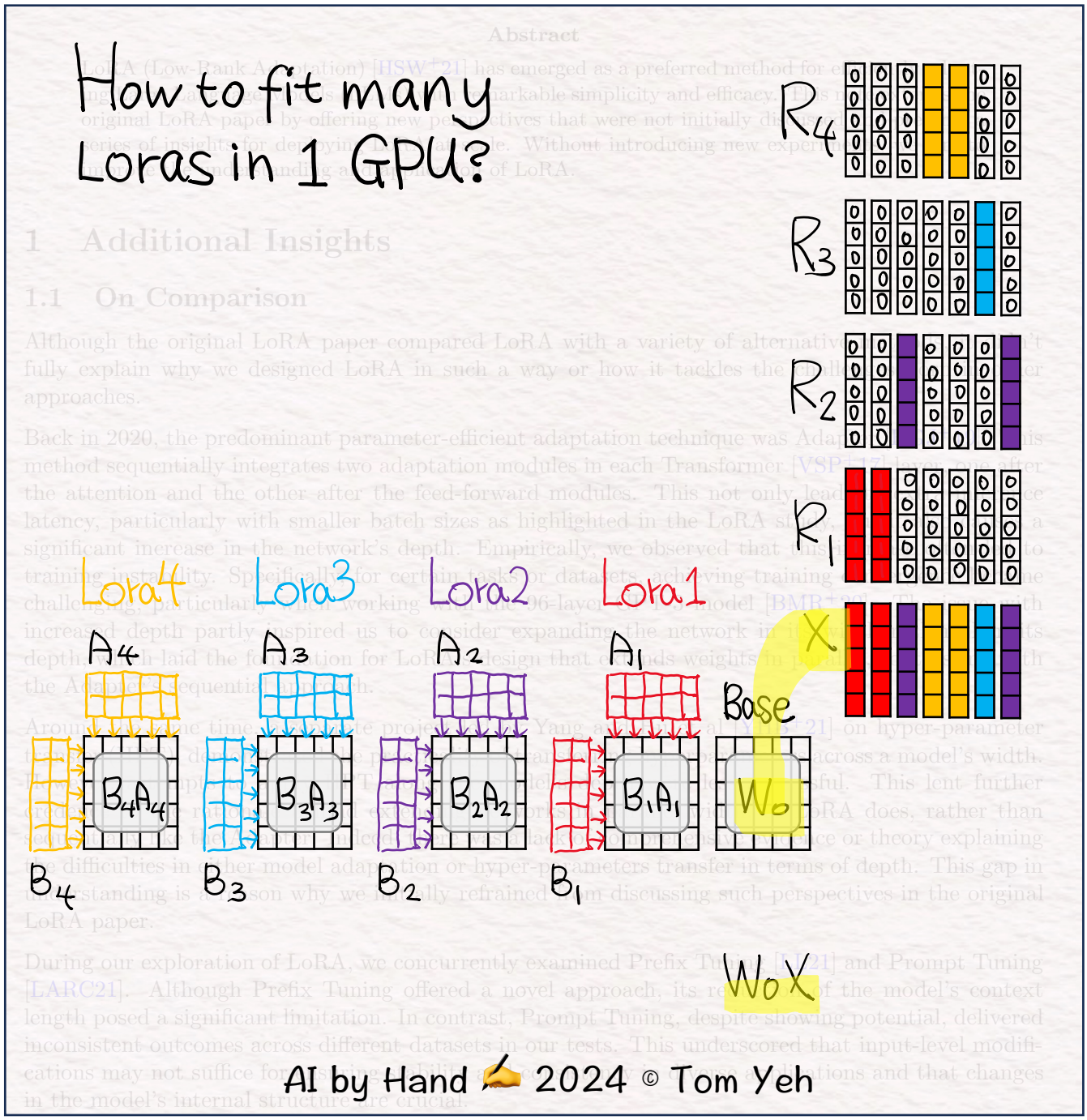
*work done while at Microsoft

†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



