Sensitivity as a Complexity Measure for Sequence Classification Tasks

Michael Hahn

Stanford University

with Richard Futrell and Dan Jurafsky

March 31, 2021

The Stanford Natural Language Processing Group

Motivation

What makes some NLP tasks harder and others easier?

Motivation

What makes some NLP tasks harder and others easier?

Simple models based on lexical classifiers provide good performance on some tasks.

sentiment analysis

POS tagging

...

Motivation

What makes some NLP tasks harder and others easier?

Simple models based on lexical classifiers provide good performance on some tasks.

- sentiment analysis
- POS tagging
- ...

On other tasks, strong performance attained only recently with massive pretrained models.

- Winograd sentences
- Entailment
- ...

Motivation

What makes some NLP tasks harder and others easier?

Simple models based on lexical classifiers provide good performance on some tasks.

sentiment analysis

POS tagging

...

On other tasks, strong performance attained only recently with massive pretrained models.

Winograd sentences

Entailment

...

This talk: Propose a theoretical framework to formalize and capture these differences.

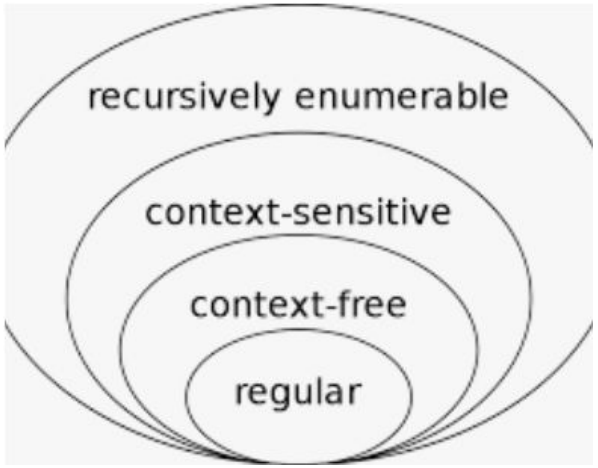
Existing Complexity Metrics

Chomsky Hierarchy (Chomsky, 1956)

Existing Complexity Metrics

Chomsky Hierarchy (Chomsky, 1956)

prominent classification of formal languages by complexity

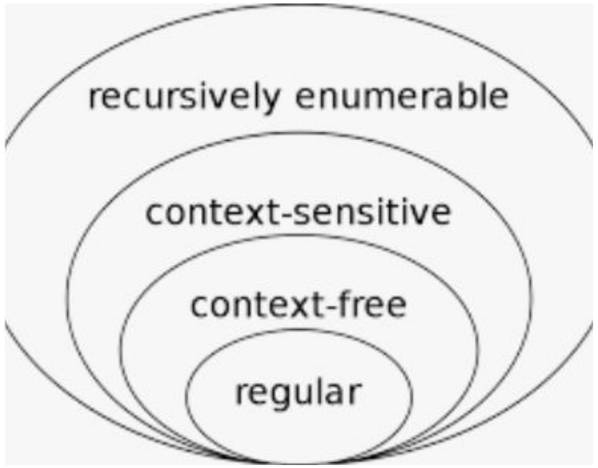


Existing Complexity Metrics

Chomsky Hierarchy (Chomsky, 1956)

prominent classification of formal languages by complexity

asymptotic worst-case complexity

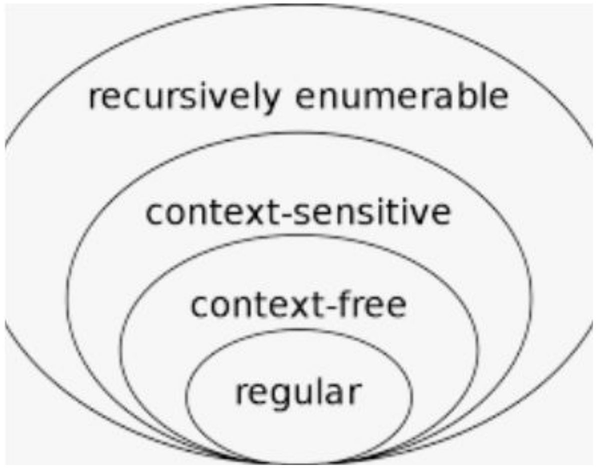


Existing Complexity Metrics

Chomsky Hierarchy (Chomsky, 1956)

prominent classification of formal languages by complexity

asymptotic worst-case complexity



does not measure how hard it is to achieve high accuracy on realistic task distributions.

Existing Complexity Metrics

Kolmogorov complexity (Li and Vitányi, 1993)

Existing Complexity Metrics

Kolmogorov complexity (Li and Vitányi, 1993)

length of the shortest program producing an output

Existing Complexity Metrics

Kolmogorov complexity (Li and Vitányi, 1993)

length of the shortest program producing an output

uncomputable

Existing Complexity Metrics

Kolmogorov complexity (Li and Vitányi, 1993)

length of the shortest program producing an output

uncomputable

well-defined only in the asymptotic limit

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

Sensitivity Bounds for ML Methods

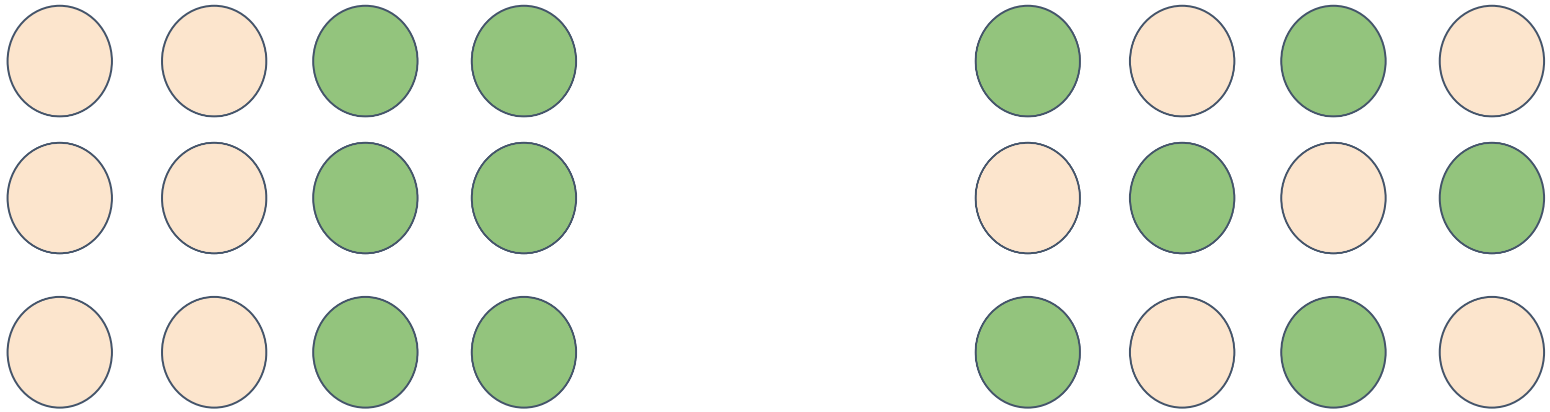
Sensitivity and Difficulty of NLP Tasks

Sensitivity

<u>Idea:</u> Tasks are difficult when they have complex decision boundaries.

Simple Task

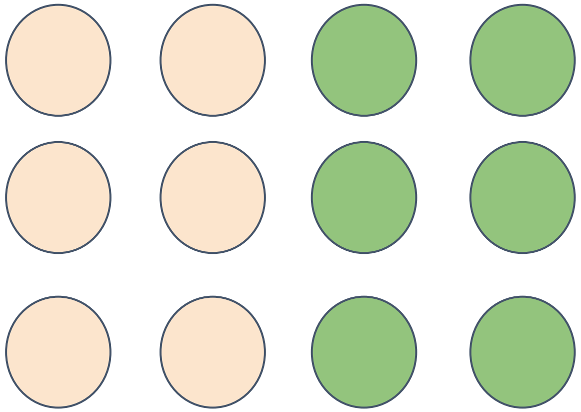
Difficult Task



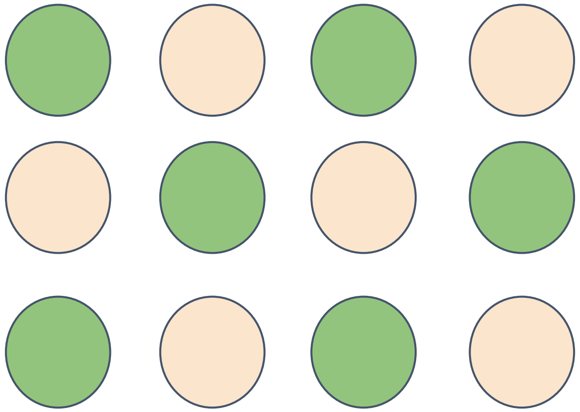
Sensitivity

Idea: Tasks are difficult when they have complex decision boundaries.

Simple Task



Difficult Task

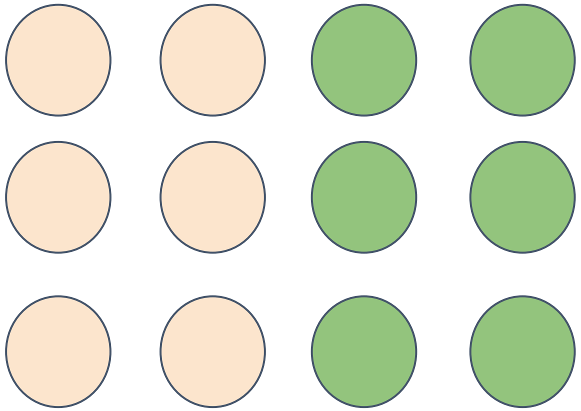


Recent work highlights need to evaluate NLP models at their decision boundary (e.g., Levesque et al., 2011; Jia and Liang, 2017; Ribeiro et al., 2018; Gardner et al., 2019; Kaushik et al., 2019).

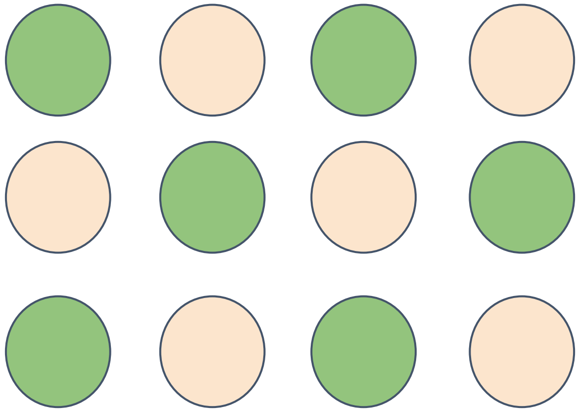
Sensitivity

When is a decision boundary complex?

Simple Task



Difficult Task

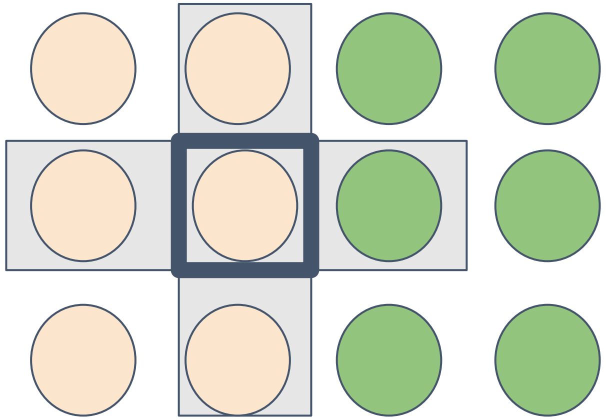


Sensitivity

When is a decision boundary complex?

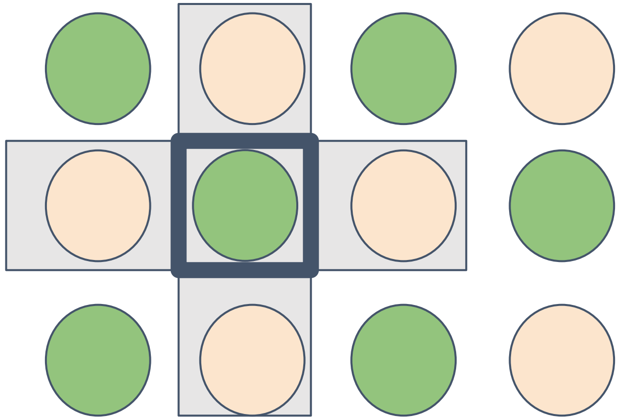
When the label often varies between neighbors!

Simple Task



Most neighbors have the same label as the point

Difficult Task



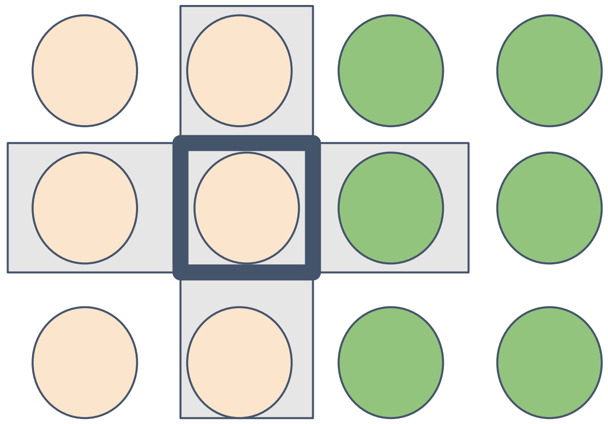
Neighbors have the opposite label as the point

Sensitivity

When is a decision boundary complex?

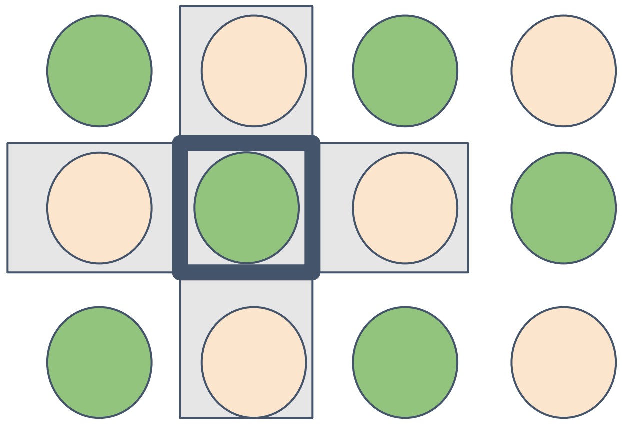
When the label often varies between neighbors!

Simple Task



Can predict label even looking at part of the input.

Difficult Task



Changing any part of the input can flip the label.

