

Precise Verification of Transformers through ReLU-Catalyzed Abstraction Refinement

Hengjie Liu¹, Zhenya Zhang^{1,2}, Jianjun Zhao¹

¹Kyushu University, Fukuoka, Japan

²National Institute of Informatics, Tokyo, Japan

Lisbon, Portugal | September 25, 2026



KYUSHU UNIVERSITY

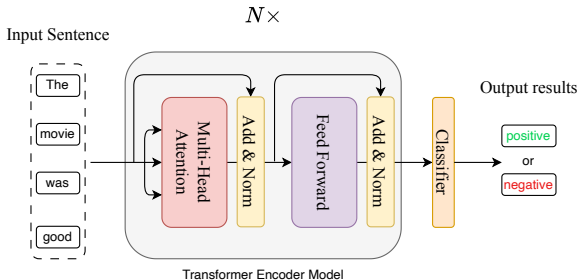
Introduction

Preliminary

Methodology

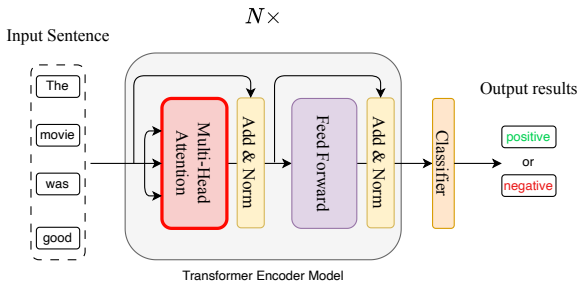
Experiments

What Is a Transformer?



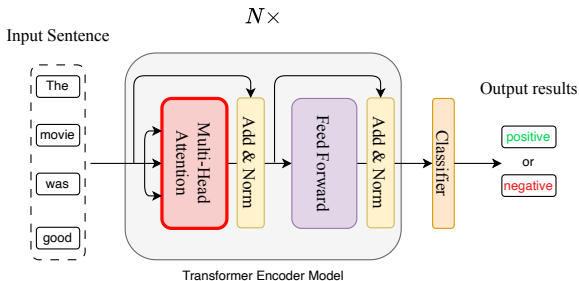
- A Transformer is a **complex neural network architecture** for sequence data (Vaswani et al., 2017).

What Is a Transformer?



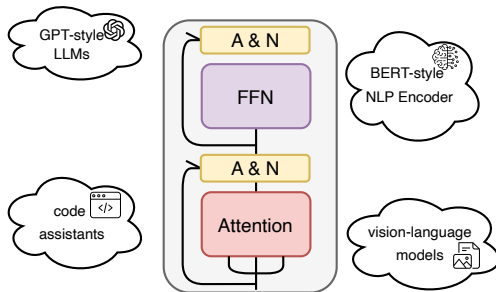
- ▶ A Transformer is a **complex neural network architecture** for sequence data (Vaswani et al., 2017).
- ▶ Its core idea is **attention**: each token can use information from other tokens.

What Is a Transformer?



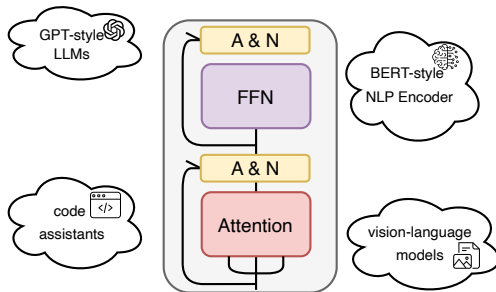
- ▶ A Transformer is a **complex neural network architecture** for sequence data (Vaswani et al., 2017).
- ▶ Its core idea is **attention**: each token can use information from other tokens.
- ▶ The model learns contextual representations, then uses them for prediction.

Why Transformers Matter



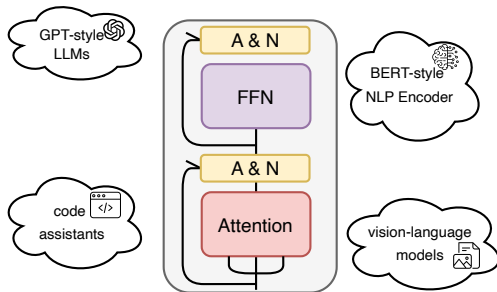
- Transformers changed machine translation, language understanding, and modern LLM applications.

Why Transformers Matter



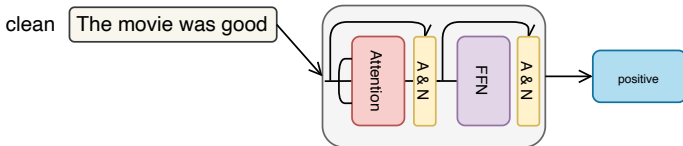
- ▶ Transformers changed machine translation, language understanding, and modern LLM applications.
- ▶ They now appear in everyday tools: chat assistants, search, summarization, code, and multimodal systems.

Why Transformers Matter



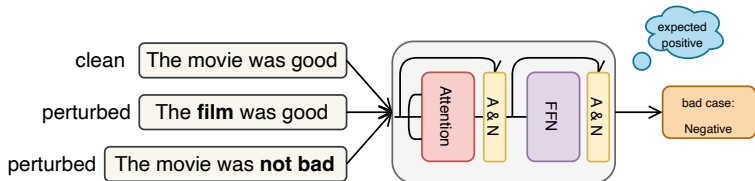
- ▶ Transformers changed machine translation, language understanding, and modern LLM applications.
- ▶ They now appear in everyday tools: chat assistants, search, summarization, code, and multimodal systems.
- ▶ As these models move into real systems, **reliability becomes a trust question.**

What Can Go Wrong?