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

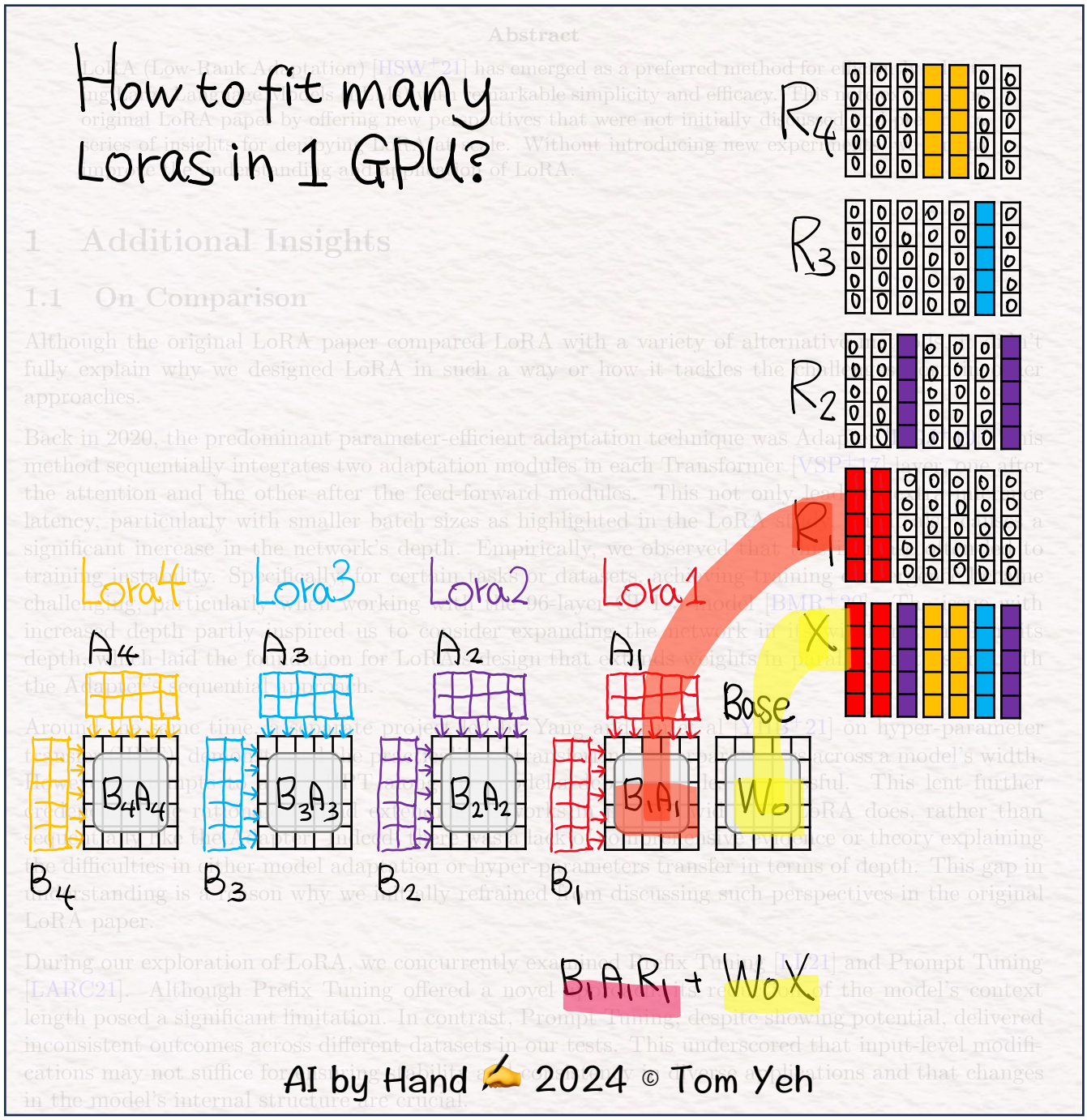
*work done while at Microsoft

†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