Sensitivity of Boolean Functions

For a function $f : \{-1,+1\}^n \rightarrow \{-1,+1\}$:

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

For a function f : $\{-1,+1\}^n \rightarrow \{-1,+1\}$:

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

Input string
x in $\{-1,+1\}^n$

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

For a function f : {-1,+1}^{n} $\longrightarrow$ {-1,+1}:

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

Input string
x in {-1,+1}^{n}

Result of flipping
the i-th bit

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

For a function f : {-1,+1}^{n} → {-1,+1}:

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

Input string
x in {-1,+1}^{n}

Result of flipping
the i-th bit

"How many Hamming neighbors of x have the opposite label?"

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity

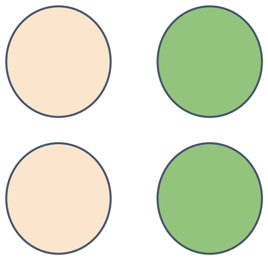
High Sensitivity

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



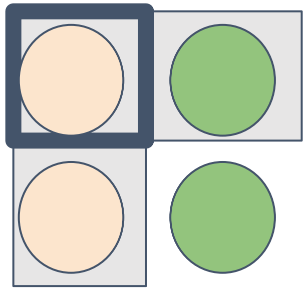
(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity

High Sensitivity

Functions that depend on few inputs

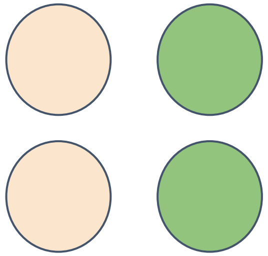


(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

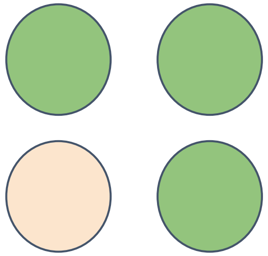
Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



Functions that are close to linear

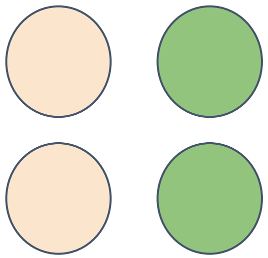


(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

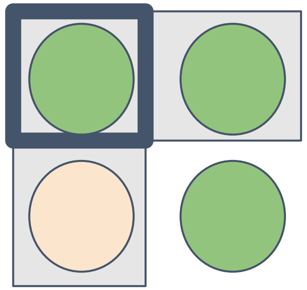
Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



Functions that are close to linear

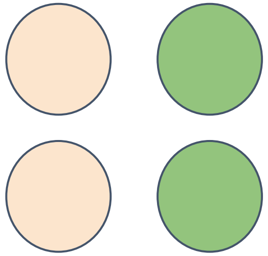


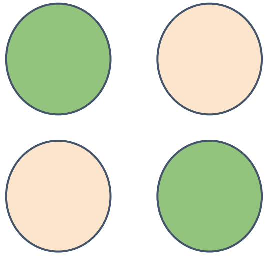
(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

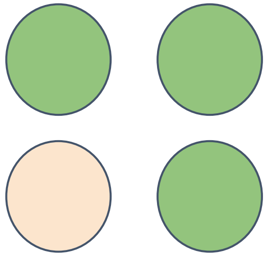
Functions that depend on few inputs





$$f_{Parity}(x) := \prod_{i=1}^{n} x_i$$

Functions that are close to linear

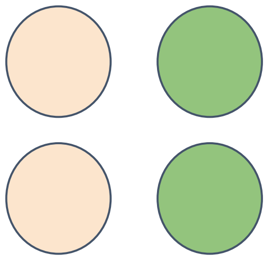


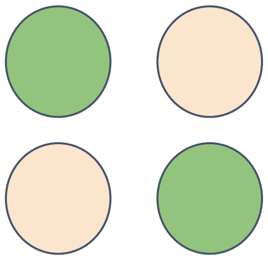
(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs

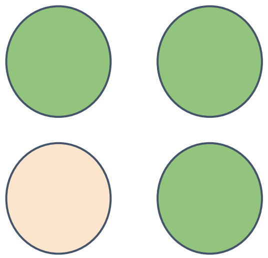




$$f_{Parity}(x) := \prod_{i=1}^{n} x_i$$

$+1 \quad +1 \quad \rightarrow \quad +1$

Functions that are close to linear

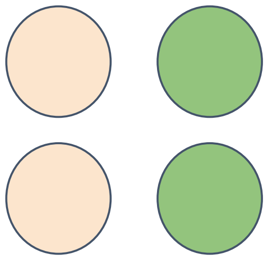


(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O’Donnell, 2014; Huang 2019)

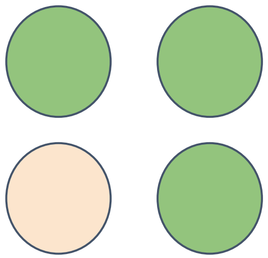
Sensitivity of Boolean Functions

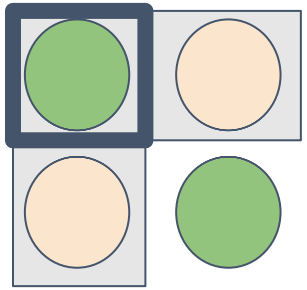
Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



Functions that are close to linear





$$f_{Parity}(x) := \prod_{i=1}^{n} x_i$$

$+1 \quad +1 \rightarrow +1$

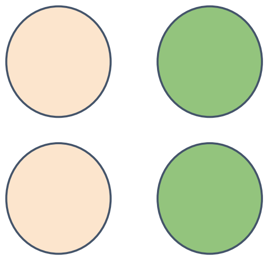
$-1 \quad +1 \rightarrow -1$

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

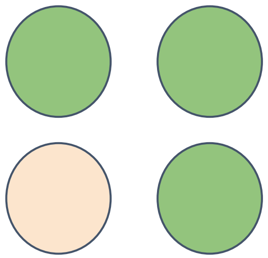
Sensitivity of Boolean Functions

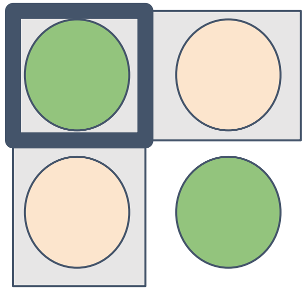
Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



Functions that are close to linear





$$f_{Parity}(x) := \prod_{i=1}^{n} x_i$$

$$+1 \quad {\color{blue}+1} \quad \rightarrow \quad {\color{blue}+1}$$

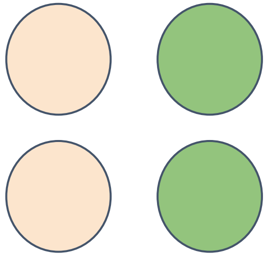
$$+1 \quad {\color{red}-1} \quad \rightarrow \quad {\color{red}-1}$$

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

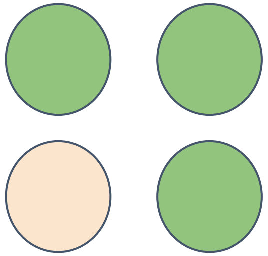
Sensitivity of Boolean Functions

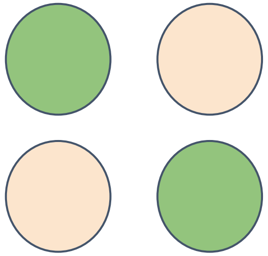
Low Sensitivity ⟷ **High Sensitivity**

Functions that depend on few inputs



Functions that are close to linear





$$f_{Parity}(x) := \prod_{i=1}^{n} x_i$$

$+1 \quad {\color{blue}+1} \rightarrow {\color{blue}+1}$

$+1 \quad {\color{red}-1} \rightarrow {\color{red}-1}$

$s(f_{Parity}, x) = n$ for any x, $|x|=n$

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Well approximated with **linear functions** ⟷ Impossible to approximate with **linear functions**

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity ⟷ **High Sensitivity**

Well approximated with linear functions ⟷ Impossible to approximate with linear functions

Shallow **decision trees** ⟷ Deep **decision trees**

(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Sensitivity of Boolean Functions

Low Sensitivity	↔	High Sensitivity
Well approximated with linear functions	↔	Impossible to approximate with linear functions
Shallow decision trees	↔	Deep decision trees
Low-degree **polynomials**	↔	High-degree **polynomials**

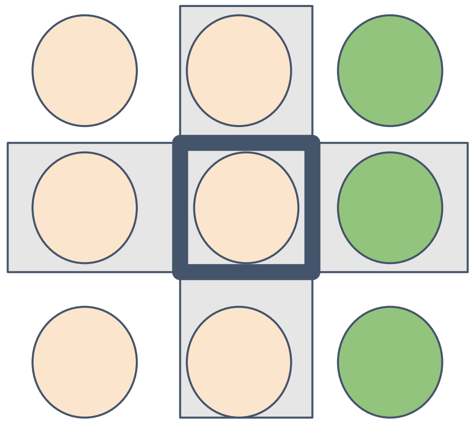
(Kahn et al., 1988; Nisan, 1991; Hatami et al., 2010; O'Donnell, 2014; Huang 2019)

Applying Sensitivity in NLP

When is a decision boundary complex?

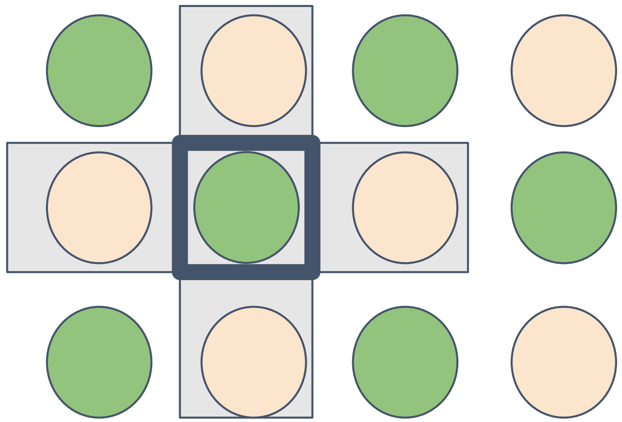
When the label often varies between neighbors!

Simple Task



Most neighbors have the same label as the point

Difficult Task



Neighbors have the opposite label as the point

Applying Sensitivity in NLP

Desiderata:

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements

$$s(f, x) := \sum_{i=1}^{n} \mathrm{Var}\left(f(X) | \forall j \neq i : X_j = x_j\right)$$

Inputs X' that agree with x in all but the i-th input

(O'Donnell, 2014, Def. 8.22)

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements
2. Nonuniform distribution

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements
2. Nonuniform distribution

Inputs typically respect grammatical structure of language

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements
2. Nonuniform distribution

Inputs typically respect grammatical structure of language

Task-specific input distributions

Applying Sensitivity in NLP

Desiderata:

1. Alphabets with more than two elements
2. Nonuniform distribution

Inputs typically respect grammatical structure of language

Task-specific input distributions

When measuring task difficulty, we want to focus on inputs that are plausible problem instances

Sensitivity

Alphabet Σ (e.g. words, BPE, characters)

Sensitivity

Alphabet Σ (e.g. words, BPE, characters)

Distribution $\prod$ over the set Σ^* of finite strings

an amazing movie

what a dumb movie

mostly boring

stunning visuals

this was hilarious

i can’t believe i wasted my
time on this dumb movie

truly incredible, great
plot and good acting

Sensitivity

Alphabet Σ (e.g. words, BPE, characters)

Distribution $\prod$ over the set Σ^* of finite strings

Classification task = Function $f : \Sigma^* \rightarrow [-1,1]$

an amazing movie **+1**

what a dumb movie **-1**

mostly boring **-1**

stunning visuals **+1**

this was hilarious **+1**

i can't believe i wasted my time on this dumb movie **-1**

truly incredible, great plot and good acting **+1**

Sensitivity

Alphabet Σ (e.g. words, BPE, characters)

Distribution ∏ over the set Σ* of finite strings

Classification task = Function f : Σ* → [-1,1]

an amazing movie +1

what a dumb movie -1

mostly boring -1

stunning visuals +1

this was hilarious +1

i can't believe i wasted my time on this dumb movie -1

truly incredible, great plot and good acting +1

Sensitivity:

In how many places can we change the input to change the output label while respecting ∏?

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

$$s(f,x) = \sum_{i=1}^{n} 1_{f(x) \neq f(x^{\oplus i})}$$

Subset Sensitivity

$$s(f,x,P) := \mathrm{Var}\left(f(X)|X \in x^{\oplus P}\right)$$

Input string in Σ*

KJHJKTGFJKJTGHHKJ

Subset Sensitivity

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

Input string in Σ*

KJHJKTGFJKJTGHHKJ

Subset of {1, …, |x|}

KJHJKTGFJKJTGHHKJ

Subset Sensitivity

$$s(f,x,P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

Input string in Σ*

KJHJKTGFJKJTGHHKJ

Subset of {1, …, |x|}

KJHJKTGFJKJTGHHKJ

Inputs that agree with x outside of positions in P.

Subset Sensitivity

$$s(f, x, P) := \mathrm{Var}\left(f(X) \middle| X \in x^{\oplus P}\right)$$

Input string in Σ*

KJHJKTGFJKJTGHHKJ

Subset of {1, …, |x|}

KJHJKTGFJKJTGHHKJ

KJIFNTGFJKBASHHKJ	-1
KJQWFTGFJKKHYHHKJ	+1
KJNFATGFJKTBZHHKJ	-1
KJMZXTGFJKUASHHKJ	-1

……

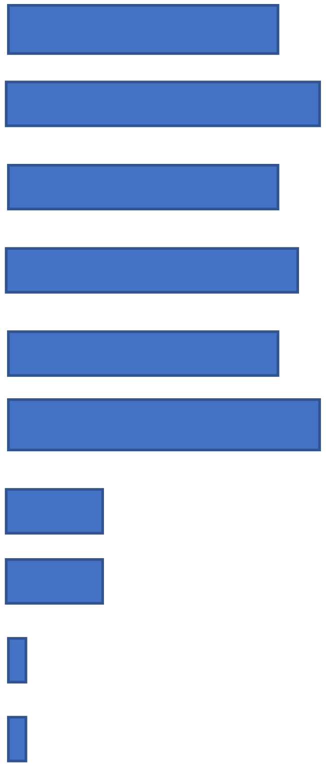
$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

a gorgeous , witty , seductive movie . (+1)

$$s(f,x,P) := \mathrm{Var}\left(f(X)|X \in x^{\oplus P}\right)$$

a gorgeous , witty , seductive movie . (+1)

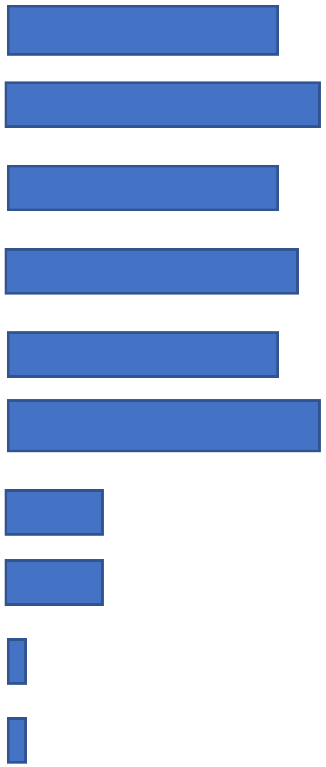
A brilliant, witty, seductive movie.
A cute, witty, seductive movie.
A sexy, witty, seductive movie.
A shocking, witty, seductive movie.
A stylish, witty, seductive movie.
A charming, witty, seductive movie.
A boring, witty, seductive movie.
A bad, witty, seductive movie.
A crocodile, witty, seductive movie.
A oxymoron, witty, seductive movie.



$$s(f,x,P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

a gorgeous , witty , seductive movie . (+1)

A brilliant, witty, seductive movie.
A cute, witty, seductive movie.
A sexy, witty, seductive movie.
A shocking, witty, seductive movie.
A stylish, witty, seductive movie.
A charming, witty, seductive movie.
A boring, witty, seductive movie.
A bad, witty, seductive movie.
A crocodile, witty, seductive movie.
A oxymoron, witty, seductive movie.