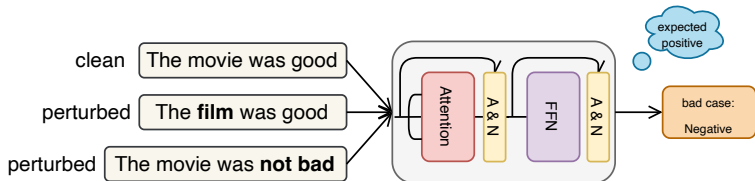
- A clean sentence can receive the expected prediction.

What Can Go Wrong?



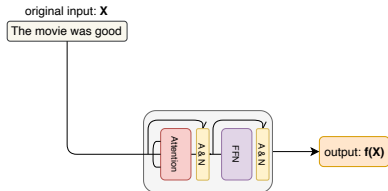
- A clean sentence can receive the expected prediction.
- Meaning-preserving perturbations can still flip that prediction.

What Can Go Wrong?



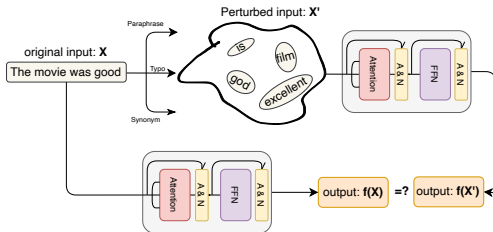
- ▶ A clean sentence can receive the expected prediction.
- ▶ Meaning-preserving perturbations can still flip that prediction.
- ▶ We want to **formally prove** that the prediction is unchanged for every permitted perturbation.

What Should Verification Prove?



- We provide a **formal guarantee** over the entire set.

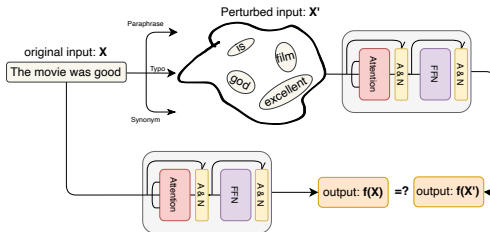
What Should Verification Prove?



Specification: $\forall X' \in \text{Perturbed}(X), \quad f(X') = f(X).$

- We provide a **formal guarantee** over the entire set.
- Every specified input X' must preserve the original prediction.

What Should Verification Prove?



Specification: $\forall X' \in \text{Perturbed}(X), \quad f(X') = f(X).$

- ▶ We provide a **formal guarantee** over the entire set.
- ▶ Every specified input X' must preserve the original prediction.
- ▶ The proof covers the whole set without enumerating its members.

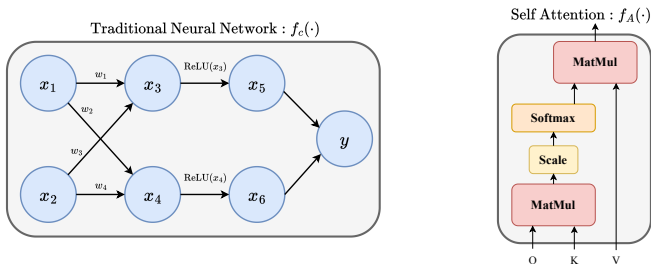
Introduction

Preliminary

Methodology

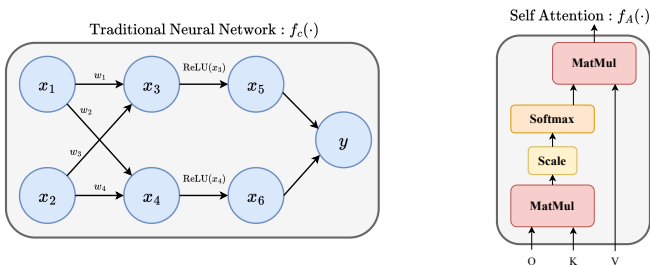
Experiments

What Changes from Traditional Networks?



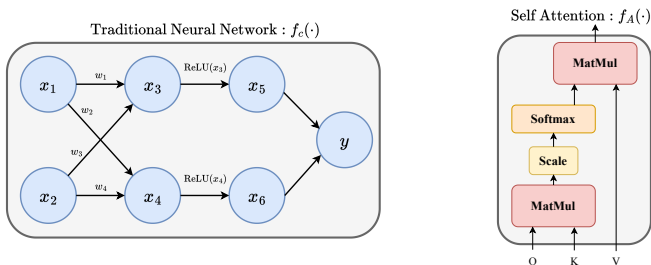
- In a traditional network, the main nonlinear step is usually an activation such as $h = \text{ReLU}(Wx + b)$.

What Changes from Traditional Networks?



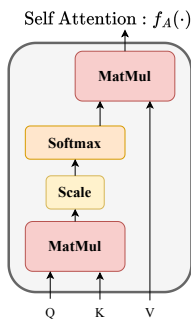
- In a traditional network, the main nonlinear step is usually an activation such as $h = \text{ReLU}(Wx + b)$.
- Transformer attention adds an extra MatMul: $S = QK^\top$.

What Changes from Traditional Networks?



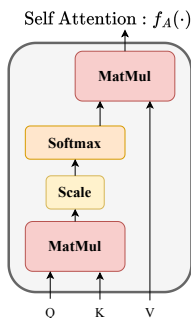
- In a traditional network, the main nonlinear step is usually an activation such as $h = \text{ReLU}(Wx + b)$.
- Transformer attention adds an extra MatMul: $S = QK^\top$.
- Each score contains products of two changing values, so it is a **bilinear nonlinearity** and harder to bound tightly.

Attention Creates Token Interactions



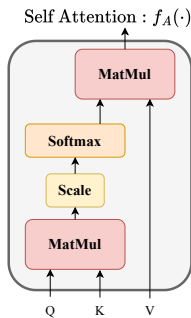
- The first MatMul forms attention scores: $S = QK^\top$.