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

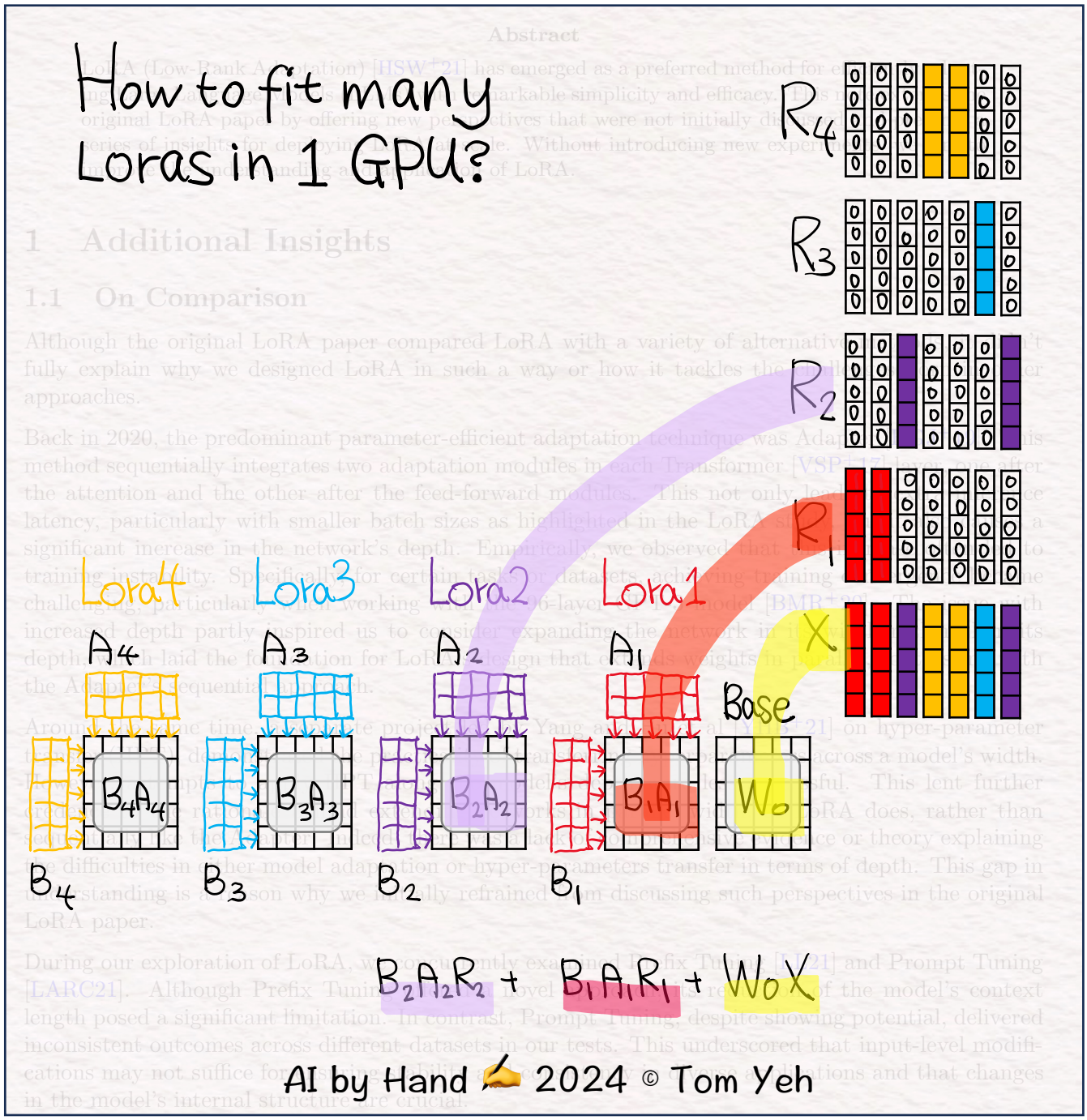
*work done while at Microsoft