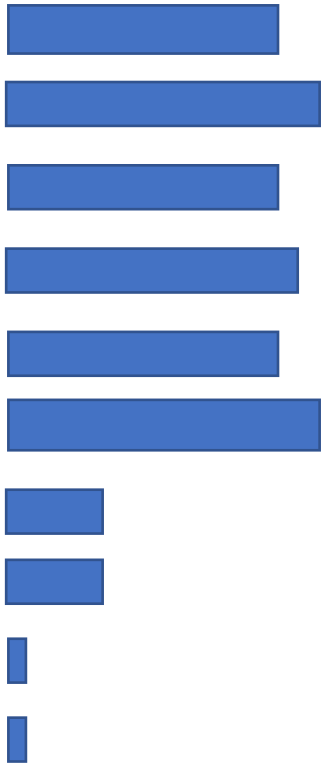


All high-probability neighbors have positive sentiment

$$s(f,x,P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

a gorgeous , witty , seductive movie . (+1)

A brilliant, witty, seductive movie.
A cute, witty, seductive movie.
A sexy, witty, seductive movie.
A shocking, witty, seductive movie.
A stylish, witty, seductive movie.
A charming, witty, seductive movie.
A boring, witty, seductive movie.
A bad, witty, seductive movie.
A crocodile, witty, seductive movie.
A oxymoron, witty, seductive movie.



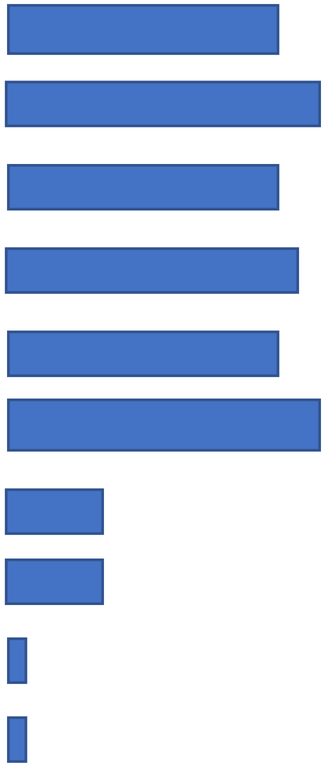
All high-probability neighbors have positive sentiment

s(f,x,P) is low

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

a gorgeous , witty , seductive movie . (+1)

A brilliant, witty, seductive movie.
A cute, witty, seductive movie.
A sexy, witty, seductive movie.
A shocking, witty, seductive movie.
A stylish, witty, seductive movie.
A charming, witty, seductive movie.
A boring, witty, seductive movie.
A bad, witty, seductive movie.
A crocodile, witty, seductive movie.
A oxymoron, witty, seductive movie.



All high-probability neighbors have positive sentiment

s(f,x,P) is low

“We still know the label even if we don’t know the blue word.”

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

A gorgeous, witty, seductive movie.

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

A gorgeous, witty, seductive movie.

A brilliant, amazing, convincing movie.
A boring, annoying, disappointing movie.

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

A gorgeous, witty, seductive movie.

A brilliant, amazing, convincing movie.
A boring, annoying, disappointing movie.

Both sentiments represented among high-probability neighbors

$$s(f,x,P) := \mathrm{Var}\left(f(X)|X \in x^{\oplus P}\right)$$

A gorgeous, witty, seductive movie.

A brilliant, amazing, convincing movie.
A boring, annoying, disappointing movie.

Both sentiments represented among high-probability neighbors

s(f,x,P) is high

$$s(f, x, P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

A gorgeous, witty, seductive movie.

A brilliant, amazing, convincing movie.
A boring, annoying, disappointing movie.

Both sentiments represented among high-probability neighbors

s(f,x,P) is high

"We don't know the label if we don't know the blue words."

Block Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Partitions of {1, …, |x|}
into disjoint subsets

Block Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

"In how many different places can we change the input to flip the label?"

Block Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

“In how many different places can we change the input to flip the label?”

...while respecting input distribution ∏.

Block Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

“In how many different places can we change the input to flip the label?”

...while respecting input distribution ∏.

New probabilistic adaptation of previously-defined block sensitivity of Boolean functions (Nisan, 1991; Bernasconi, 1996; Hatami et al., 2010).

Low Sensitivity:

a gorgeous , witty , seductive movie .

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

$s(f,x,P) = 0.93$

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

s(f,x,P) = 0.93

block sensitivity: 0.93

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

High Sensitivity:

a painfully funny ode to bad behavior .

s(f,x,P) = 0.93

block sensitivity: 0.93

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie . s(f,x,P) = 0.93

block sensitivity: 0.93

High Sensitivity:

a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior . s(f,x,P) = 0.96

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie . s(f,x,P) = 0.93

block sensitivity: 0.93

High Sensitivity:

a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior . s(f,x,P) = 0.96
2. a painfully bleak ode to bad behavior . s(f,x,P) = 0.74

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie . — s(f,x,P) = 0.93

block sensitivity: 0.93

High Sensitivity:

a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior . — s(f,x,P) = 0.96
2. a painfully bleak ode to bad behavior . — s(f,x,P) = 0.74
3. a painfully funny ode to bad movies . — s(f,x,P) = 0.18

block sensitivity: 1.88

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

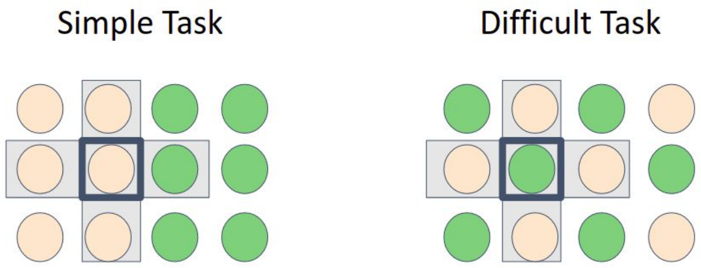
Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

formalizes complexity of decision boundary



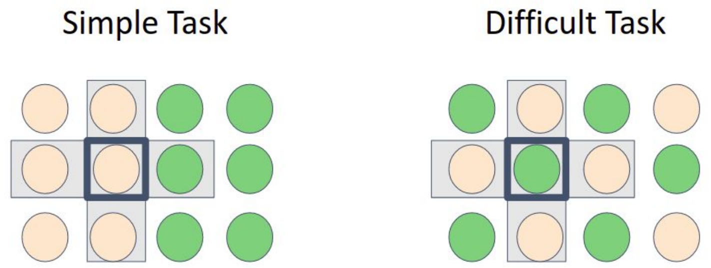
Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

formalizes complexity of decision boundary



generalizes theory from Boolean functions to general sequence classification

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \dots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

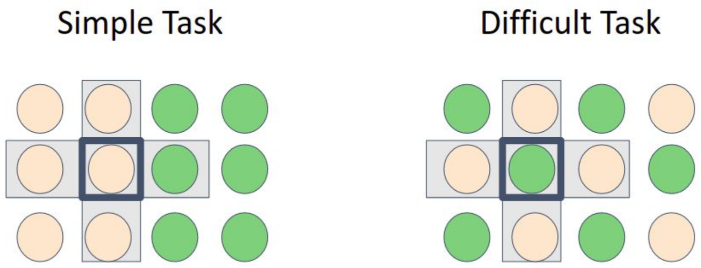
Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

formalizes complexity of decision boundary



generalizes theory from Boolean functions to general sequence classification

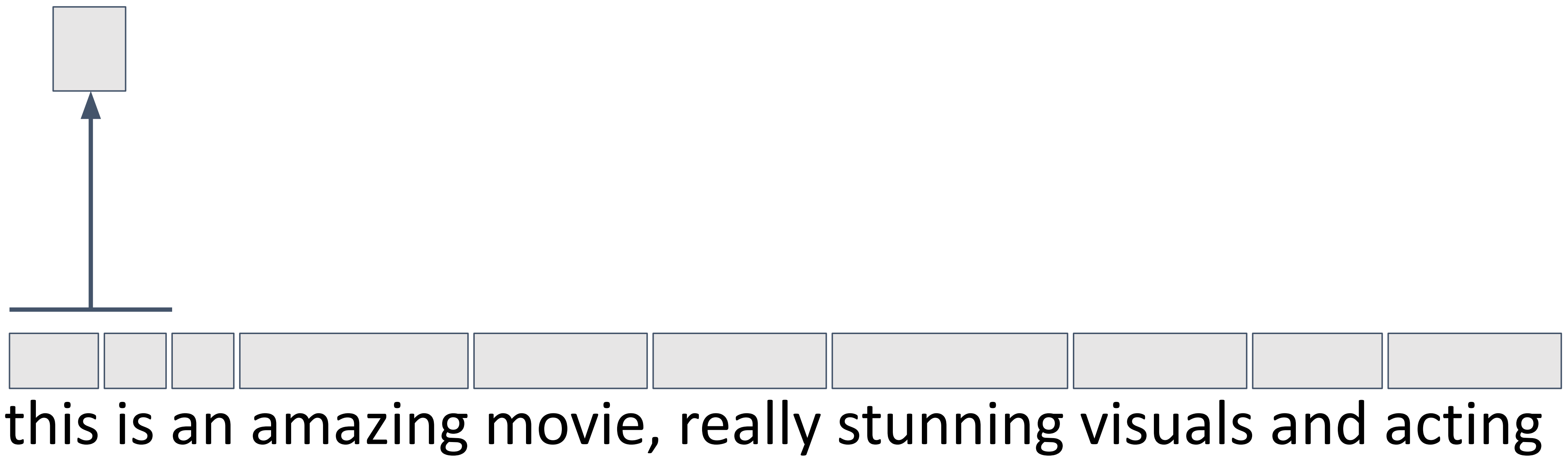
$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