Attention Creates Token Interactions



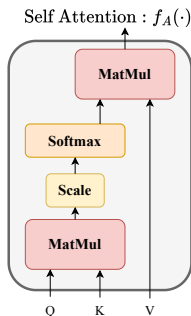
- The first MatMul forms attention scores: $S = QK^\top$.
- S_{ij} decides how strongly token i attends to token j .

Attention Creates Token Interactions



- ▶ The first MatMul forms attention scores: $S = QK^\top$.
- ▶ S_{ij} decides how strongly token i attends to token j .
- ▶ Because Q and K both depend on the perturbed input, these scores become uncertain.

Attention Creates Token Interactions



- ▶ The first MatMul forms attention scores: $S = QK^\top$.
- ▶ S_{ij} decides how strongly token i attends to token j .
- ▶ Because Q and K both depend on the perturbed input, these scores become uncertain.
- ▶ Verification must bound the scores before Softmax and value mixing.

The Bottleneck Hides in One Attention Score

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{i1} & q_{i2} & q_{i3} \\ q_{n1} & q_{n2} & q_{n3} \end{bmatrix} \quad K^T = \begin{bmatrix} k_{11} & k_{1j} & k_{1m} \\ k_{21} & k_{2j} & k_{2m} \\ k_{31} & k_{3j} & k_{3m} \end{bmatrix} \Rightarrow S = QK^T$$

$$S_{ij} = \sum_{h=1}^d q_{ih} k_{hj}$$

- The task decomposes from a matrix product, to one attention score, to many scalar products.

The Bottleneck Hides in One Attention Score

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{i1} & q_{i2} & q_{i3} \\ q_{n1} & q_{n2} & q_{n3} \end{bmatrix} \quad K^T = \begin{bmatrix} k_{11} & k_{1j} & k_{1m} \\ k_{21} & k_{2j} & k_{2m} \\ k_{31} & k_{3j} & k_{3m} \end{bmatrix} \quad \Rightarrow \quad S = QK^T$$

$$S_{ij} = \sum_{h=1}^d q_{ih} k_{hj}$$

- The task decomposes from a matrix product, to one attention score, to many scalar products.
- Each scalar term is $z = xy$, where both x and y may be uncertain.

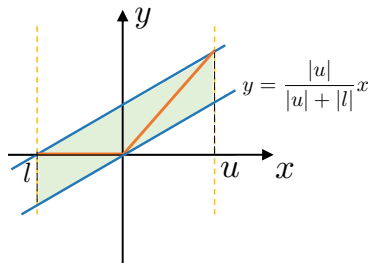
The Bottleneck Hides in One Attention Score

$$Q = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{i1} & q_{i2} & q_{i3} \\ q_{n1} & q_{n2} & q_{n3} \end{bmatrix} \quad K^T = \begin{bmatrix} k_{11} & k_{1j} & k_{1m} \\ k_{21} & k_{2j} & k_{2m} \\ k_{31} & k_{3j} & k_{3m} \end{bmatrix} \Rightarrow S = QK^T$$

$$S_{ij} = \sum_{h=1}^d q_{ih} k_{hj}$$

- ▶ The task decomposes from a matrix product, to one attention score, to many scalar products.
- ▶ Each scalar term is $z = xy$, where both x and y may be uncertain.
- ▶ The core subproblem is therefore: find sound lower and upper bounds for multiplying two numbers.

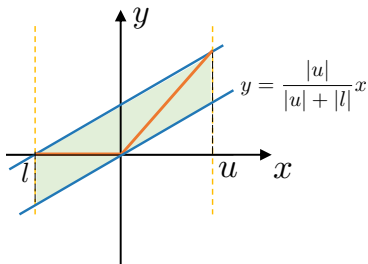
Why Linear Relaxation Matters



Linear relaxation of ReLU

- ReLU is nonlinear, but over $x \in [l, u]$ it can be enclosed by sound linear bounds.

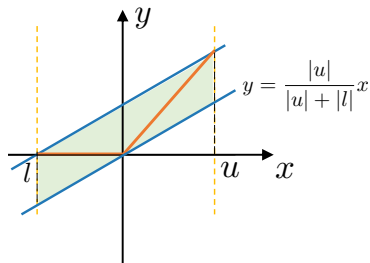
Why Linear Relaxation Matters



Linear relaxation of ReLU

- ReLU is nonlinear, but over $x \in [l, u]$ it can be enclosed by sound linear bounds.
- This **linear relaxation** replaces exact nonlinear behavior with a tractable over-approximation (Zhang et al., 2018).

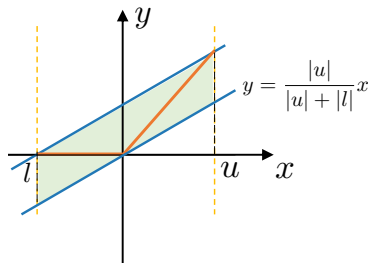
Why Linear Relaxation Matters



Linear relaxation of ReLU

- ▶ ReLU is nonlinear, but over $x \in [l, u]$ it can be enclosed by sound linear bounds.
- ▶ This **linear relaxation** replaces exact nonlinear behavior with a tractable over-approximation (Zhang et al., 2018).
- ▶ Linear bounds propagate quickly through later matrix multiplications and additions.

Why Linear Relaxation Matters



Linear relaxation of ReLU

- ▶ ReLU is nonlinear, but over $x \in [l, u]$ it can be enclosed by sound linear bounds.
- ▶ This **linear relaxation** replaces exact nonlinear behavior with a tractable over-approximation (Zhang et al., 2018).
- ▶ Linear bounds propagate quickly through later matrix multiplications and additions.
- ▶ We want the same advantage for $z = xy$, but both factors depend on the perturbed input and are harder to bound tightly.