†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



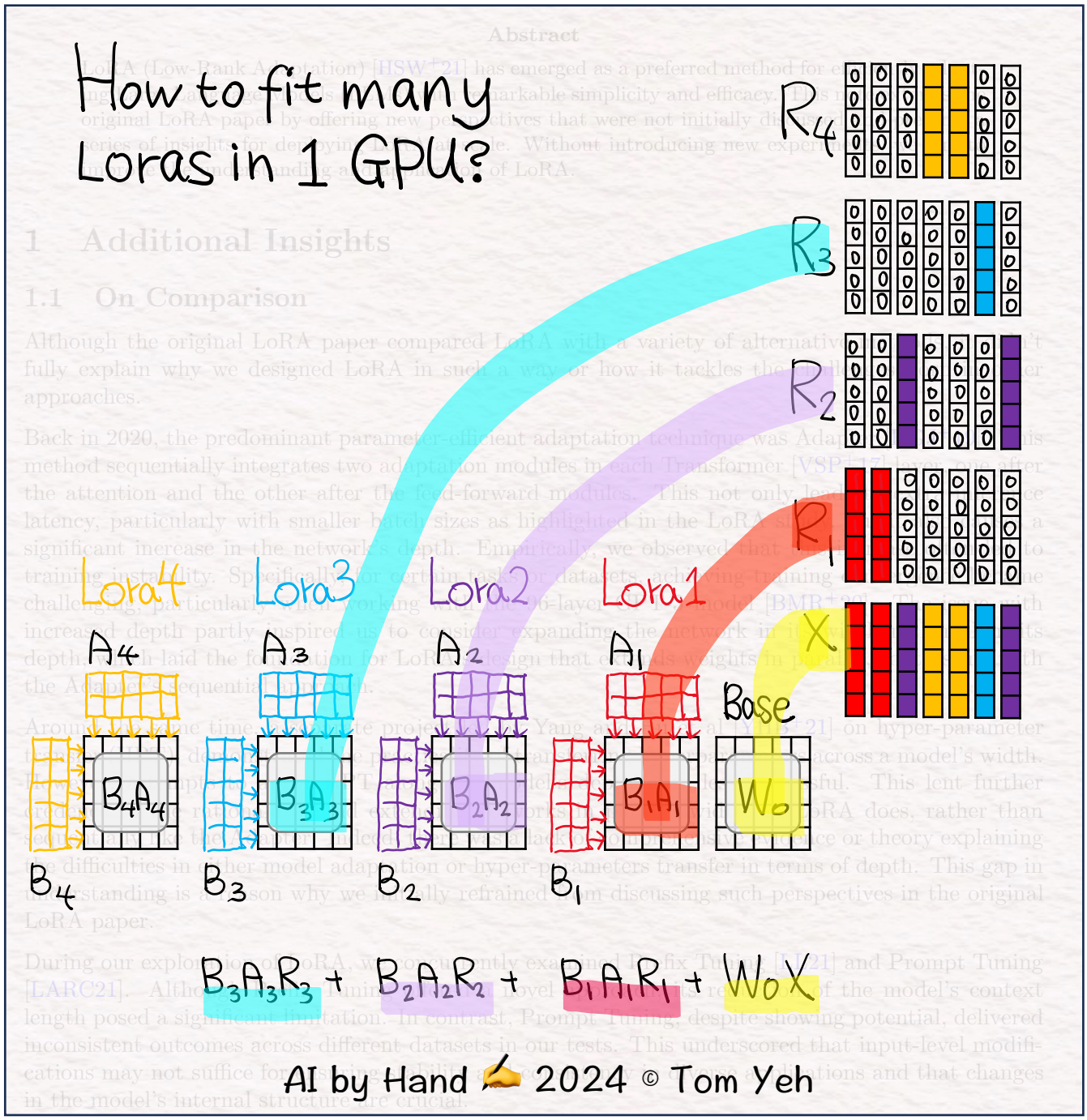
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