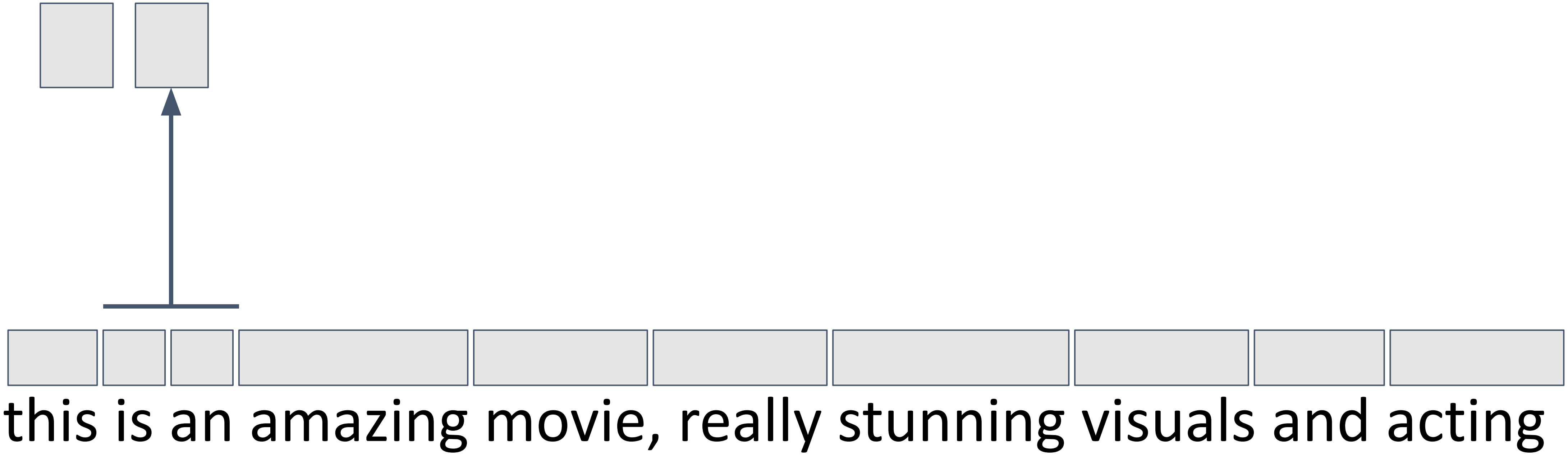
Sensitivity Bounds for ML Methods

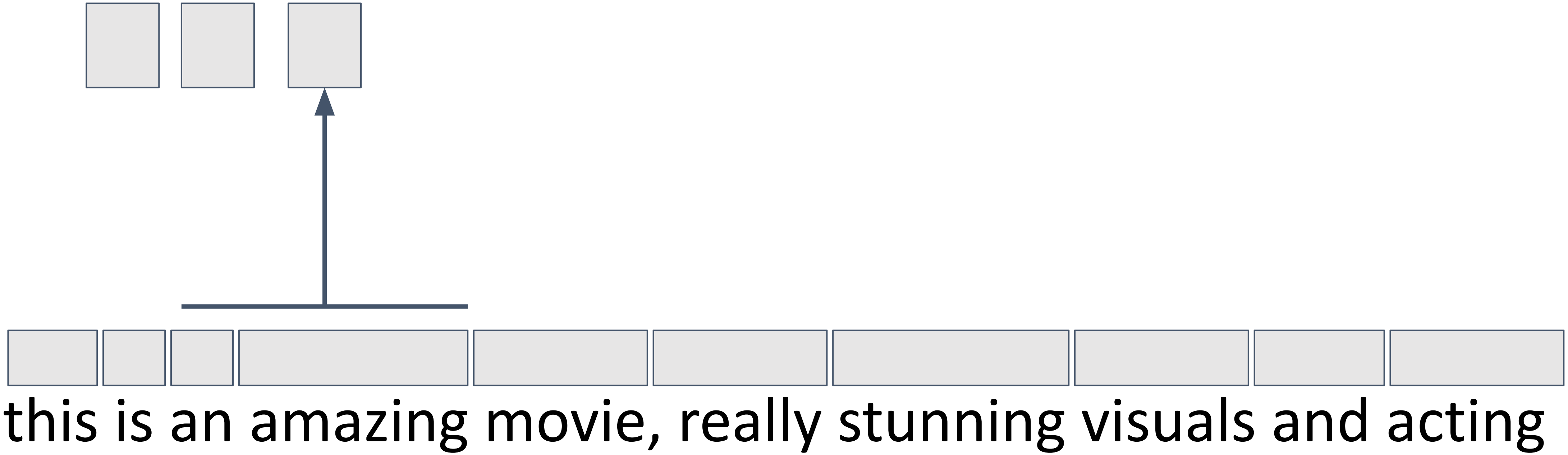
Sensitivity and Difficulty of NLP Tasks

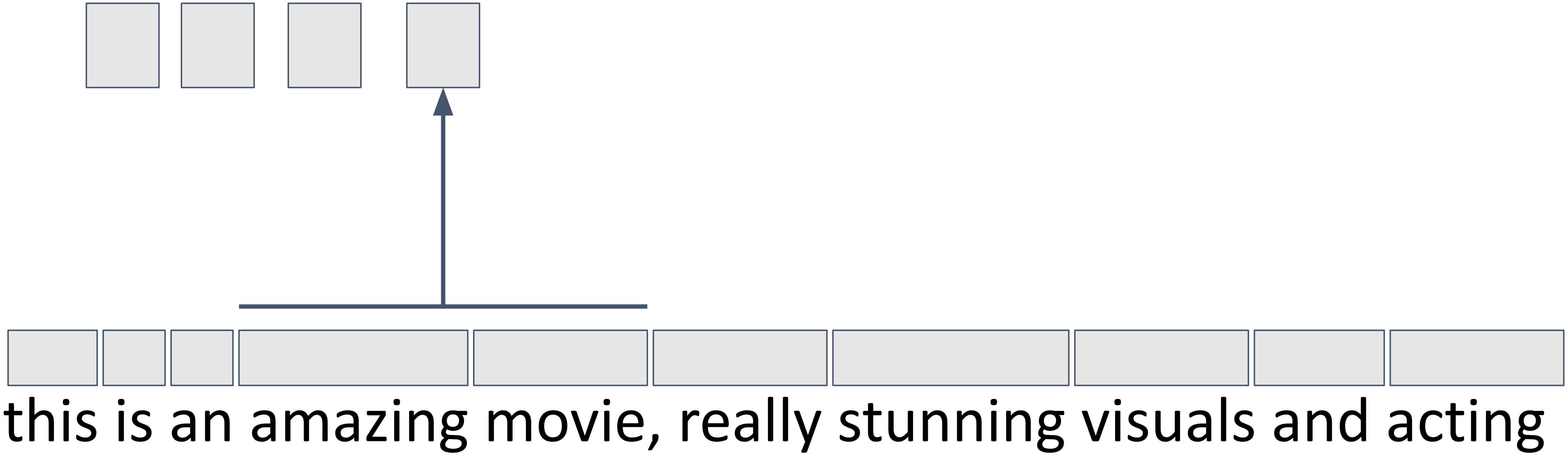
	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...		
LSTM		
Transformer		

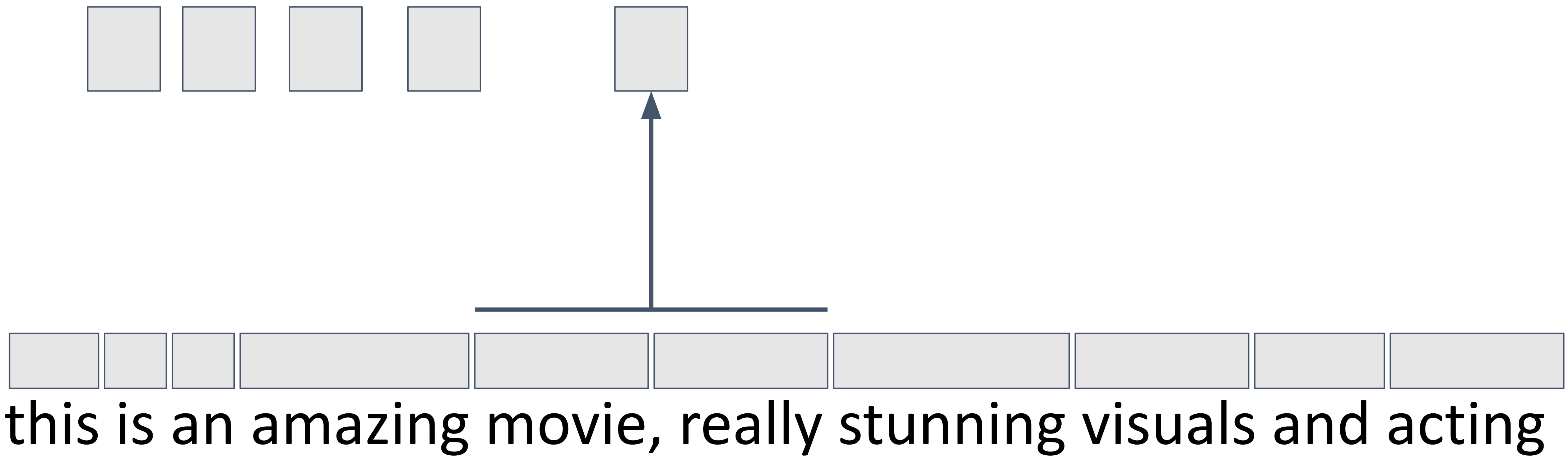
this is an amazing movie, really stunning visuals and acting