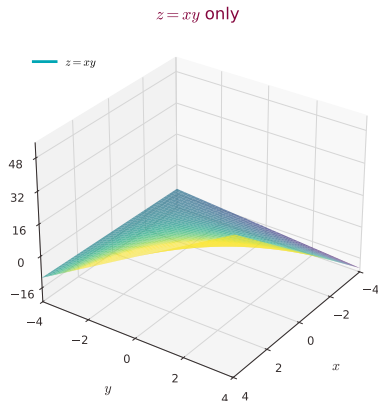
Introduction

Preliminary

Methodology

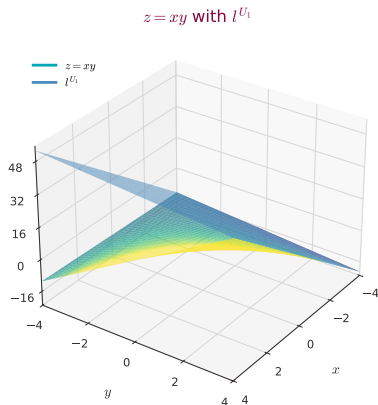
Experiments

Why One Sound Plane Is Not Enough



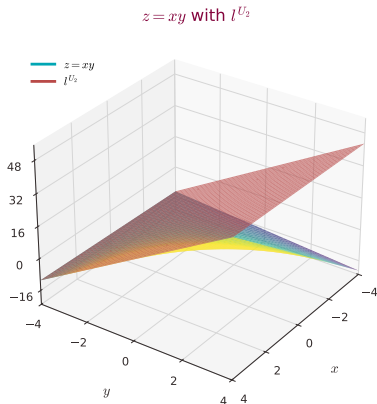
- The scalar product forms the bilinear surface $z = xy$.

Why One Sound Plane Is Not Enough



- ▶ The scalar product forms the bilinear surface $z = xy$.
- ▶ Prior Transformer verifiers use one sound upper plane, l^{U_1} (Shi et al., 2020).

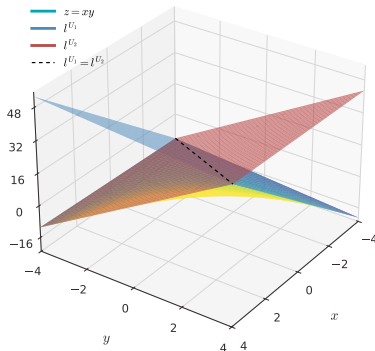
Why One Sound Plane Is Not Enough



- ▶ The scalar product forms the bilinear surface $z = xy$.
- ▶ Prior Transformer verifiers use one sound upper plane, l^{U_1} (Shi et al., 2020).
- ▶ We introduce the dual plane l^{U_2} , which is also a sound upper bound.

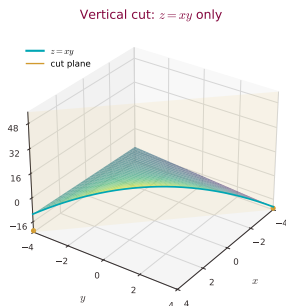
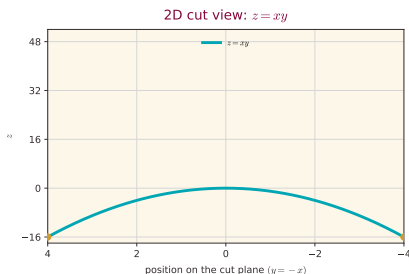
Why One Sound Plane Is Not Enough

Two upper planes intersect along $l^{U_1} = l^{U_2}$



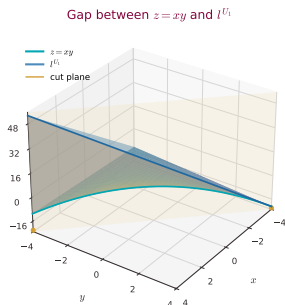
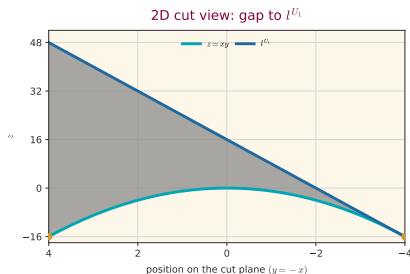
- ▶ The scalar product forms the bilinear surface $z = xy$.
- ▶ Prior Transformer verifiers use one sound upper plane, l^{U_1} (Shi et al., 2020).
- ▶ We introduce the dual plane l^{U_2} , which is also a sound upper bound.
- ▶ The two planes intersect, and **each is tighter in a different region**.

The Ideal Move: Use the Tighter Plane



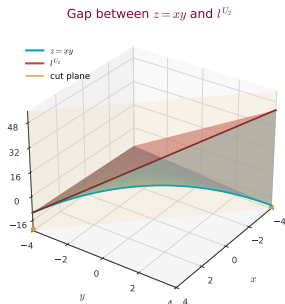
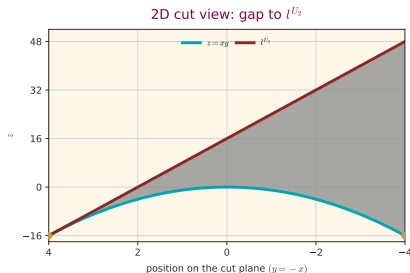
- The yellow plane slices the 3D surfaces; the right panel is its front view.