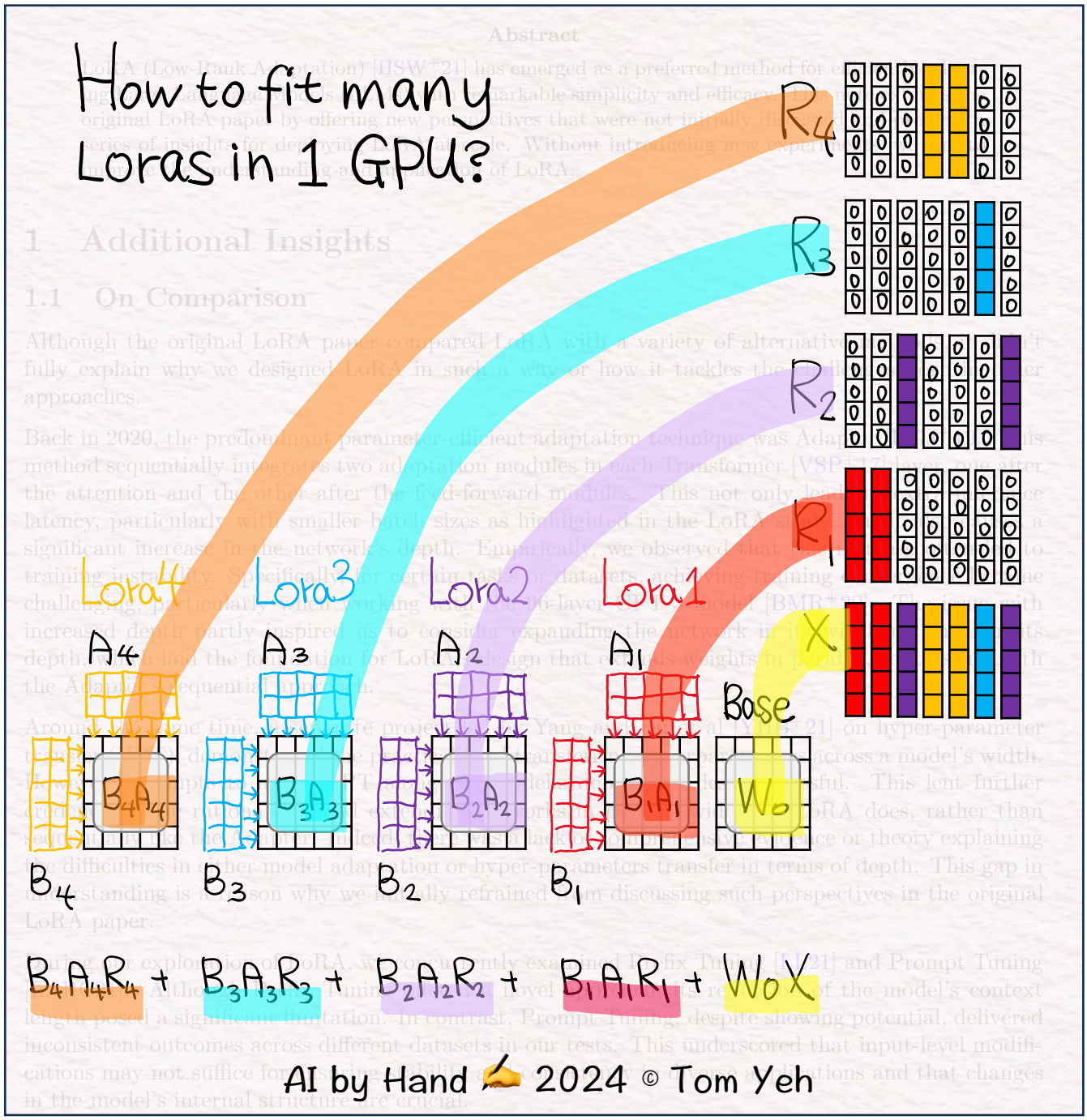
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