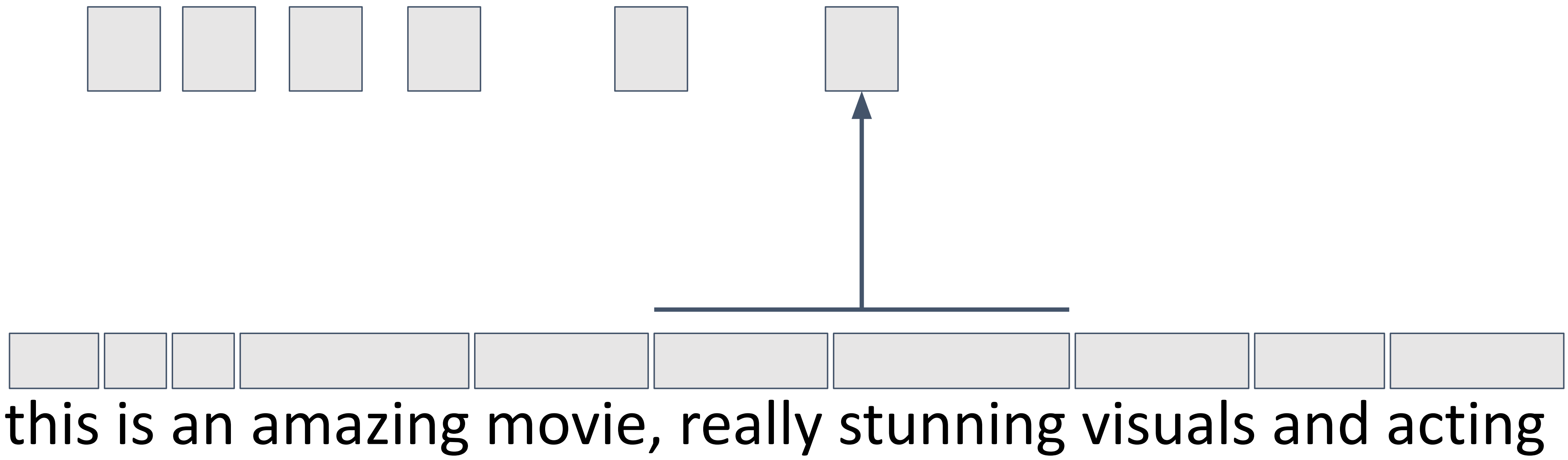

this is an amazing movie, really stunning visuals and acting

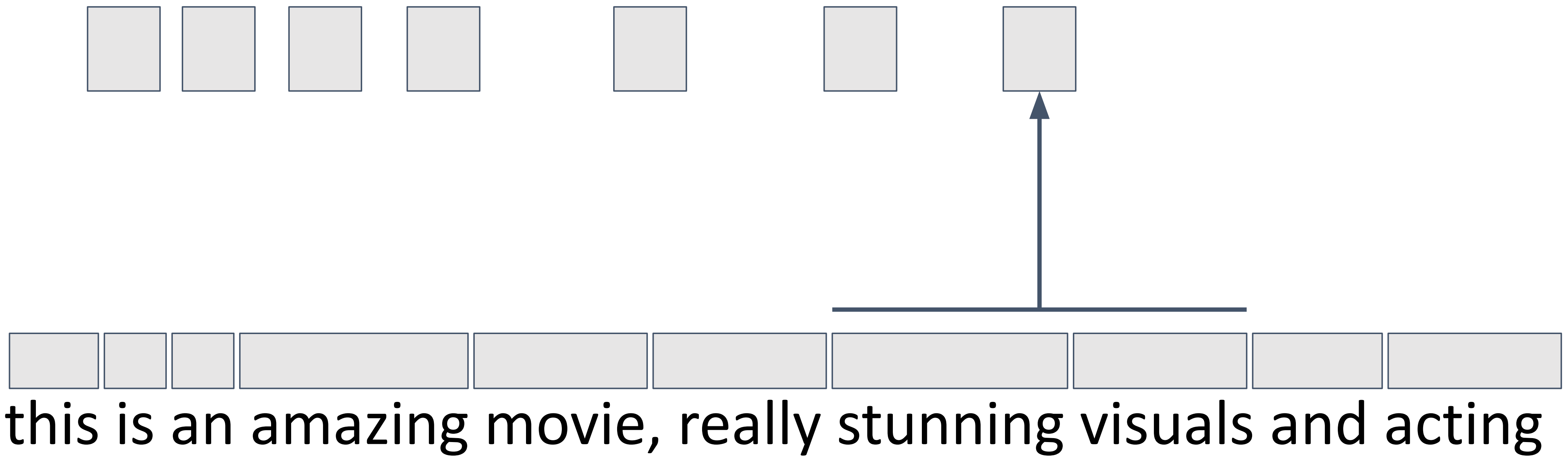

this is an amazing movie, really stunning visuals and acting

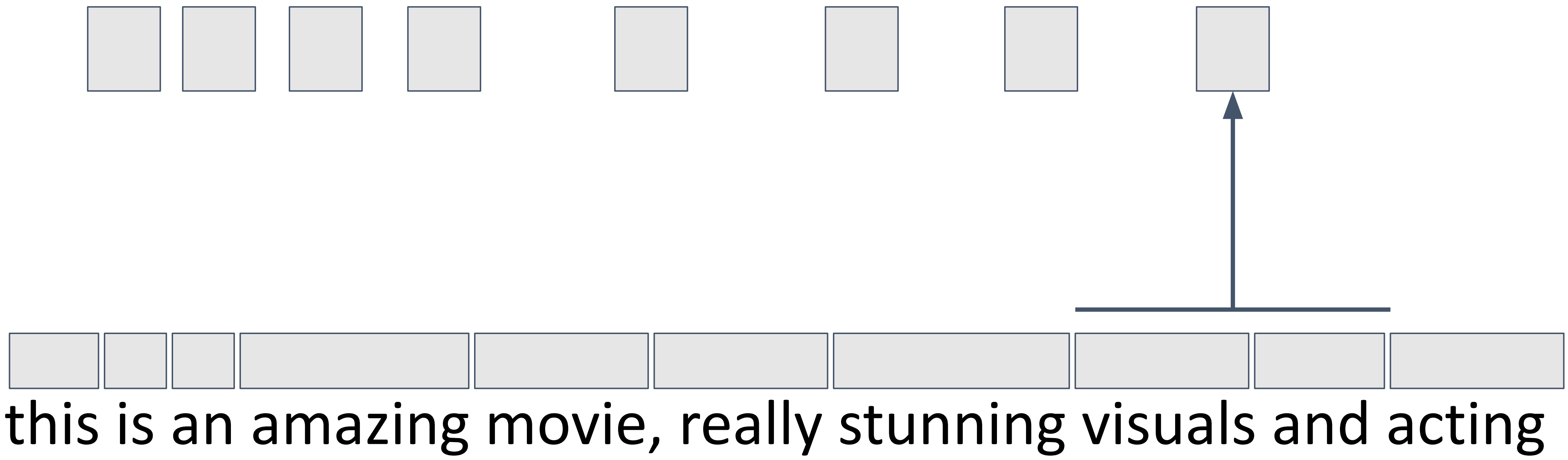

this is an amazing movie, really stunning visuals and acting

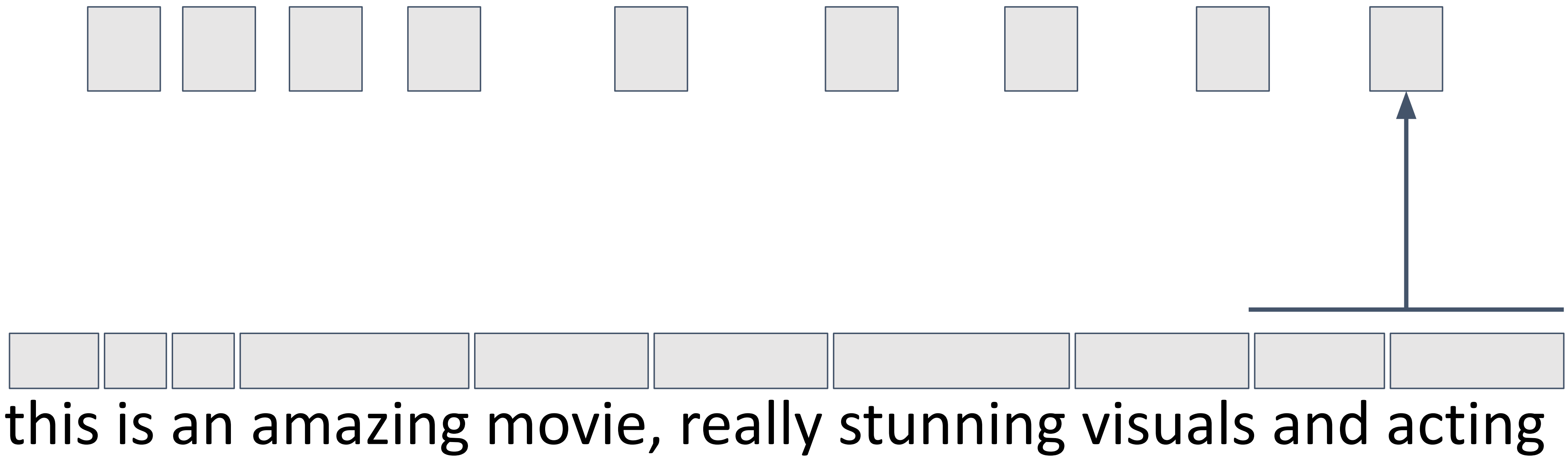

this is an amazing movie, really stunning visuals and acting

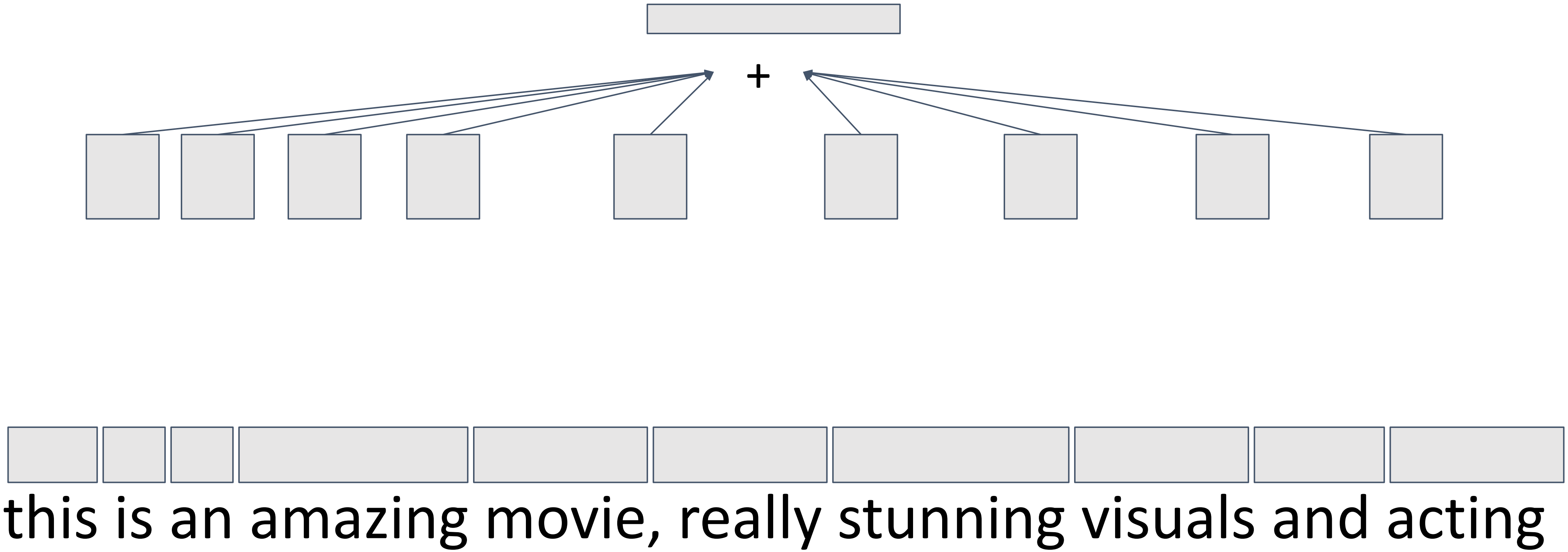

this is an amazing movie, really stunning visuals and acting

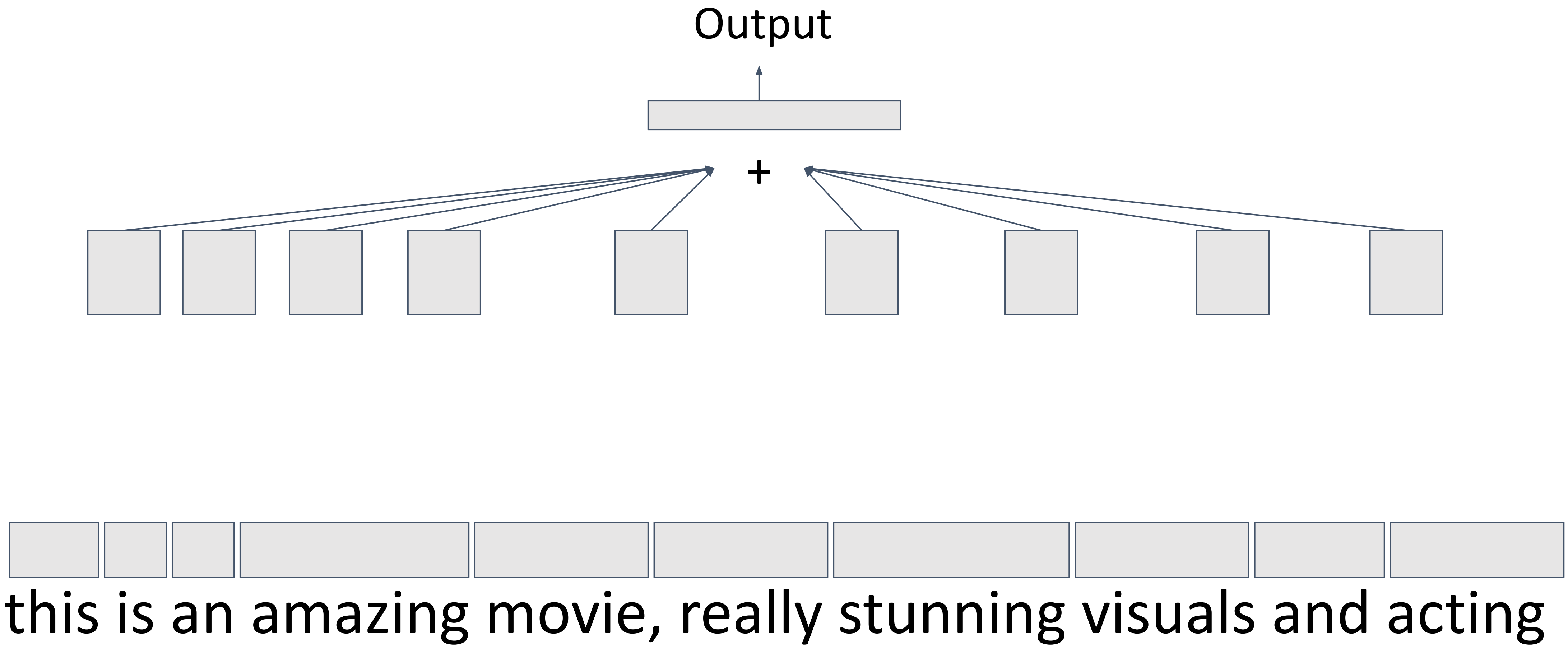

this is an amazing movie, really stunning visuals and acting

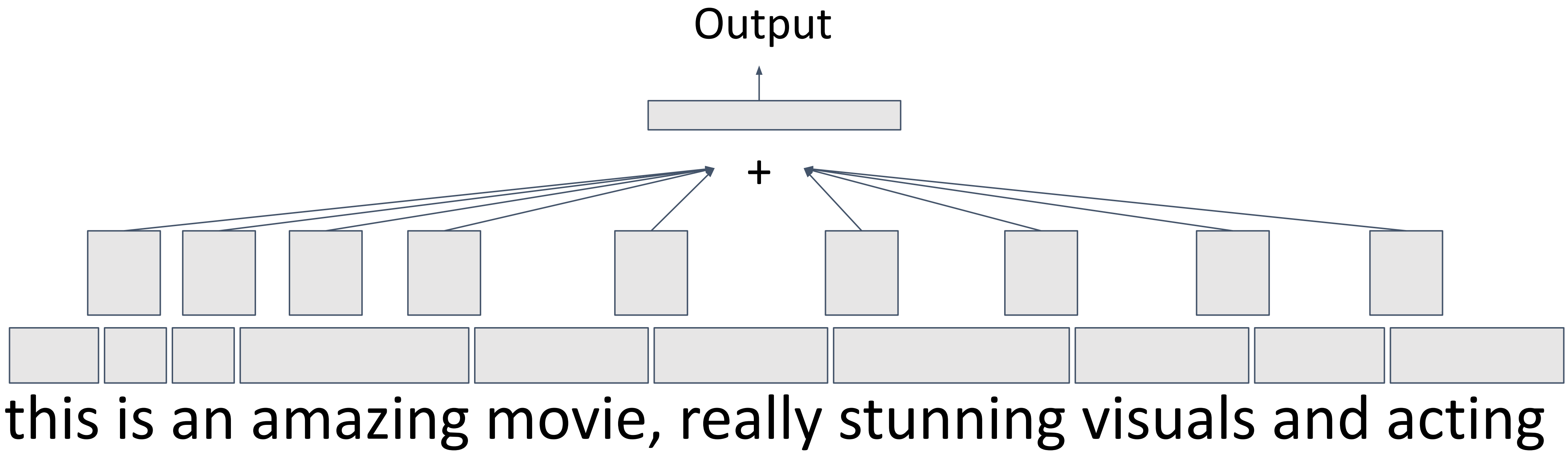

this is an amazing movie, really stunning visuals and acting

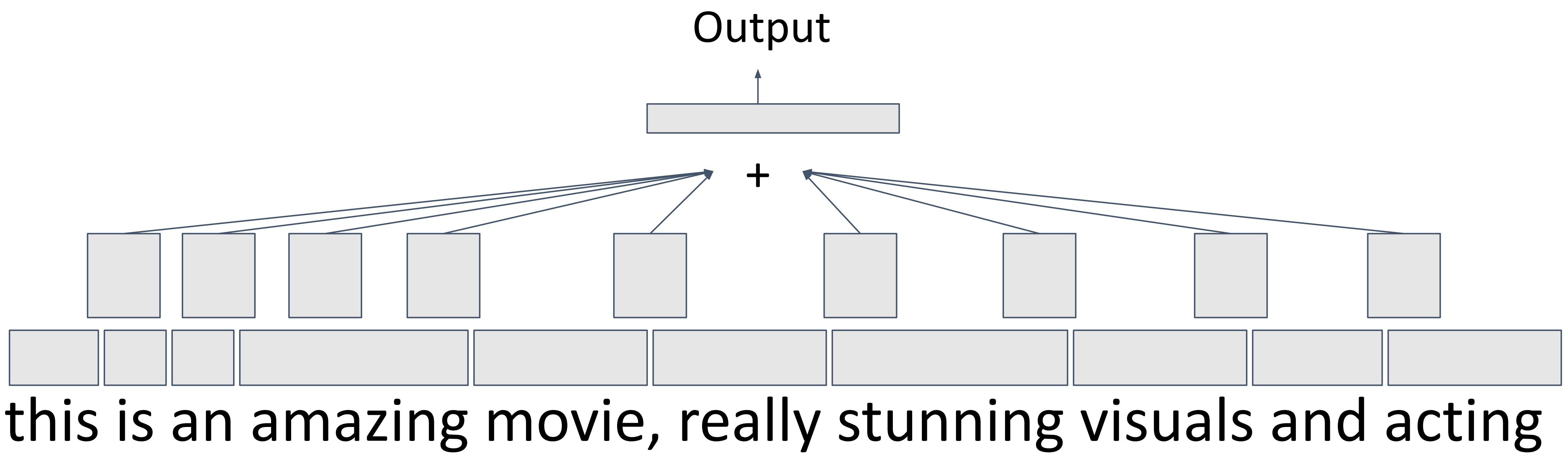

this is an amazing movie, really stunning visuals and acting


this is an amazing movie, really stunning visuals and acting

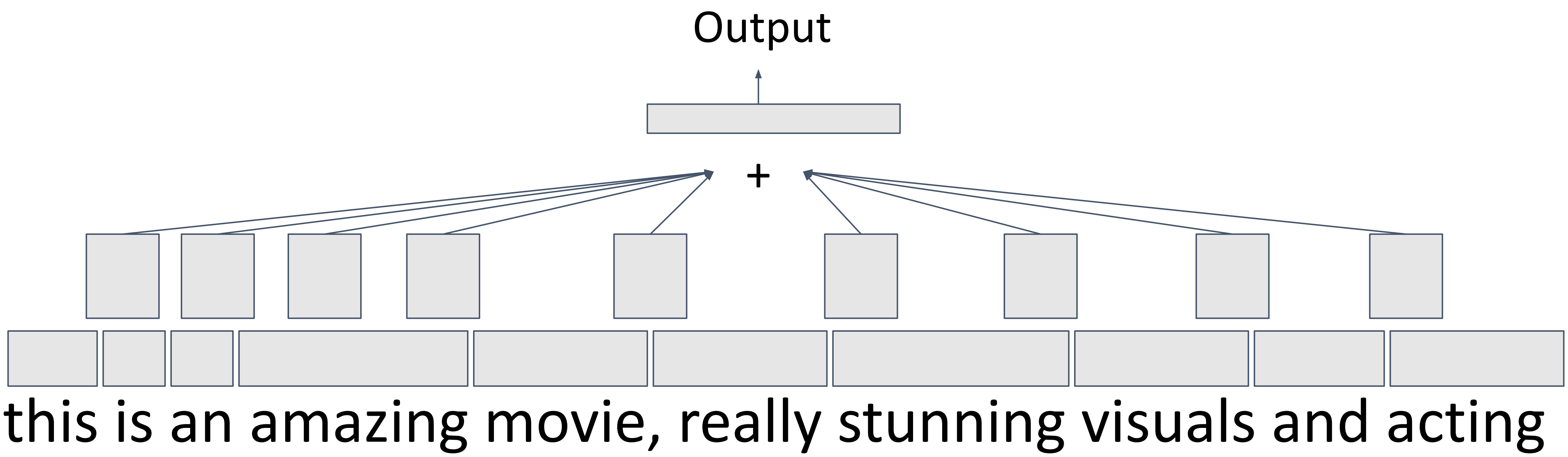

+
this is an amazing movie, really stunning visuals and acting


Output
+
this is an amazing movie, really stunning visuals and acting


Output
+