The Ideal Move: Use the Tighter Plane



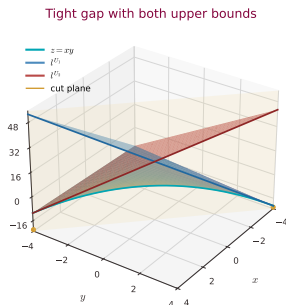
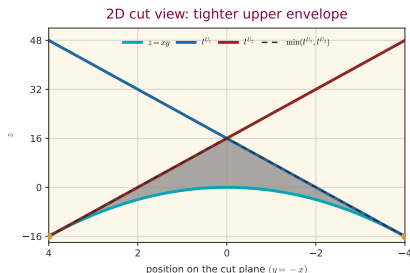
- ▶ The yellow plane slices the 3D surfaces; the right panel is its front view.
- ▶ l^{U_1} is tight on one side and loose on the other.

The Ideal Move: Use the Tighter Plane



- ▶ The yellow plane slices the 3D surfaces; the right panel is its front view.
- ▶ l^{U_1} is tight on one side and loose on the other.
- ▶ l^{U_2} is tight on the complementary side.

The Ideal Move: Use the Tighter Plane



- ▶ The yellow plane slices the 3D surfaces; the right panel is its front view.
- ▶ l^{U_1} is tight on one side and loose on the other.
- ▶ l^{U_2} is tight on the complementary side.
- ▶ Choose $l^* = \min(l^{U_1}, l^{U_2})$ for the **tighter upper envelope**.

The ReLU Trick: Make the Choice Linear Again I

The pointwise minimum is the ideal upper envelope, but it is no longer one plane.

The ReLU Trick: Make the Choice Linear Again I

The pointwise minimum is the ideal upper envelope, but it is no longer one plane.

Let $t = l^{U_1} - l^{U_2}$. Then

$$l^* = \min(l^{U_1}, l^{U_2}) = l^{U_1} - \text{ReLU}(t)$$

The ReLU gives an **exact representation** of the piecewise-linear choice.

The ReLU Trick: Make the Choice Linear Again I

The pointwise minimum is the ideal upper envelope, but it is no longer one plane.

Let $t = l^{U_1} - l^{U_2}$. Then

$$l^* = \min(l^{U_1}, l^{U_2}) = l^{U_1} - \text{ReLU}(t)$$

The ReLU gives an **exact representation** of the piecewise-linear choice.

Why ReLU? It lets us directly reuse mature, well-established abstraction methods developed for ReLU networks.

The ReLU Trick: Make the Choice Linear Again II

$$l^* = \min(l^{U_1}, l^{U_2}) = l^{U_1} - \text{ReLU}(t)$$

$$\boxed{\text{ReLU}(t) \geq \alpha t, \alpha \in [0, 1]}$$

 $\implies$

$$\boxed{l^* \leq l_\alpha^* = (1 - \alpha)l^{U_1} + \alpha l^{U_2}}$$

- The lower ReLU relaxation gives a sound upper plane.

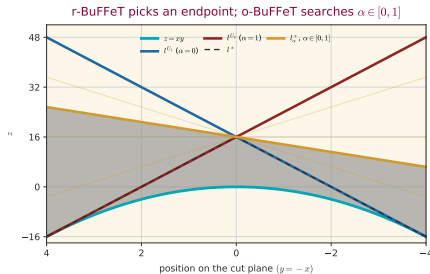
The ReLU Trick: Make the Choice Linear Again II

$$l^* = \min(l^{U_1}, l^{U_2}) = l^{U_1} - \text{ReLU}(t)$$

$$\boxed{\text{ReLU}(t) \geq \alpha t, \alpha \in [0, 1]}$$

 $\implies$

$$\boxed{l^* \leq l_\alpha^* = (1 - \alpha)l^{U_1} + \alpha l^{U_2}}$$



- The lower ReLU relaxation gives a sound upper plane.
- The bound of Shi et al. (2020) is exactly $\alpha = 0$, so it is a **special case**; $\alpha = 1$ gives the dual plane.

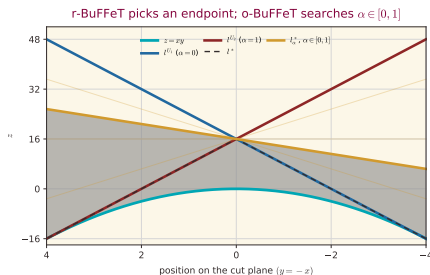
The ReLU Trick: Make the Choice Linear Again II

$$l^* = \min(l^{U_1}, l^{U_2}) = l^{U_1} - \text{ReLU}(t)$$

$$\boxed{\text{ReLU}(t) \geq \alpha t, \alpha \in [0, 1]}$$

 $\implies$

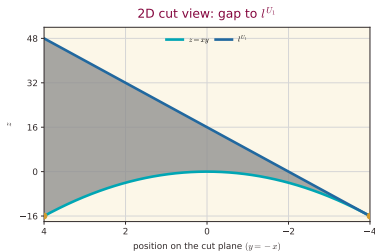
$$\boxed{l^* \leq l_\alpha^* = (1 - \alpha)l^{U_1} + \alpha l^{U_2}}$$



- The lower ReLU relaxation gives a sound upper plane.
- The bound of Shi et al. (2020) is exactly $\alpha = 0$, so it is a **special case**; $\alpha = 1$ gives the dual plane.
- Intermediate α values blend the two planes.
How should we choose them?