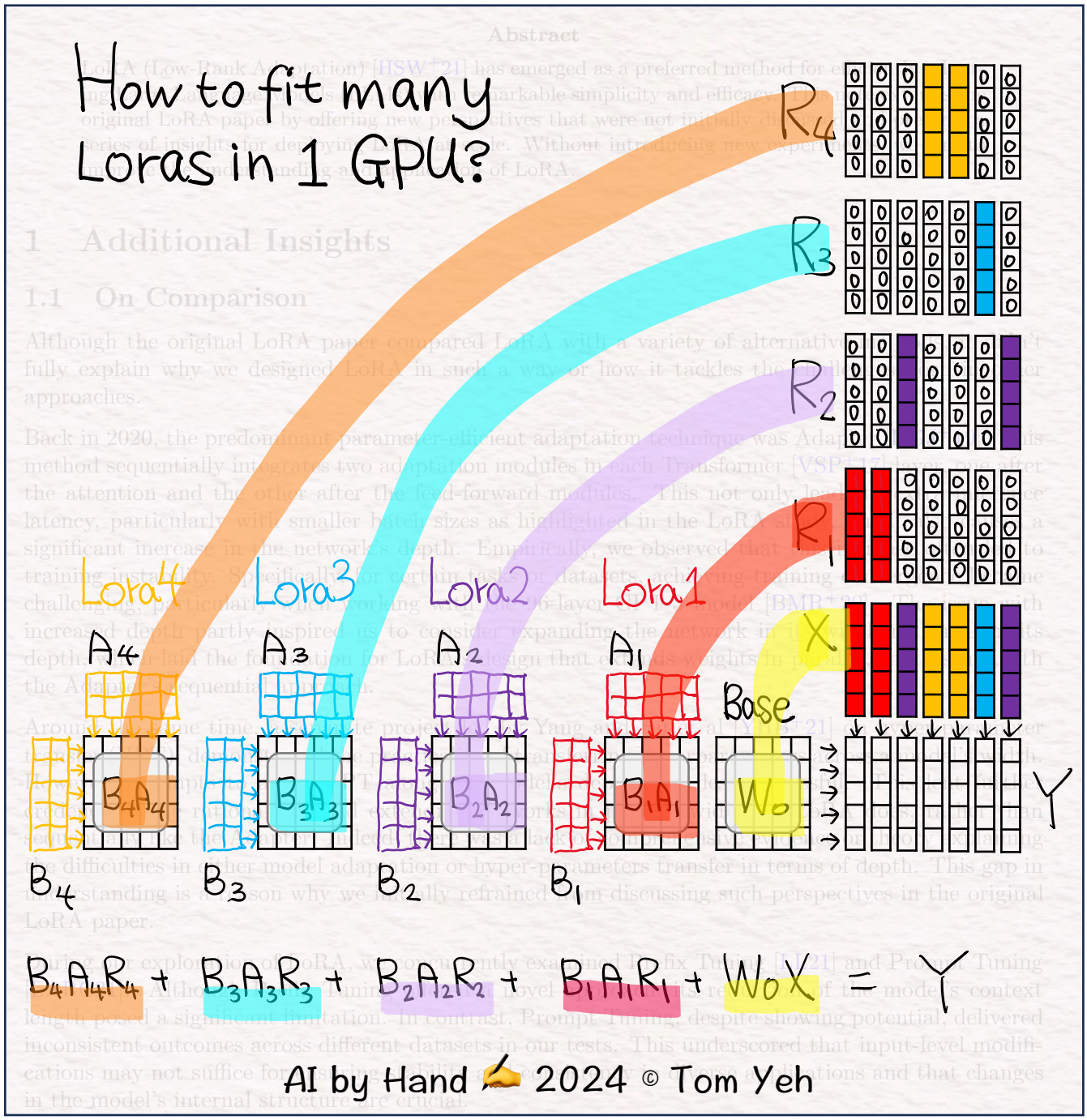
LoRA distinguishes itself by implementing adaptations at the matrix level, a more streamlined approach compared to the Adapter's addition of extra layers. This granular level of adaptation allows