this is an amazing movie, really stunning visuals and acting

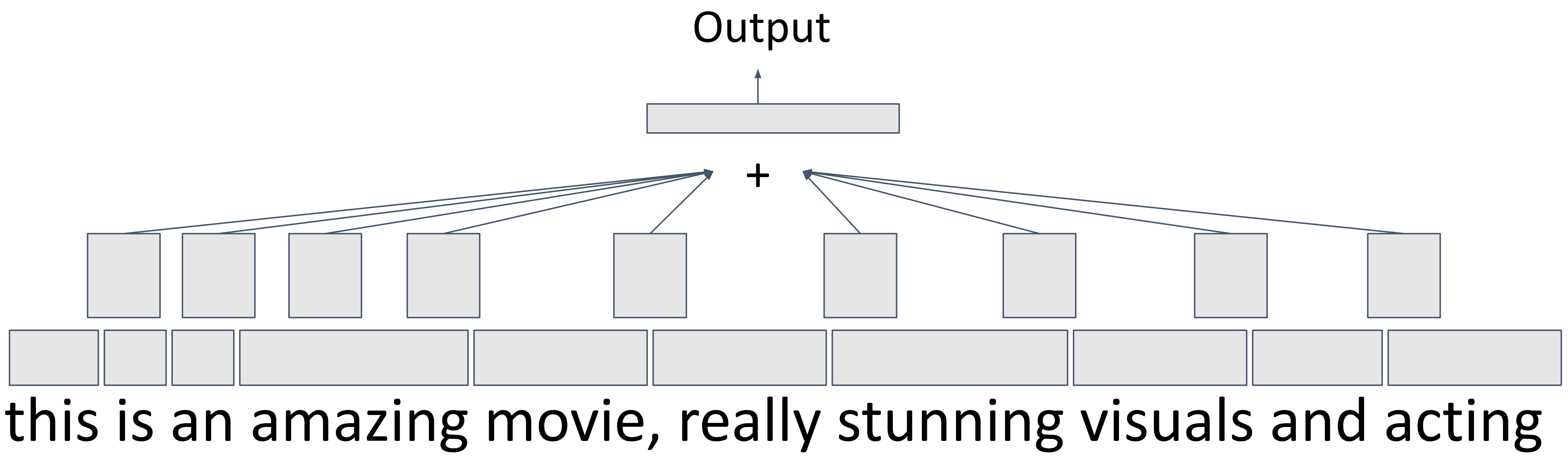


Averaging word embeddings to derive sentence embeddings (Wieting et al., 2015; Arora et al., 2017; Ethayarajh, 2018)



Averaging word embeddings to derive sentence embeddings (Wieting et al., 2015; Arora et al., 2017; Ethayarajh, 2018)

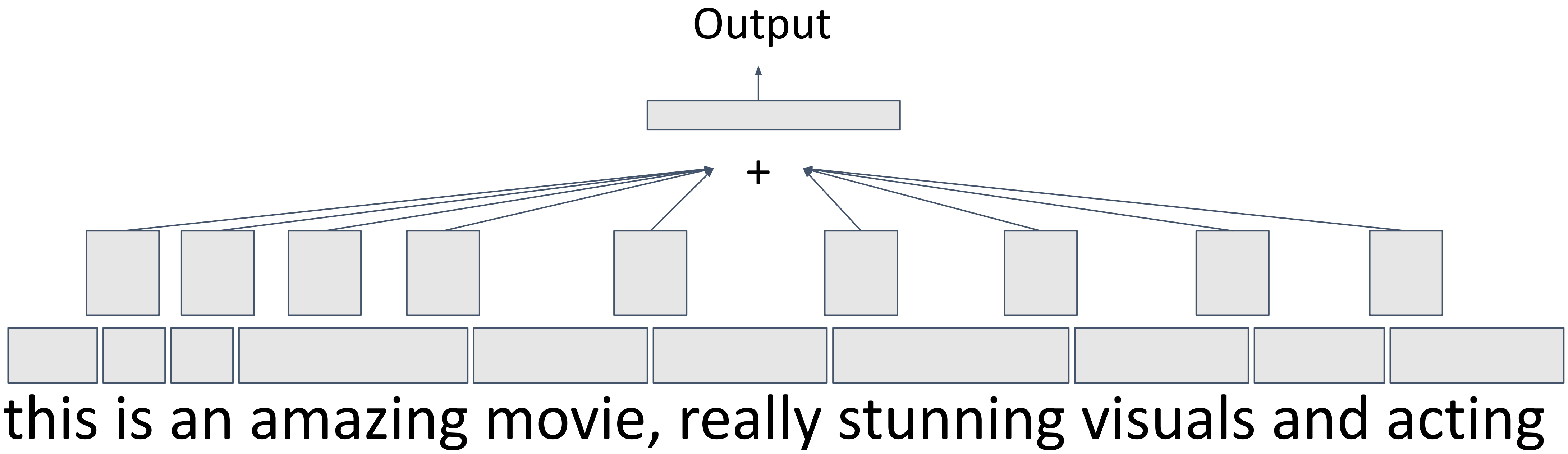
Convolutional Networks (Kim 2016) with Average Pooling



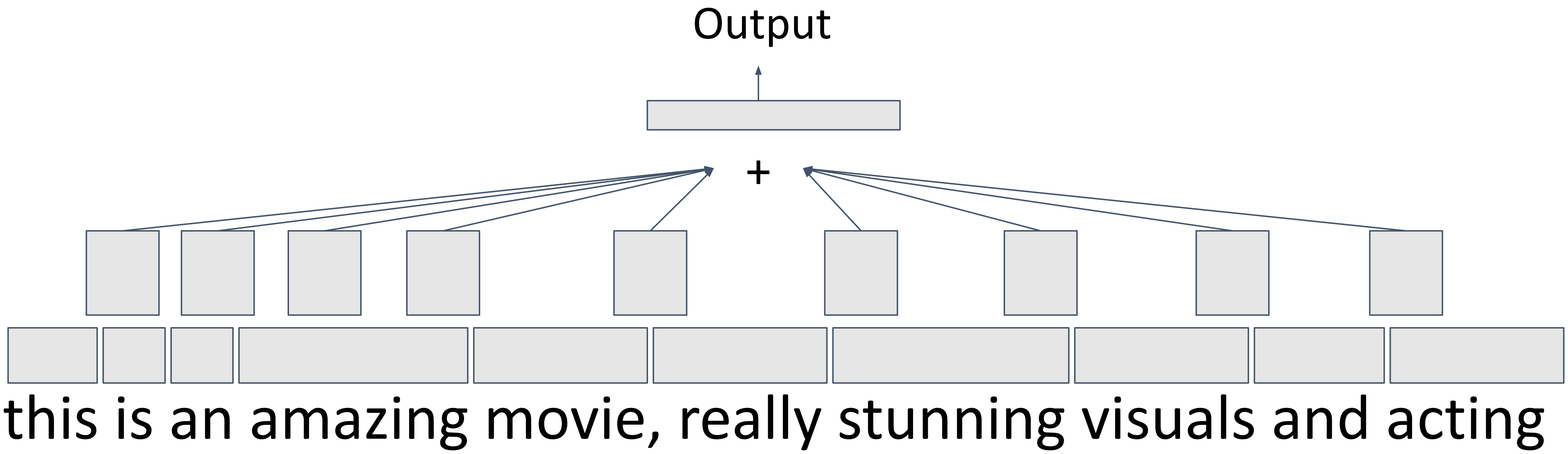
Averaging word embeddings to derive sentence embeddings (Wieting et al., 2015; Arora et al., 2017; Ethayarajh, 2018)

Convolutional Networks (Kim 2016) with Average Pooling

Log-linear models and SVMs using n-gram features

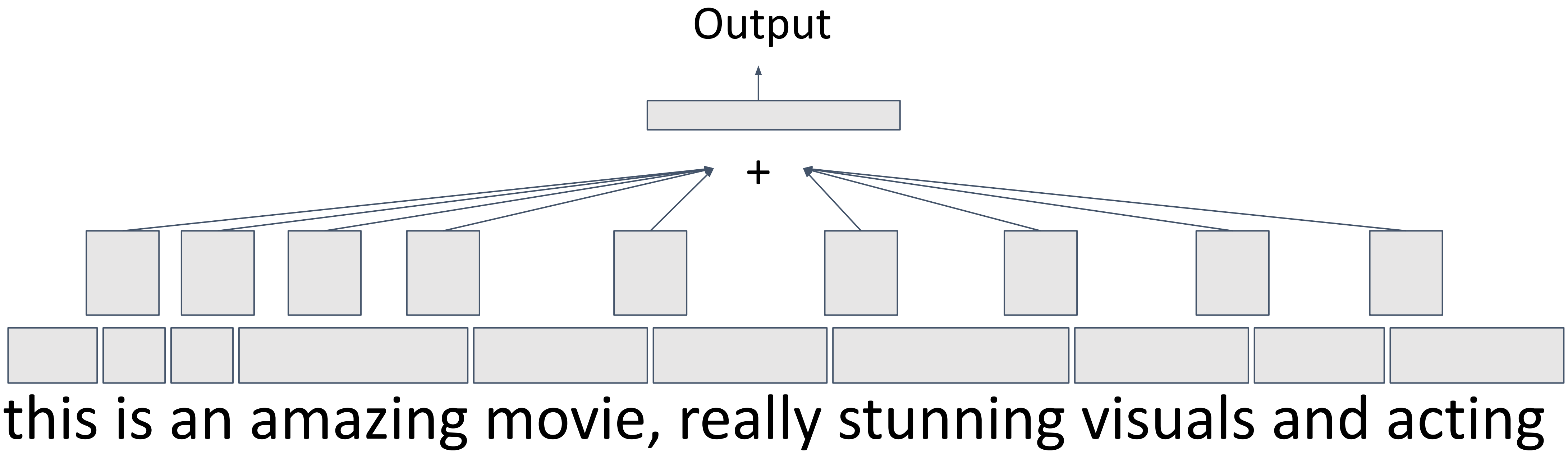


$$bs(f,x) \leq 2L^2C^2k^2$$



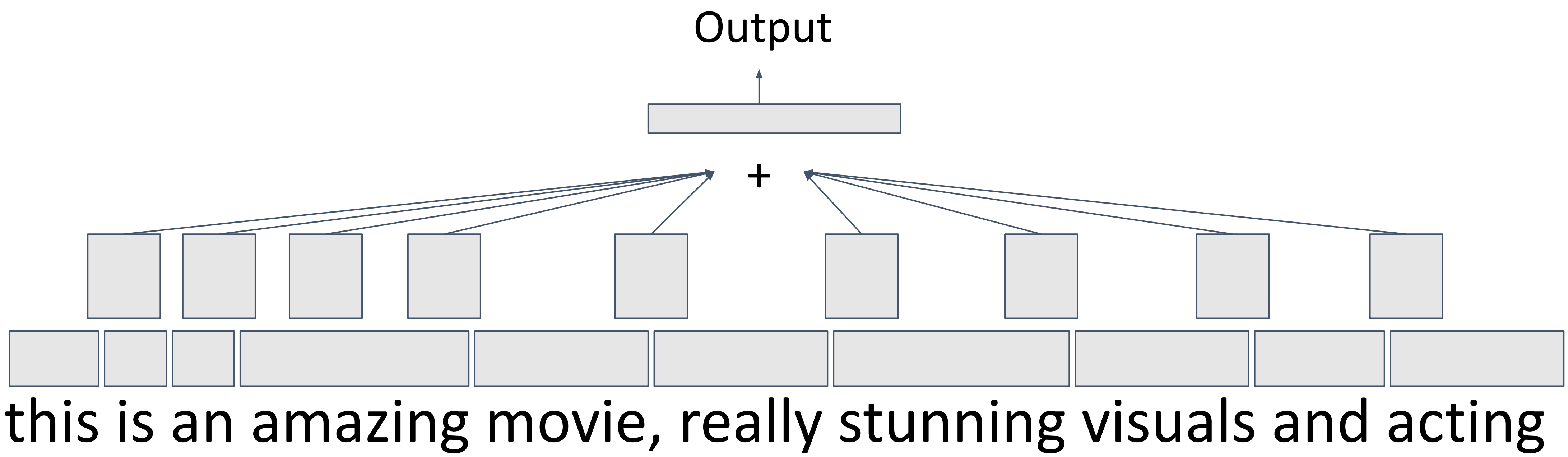
$$bs(f,x) \leq 2L^2C^2k^2$$

Lipschitz
constant of
output function



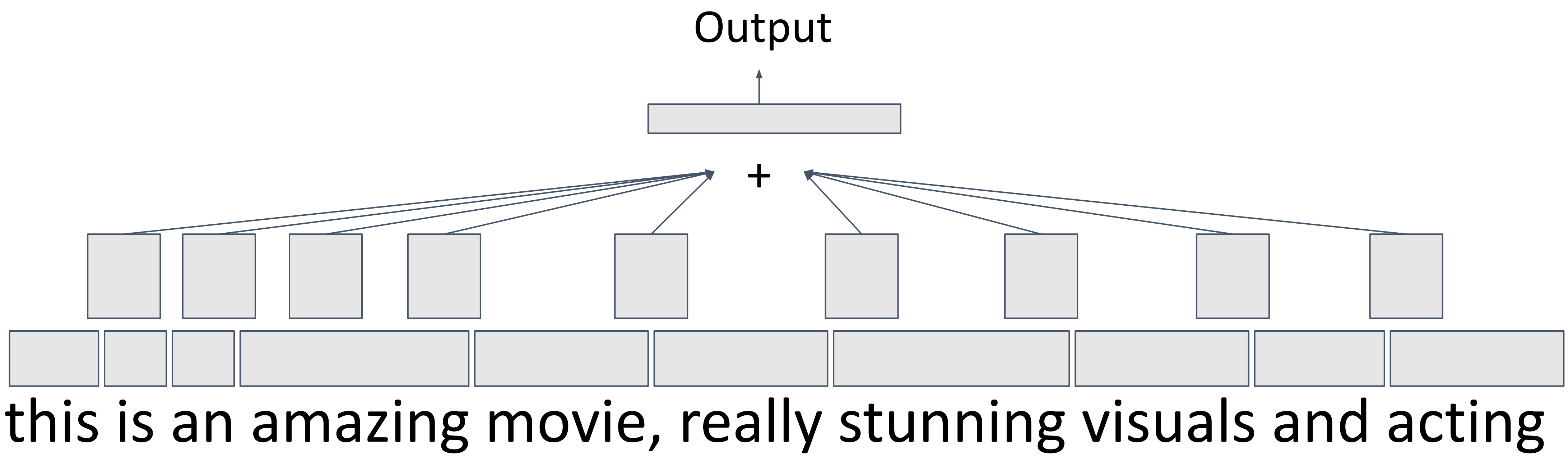
$$bs(f,x) \leq 2L^2C^2k^2$$

Norm of vectors



$$bs(f,x) \leq 2L^2C^2k^2$$

Window width



$$bs(f,x) \le 2L^2C^2k^2$$

independent of the input length!