r-BuFFeT: Rule-Based Selection

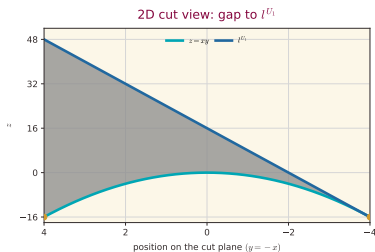
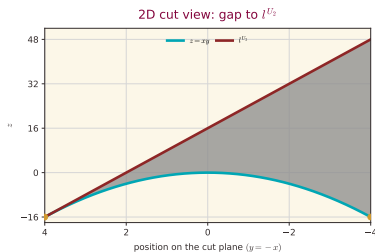
Range-based ReLU selection (Singh et al., 2019) for $t = l^{U_1} - l^{U_2} \in [a, b]$:



► If $|a| > |b|$, choose $\alpha = 0$, hence $l_{\alpha}^* = l^{U_1}$.

r-BuFFeT: Rule-Based Selection

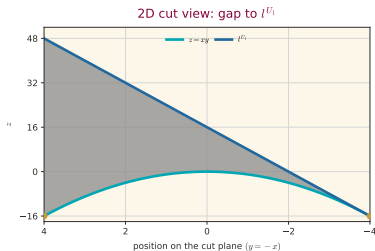
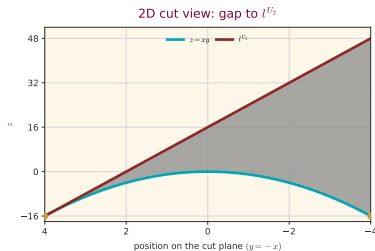
Range-based ReLU selection (Singh et al., 2019) for $t = l^{U_1} - l^{U_2} \in [a, b]$:



- If $|a| > |b|$, choose $\alpha = 0$, hence $l_{\alpha}^* = l^{U_1}$.
- If $|b| > |a|$, choose $\alpha = 1$, hence $l_{\alpha}^* = l^{U_2}$.

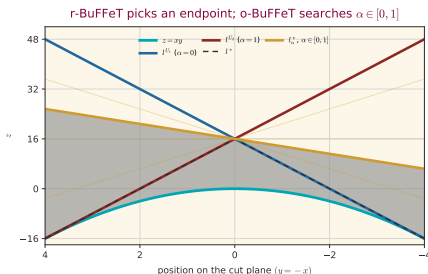
r-BuFFeT: Rule-Based Selection

Range-based ReLU selection (Singh et al., 2019) for $t = l^{U_1} - l^{U_2} \in [a, b]$:



- If $|a| > |b|$, choose $\alpha = 0$, hence $l_{\alpha}^* = l^{U_1}$.
- If $|b| > |a|$, choose $\alpha = 1$, hence $l_{\alpha}^* = l^{U_2}$.
- If $[a, b]$ does not cross zero, ReLU is already linear; otherwise r-BuFFeT makes this endpoint choice in one pass.

o-BuFFeT: Optimize α Like Network Training



Optimization loop

Inspired by Xu et al. (2021).

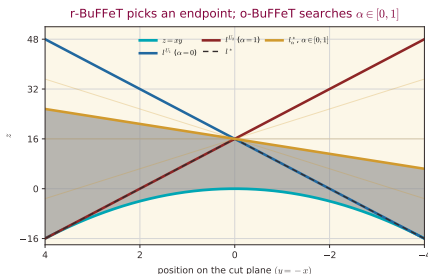
- The network contains many α values; optimize them jointly to improve $M(\alpha)$.

$$\alpha = (\alpha_1, \dots, \alpha_m) \in [0, 1]^m,$$

$$\mathcal{L}(\alpha) = \log\left(1 + e^{-M(\alpha)}\right),$$

$$\alpha^* = \arg \min_{\alpha \in [0, 1]^m} \mathcal{L}(\alpha).$$

o-BuFFeT: Optimize α Like Network Training



$$\alpha = (\alpha_1, \dots, \alpha_m) \in [0, 1]^m,$$

$$\mathcal{L}(\alpha) = \log\left(1 + e^{-M(\alpha)}\right),$$

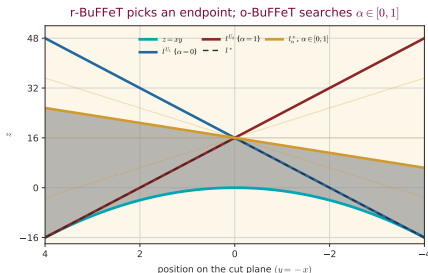
$$\alpha^* = \arg \min_{\alpha \in [0, 1]^m} \mathcal{L}(\alpha).$$

Optimization loop

Inspired by Xu et al. (2021).

- The network contains many α values; optimize them jointly to improve $M(\alpha)$.
- Keep every $\alpha_j \in [0, 1]$ for soundness; stop when $M(\alpha) > 0$.

o-BuFFeT: Optimize α Like Network Training



$$\alpha = (\alpha_1, \dots, \alpha_m) \in [0, 1]^m,$$

$$\mathcal{L}(\alpha) = \log\left(1 + e^{-M(\alpha)}\right),$$

$$\alpha^* = \arg \min_{\alpha \in [0, 1]^m} \mathcal{L}(\alpha).$$

Optimization loop

Inspired by Xu et al. (2021).

- ▶ The network contains many α values; optimize them jointly to improve $M(\alpha)$.
- ▶ Keep every $\alpha_j \in [0, 1]$ for soundness; stop when $M(\alpha) > 0$.
- ▶ A forward pass computes M ; autodiff and Adam update all α values like network training.

Introduction

Preliminary

Methodology

Experiments

Experimental Setting and Metrics

Parameters	Standard	TinyBERT
# Layers	1, 2, 3, 6	4
Attention heads	8	12
Activation	ReLU	ReLU
Hidden size	512	312/1200
Accuracy (%)	82.3–83.2 (SST) 91.2–91.4 (Yelp)	80.9 (SST) 91.5 (Yelp)

- Datasets: SST (Socher et al., 2013) and Yelp polarity (Zhang et al., 2015).