*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



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

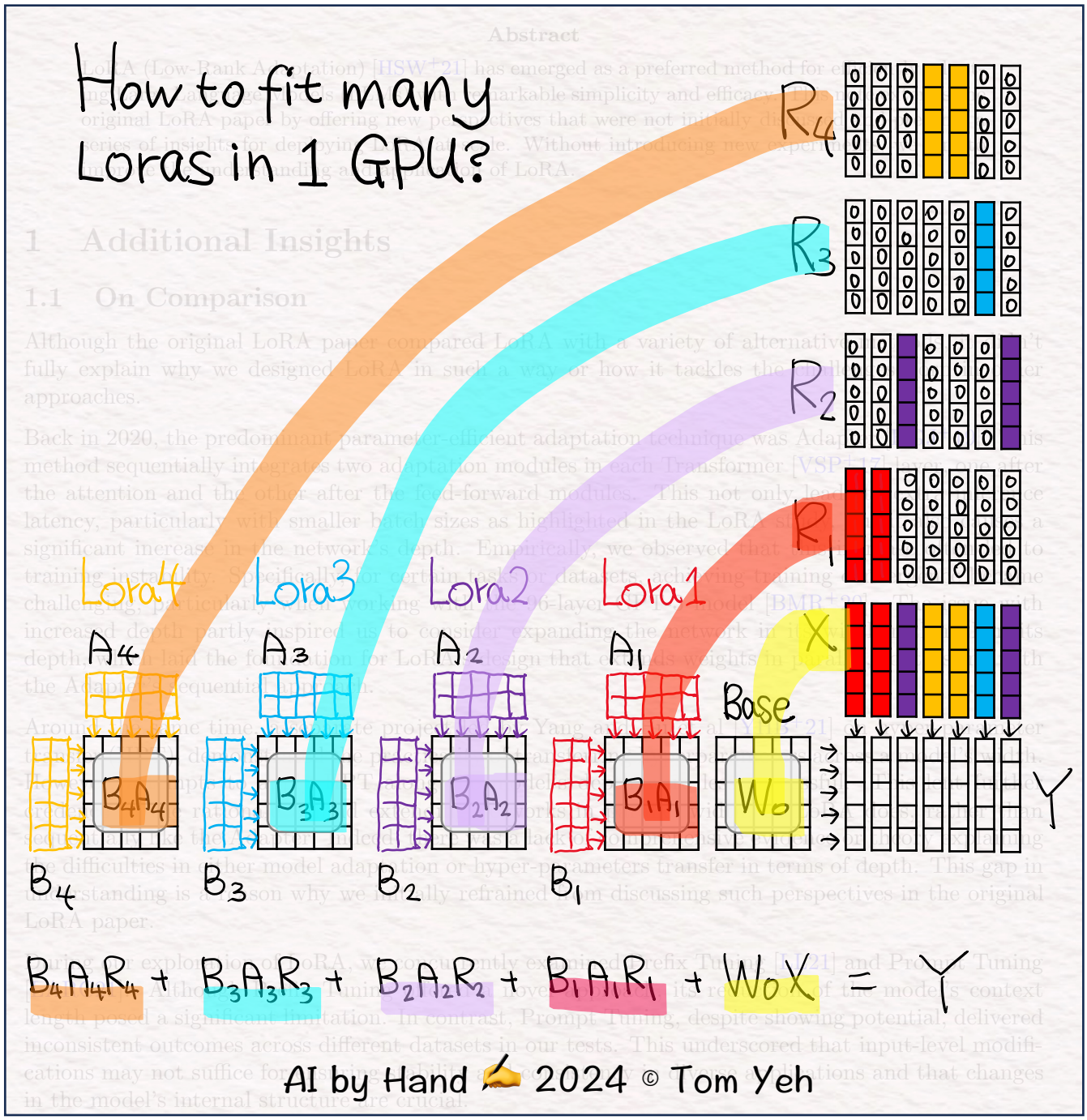
*work done while at Microsoft

†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



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