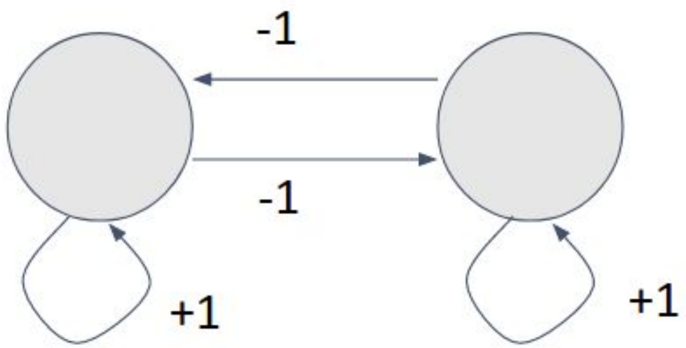
	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...		
LSTM		
Transformer		

	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...	No	
LSTM		
Transformer		

	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...	No	Strict Bound
LSTM		
Transformer		

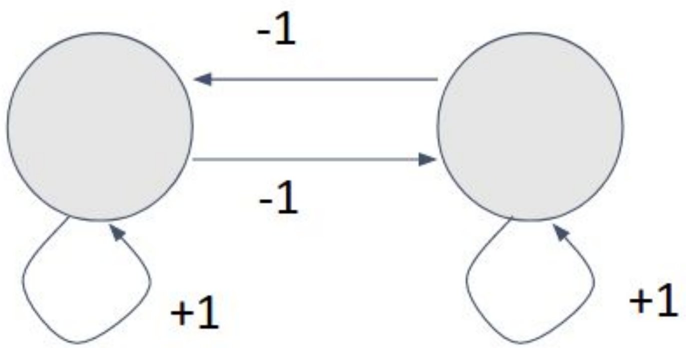
LSTMs and Transformers

RNNs and LSTMs can express PARITY because they express all regular languages (Horne and Hush, 1994)



LSTMs and Transformers

RNNs and LSTMs can express PARITY because they express all regular languages (Horne and Hush, 1994)



Transformers cannot express PARITY generalizably across input lengths

Transformers and PARITY

Proof Idea:

Assume we have a candidate transformer given.

For proof, see: Hahn (2020, TACL)

Transformers and PARITY

Proof Idea:

Assume we have a candidate transformer given.

We construct a pair of inputs that are classified the same, even though their parity differs

For proof, see: Hahn (2020, TACL)

Transformers and PARITY

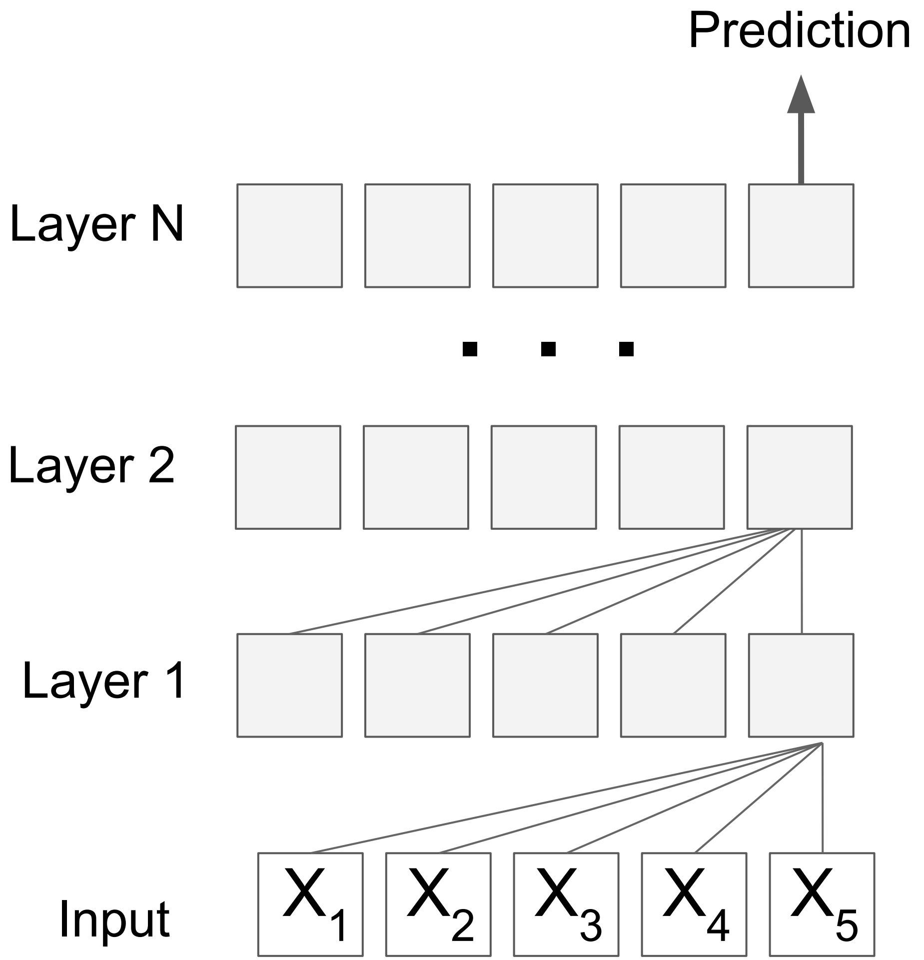
Proof Idea:

Assume we have a candidate transformer given.

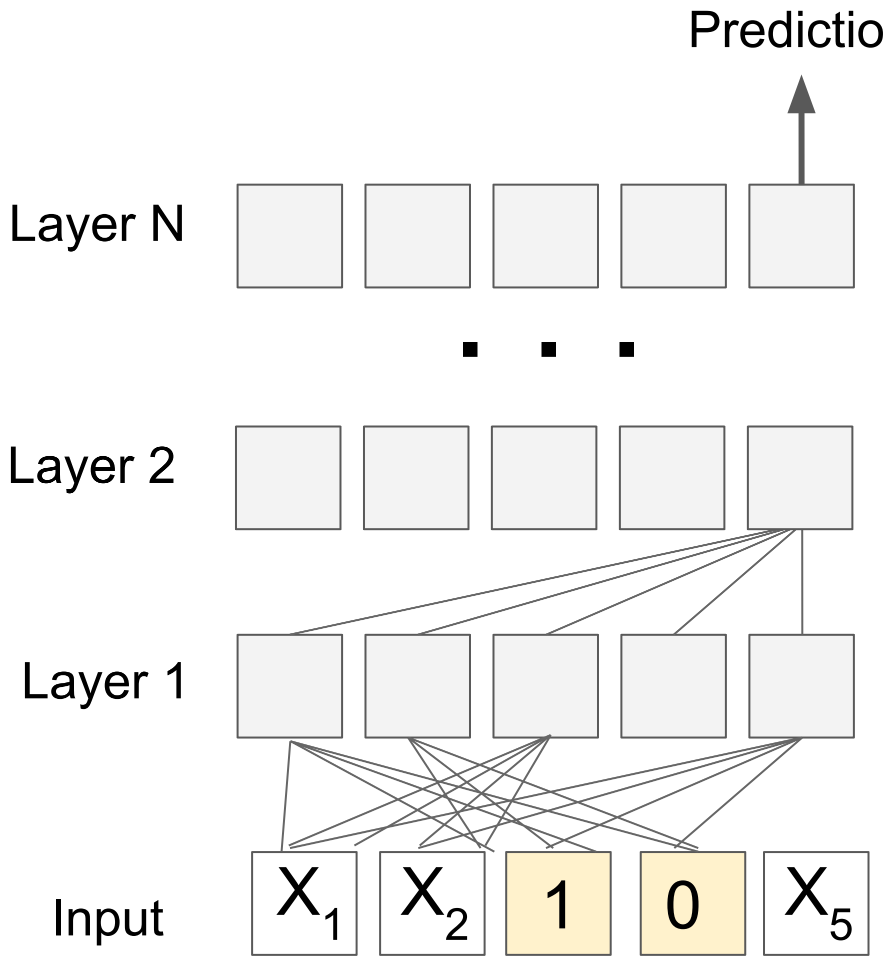
We construct a pair of inputs that are classified the same, even though their parity differs

Method: We fix a few input bits to ‘distract’ the transformer, so that it ignores most input bits.

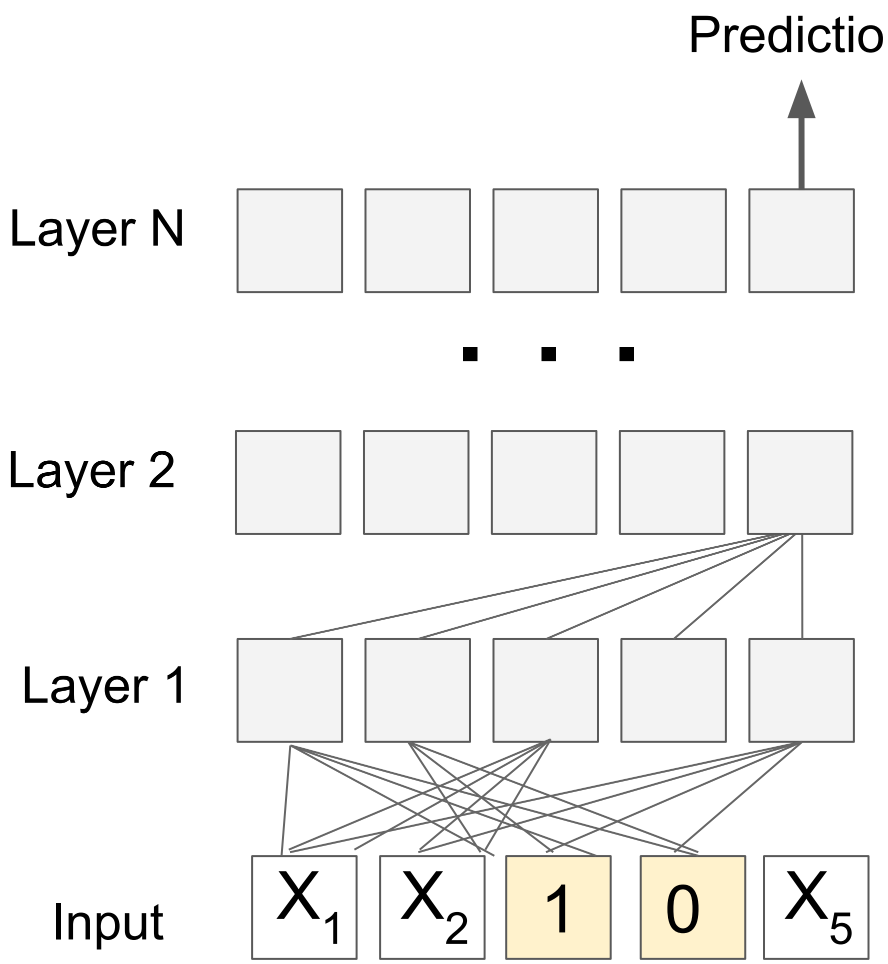
For proof, see: Hahn (2020, TACL)



For proof, see: Hahn (2020, TACL)

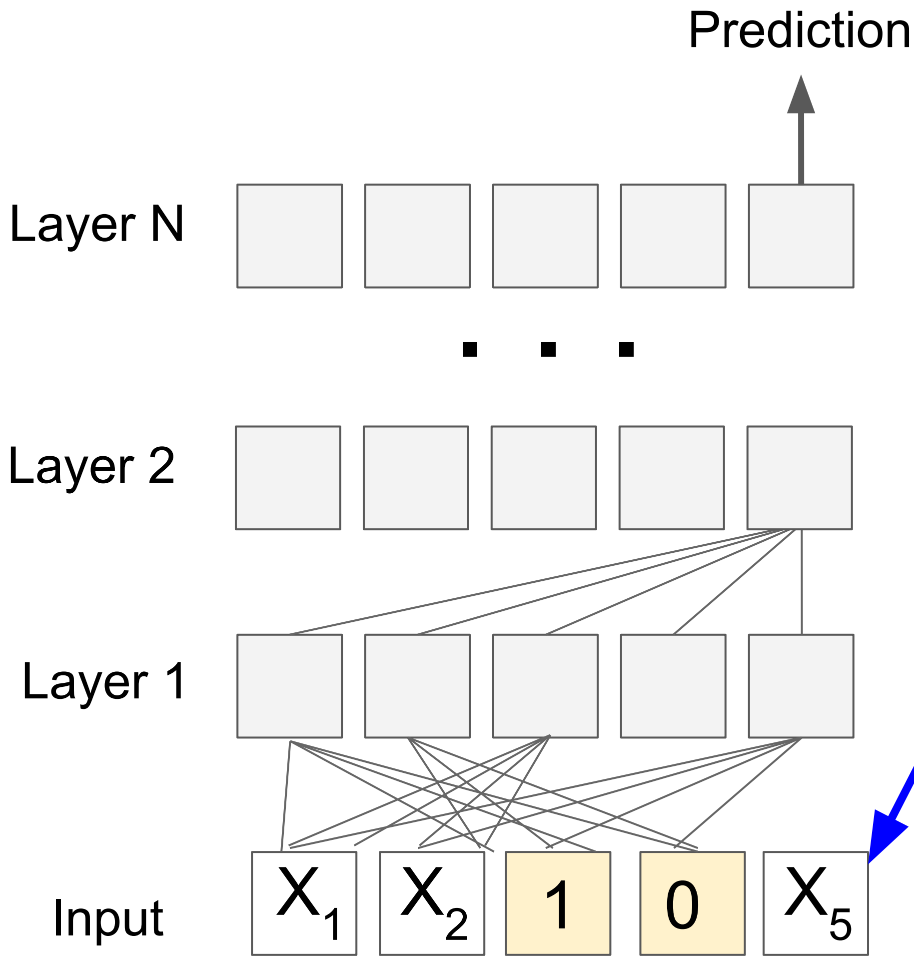


For proof, see: Hahn (2020, TACL)



Idea: Some input bits will be ignored, since their attention weights are always smaller than those of the fixed bits.