Experimental Setting and Metrics

Parameters	Standard	TinyBERT
# Layers	1, 2, 3, 6	4
Attention heads	8	12
Activation	ReLU	ReLU
Hidden size	512	312/1200
Accuracy (%)	82.3–83.2 (SST) 91.2–91.4 (Yelp)	80.9 (SST) 91.5 (Yelp)

- Datasets: SST (Socher et al., 2013) and Yelp polarity (Zhang et al., 2015).
- Models: standard encoder-only Transformers and TinyBERT.

Experimental Setting and Metrics

Parameters	Standard	TinyBERT
# Layers	1, 2, 3, 6	4
Attention heads	8	12
Activation	ReLU	ReLU
Hidden size	512	312/1200
Accuracy (%)	82.3–83.2 (SST) 91.2–91.4 (Yelp)	80.9 (SST) 91.5 (Yelp)

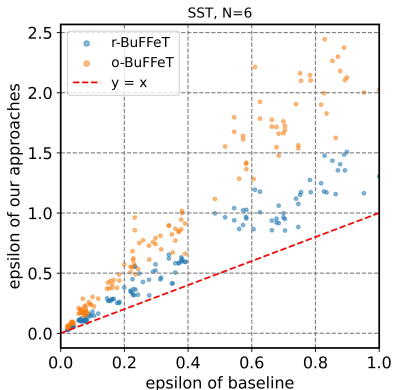
- Datasets: SST (Socher et al., 2013) and Yelp polarity (Zhang et al., 2015).
- Models: standard encoder-only Transformers and TinyBERT.
- **Precision metric:** maximal verified perturbation radius ϵ ; larger is better.

Experimental Setting and Metrics

Parameters	Standard	TinyBERT
# Layers	1, 2, 3, 6	4
Attention heads	8	12
Activation	ReLU	ReLU
Hidden size	512	312/1200
Accuracy (%)	82.3–83.2 (SST) 91.2–91.4 (Yelp)	80.9 (SST) 91.5 (Yelp)

- Datasets: SST (Socher et al., 2013) and Yelp polarity (Zhang et al., 2015).
- Models: standard encoder-only Transformers and TinyBERT.
- **Precision metric:** maximal verified perturbation radius ϵ ; larger is better.
- **Efficiency metric:** verification time cost; smaller is better.

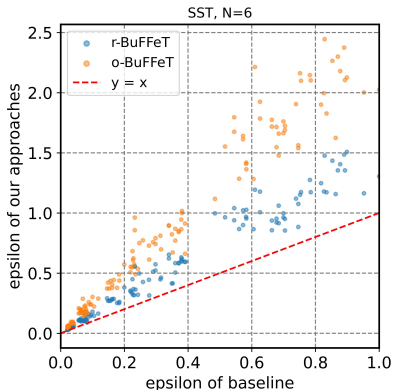
RQ1: Better Precision



Most points are above
 $y = x$.

- BuFFeT verifies larger radii than CrownBaF (Shi et al., 2020).

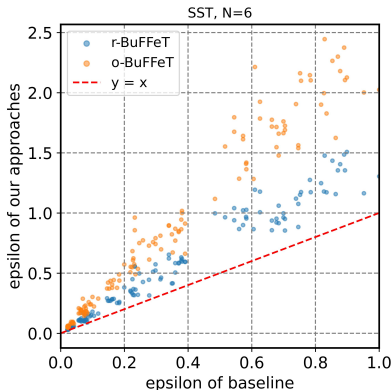
RQ1: Better Precision



Most points are above
 $y = x$.

- BuFFeT verifies larger radii than CrownBaF (Shi et al., 2020).
- On 6-layer models: r-BuFFeT up to $1.8\times$.

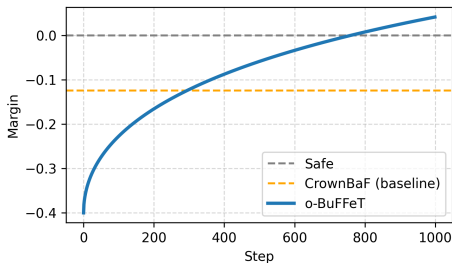
RQ1: Better Precision



Most points are above
 $y = x$.

- BuFFeT verifies larger radii than CrownBaF (Shi et al., 2020).
- On 6-layer models: r-BuFFeT up to $1.8\times$.
- o-BuFFeT up to $3.6\times$, with higher cost.

RQ2: Optimization Improves Verification

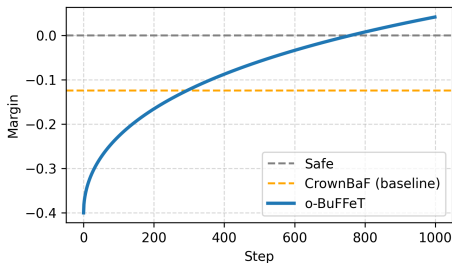


Representative optimization trajectory

o-BuFFeT optimizes α for the final margin.

- The blue curve is the verification margin; gray marks 0.

RQ2: Optimization Improves Verification

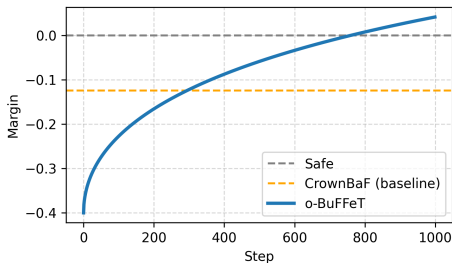


Representative optimization trajectory

o-BuFFeT optimizes α for the final margin.

- The blue curve is the verification margin; gray marks 0.
- Optimizing α raises the margin until the target radius is verified.

RQ2: Optimization Improves Verification



Representative optimization trajectory

o-BuFFeT optimizes α for the final margin.

- The blue curve is the verification margin; gray marks 0.
- Optimizing α raises the margin until the target radius is verified.
- o-BuFFeT is an **effective refinement.**

Conclusion

- Bilinear products in attention are a **precision bottleneck** for Transformer verification.

Conclusion

- ▶ Bilinear products in attention are a **precision bottleneck** for Transformer verification.
- ▶ BuFFeT fuses two sound planes through ReLU: r-BuFFeT selects an endpoint, while o-BuFFeT optimizes α .

Conclusion

- ▶ Bilinear products in attention are a **precision bottleneck** for Transformer verification.
- ▶ BuFFeT fuses two sound planes through ReLU: r-BuFFeT selects an endpoint, while o-BuFFeT optimizes α .
- ▶ Experiments confirm **more precise verification**: up to $3.6\times$ larger verified radii, while r-BuFFeT stays close to baseline runtime.

Thank you

Thanks for your listening!

Appendix: Contributions

- **ReLU-catalyzed representation.** BuFFeT fuses the existing planar dot-product bound and our proposed dual through ReLU, showing that both fixed planes are special, potentially suboptimal cases.

Appendix: Contributions

- ▶ **ReLU-catalyzed representation.** BuFFeT fuses the existing planar dot-product bound and our proposed dual through ReLU, showing that both fixed planes are special, potentially suboptimal cases.
- ▶ **Two refinement strategies.** r-BuFFeT adapts range-based ReLU selection (Singh et al., 2019); o-BuFFeT adapts optimization-based bound search (Xu et al., 2021).

Appendix: Contributions

- ▶ **ReLU-catalyzed representation.** BuFFeT fuses the existing planar dot-product bound and our proposed dual through ReLU, showing that both fixed planes are special, potentially suboptimal cases.
- ▶ **Two refinement strategies.** r-BuFFeT adapts range-based ReLU selection (Singh et al., 2019); o-BuFFeT adapts optimization-based bound search (Xu et al., 2021).
- ▶ **Evaluation and impact.** Across 10 models and 1400 verification tasks, BuFFeT improves precision for most tasks with an acceptable efficiency compromise, while bridging classic ReLU verification and Transformer verification.

Appendix: RQ3 Runtime Data

Dataset	N	baseline	r-BuFFeT	Δ	o-BuFFeT	Δ
SST	1	0.06	0.09	1.5	5.77	96.2
	2	0.17	0.26	1.5	14.70	86.5
	3	1.05	1.76	1.7	101.72	96.9
	6	11.38	12.57	1.1	973.50	85.5
	TinyBERT	4.30	5.27	1.2	281.07	65.4
Yelp	1	0.08	0.13	1.6	2.68	33.5
	2	0.17	0.23	1.4	12.97	76.3
	3	0.92	1.26	1.4	77.61	84.4
	6	8.05	9.43	1.2	673.42	83.7
	TinyBERT	2.29	3.34	1.5	175.38	76.6

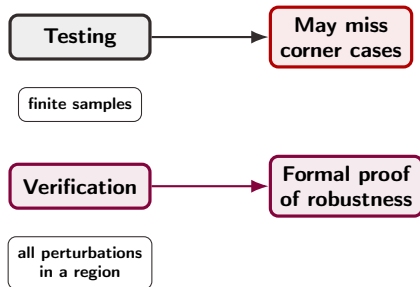
- r-BuFFeT stays close to baseline runtime and is the practical default.

Appendix: RQ3 Runtime Data

Dataset	N	baseline	r-BuFFeT	Δ	o-BuFFeT	Δ
SST	1	0.06	0.09	1.5	5.77	96.2
	2	0.17	0.26	1.5	14.70	86.5
	3	1.05	1.76	1.7	101.72	96.9
	6	11.38	12.57	1.1	973.50	85.5
	TinyBERT	4.30	5.27	1.2	281.07	65.4
Yelp	1	0.08	0.13	1.6	2.68	33.5
	2	0.17	0.23	1.4	12.97	76.3
	3	0.92	1.26	1.4	77.61	84.4
	6	8.05	9.43	1.2	673.42	83.7
	TinyBERT	2.29	3.34	1.5	175.38	76.6

- r-BuFFeT stays close to baseline runtime and is the practical default.
- o-BuFFeT pays substantially more time for more precise verification.

Appendix: Testing Is Not Enough



Testing samples a few inputs.

Verification reasons about all inputs in the perturbation region.

If the over-approximation is safe, the original outputs are safe.

References I

- Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. Robustness verification for transformers. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=BJxwPJHFWs>.
- Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin Vechev. An abstract domain for certifying neural networks. *Proc. ACM Program. Lang.*, 3(POPL), January 2019. doi: 10.1145/3290354.
- Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard, editors, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics. URL <https://aclanthology.org/D13-1170/>.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
- Kaidi Xu, Huan Zhang, Shiqi Wang, Yihan Wang, Suman Jana, Xue Lin, and Cho-Jui Hsieh. Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=nVZtXBI6LNn>.
- Huan Zhang, Tsui-Wei Weng, Pin-Yu Chen, Cho-Jui Hsieh, and Luca Daniel. Efficient neural network robustness certification with general activation functions. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS'18*, page 4944–4953, Red Hook, NY, USA, 2018. Curran Associates Inc. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/d04863f100d59b3eb688a11f95b0ae60-Paper.pdf.

References II

Xiang Zhang, Junbo Zhao, and Yann LeCun. Character-level convolutional networks for text classification. In *Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 649–657, Cambridge, MA, USA, 2015. MIT Press. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/250cf8b51c773f3f8dc8b4be867a9a02-Paper.pdf.