For proof, see: Hahn (2020, TACL)

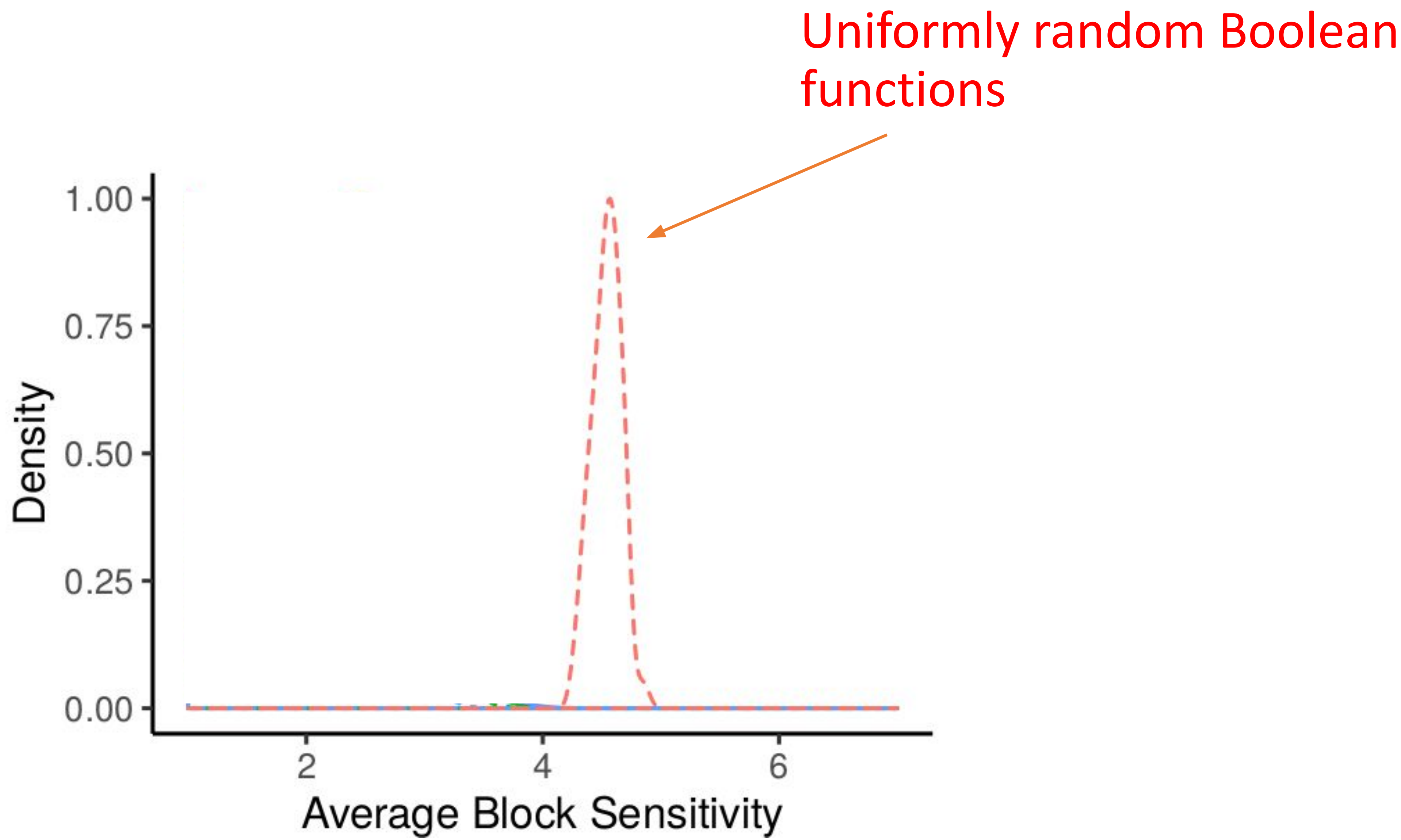


The entire network ignores this bit!

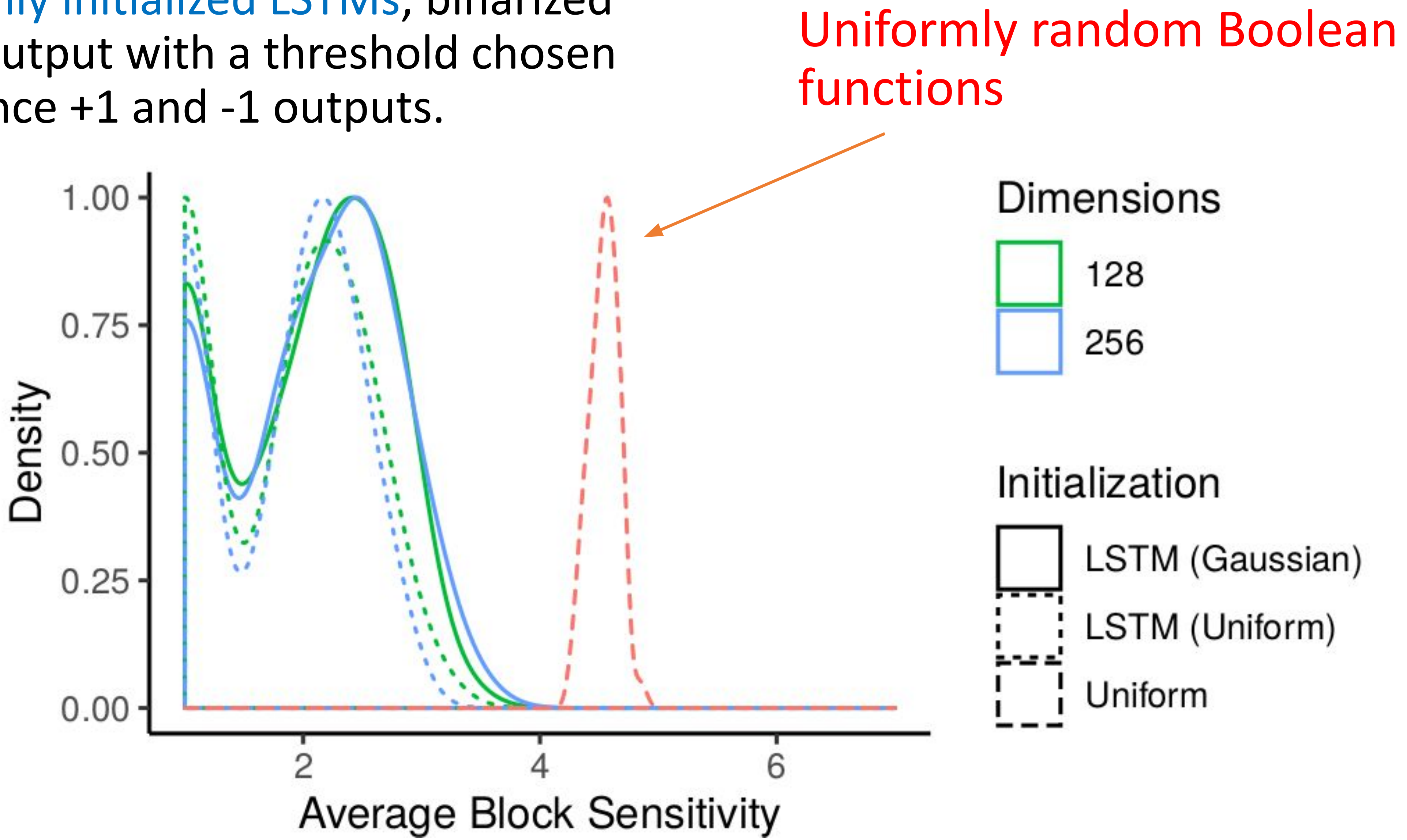
Consequence: It could not have modeled PARITY, since every bit matters for PARITY.

For proof, see: Hahn (2020, TACL)

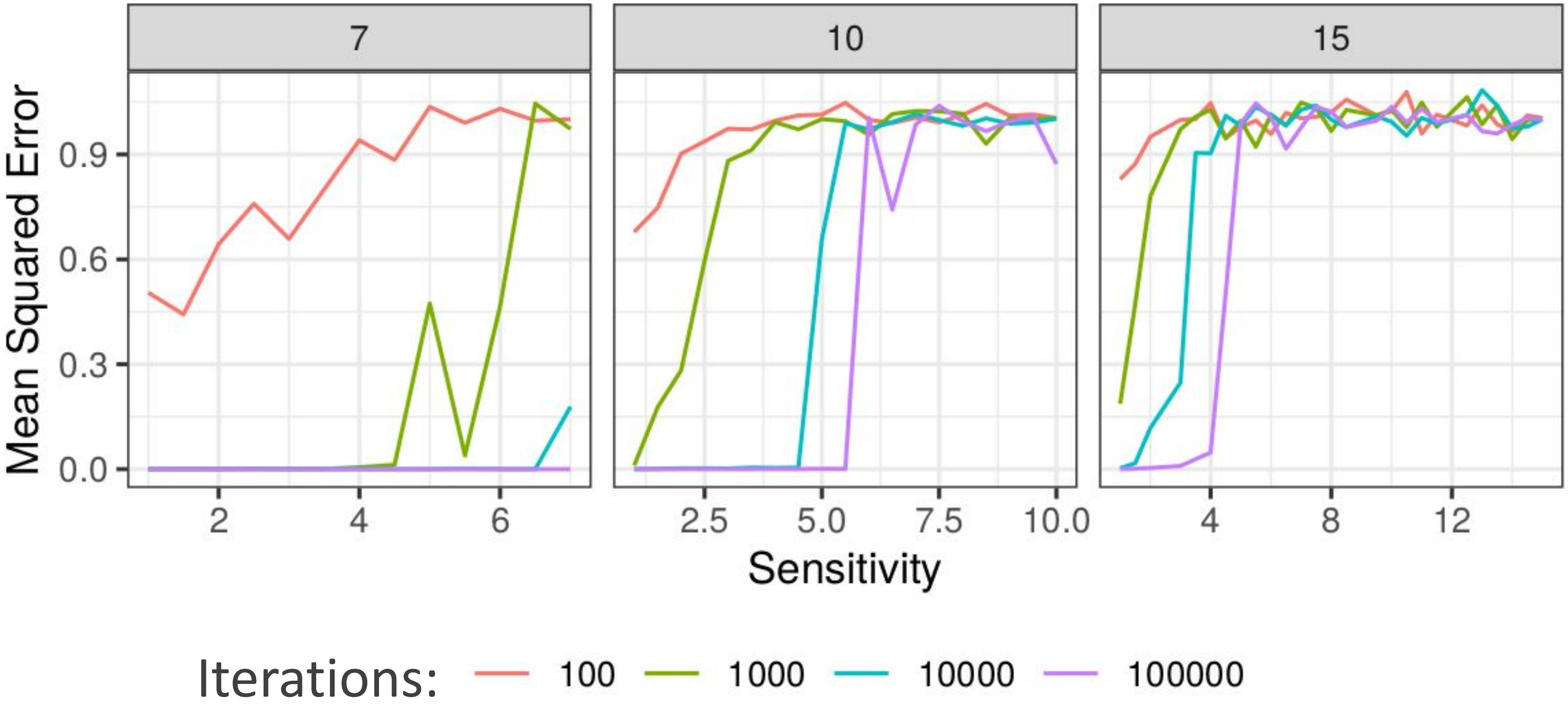
	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...	No	Strict Bound
LSTM	Yes	
Transformer	No	


Uniformly random Boolean functions
1.00
0.75
0.50
0.25
0.00
Density
2
4
6
Average Block Sensitivity

Randomly initialized LSTMs, binarized scalar output with a threshold chosen to balance +1 and -1 outputs.



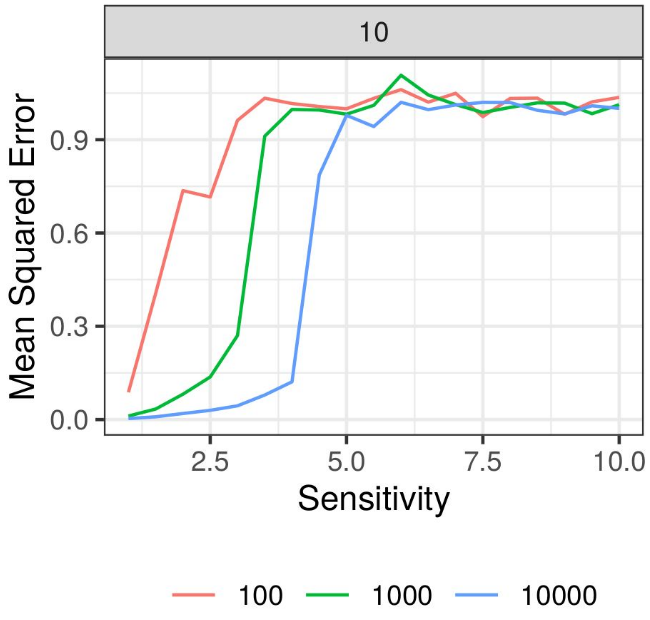
Low-sensitivity functions are more learnable



- LSTM with 128 units, Adam (lr 0.003, batch size 32)
- No train/test split – this tests fitting ability, not generalization.

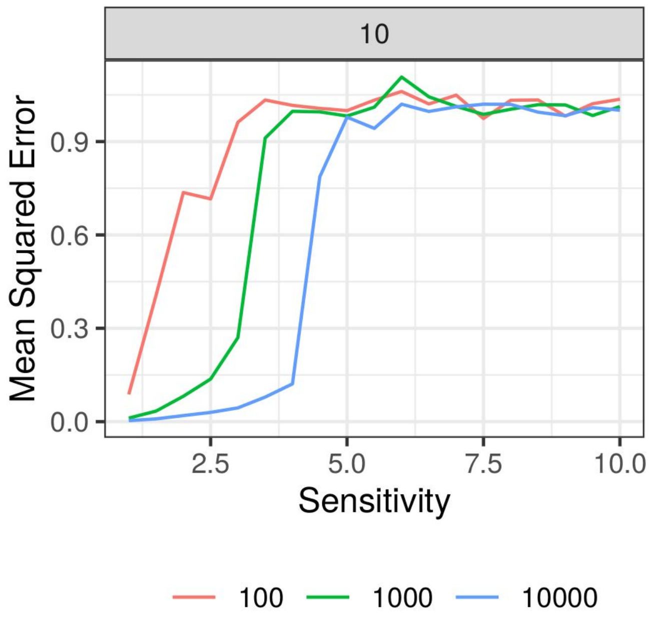
Low-sensitivity functions are more learnable

With a transformer
(4 layers, 4 heads, 32 units)

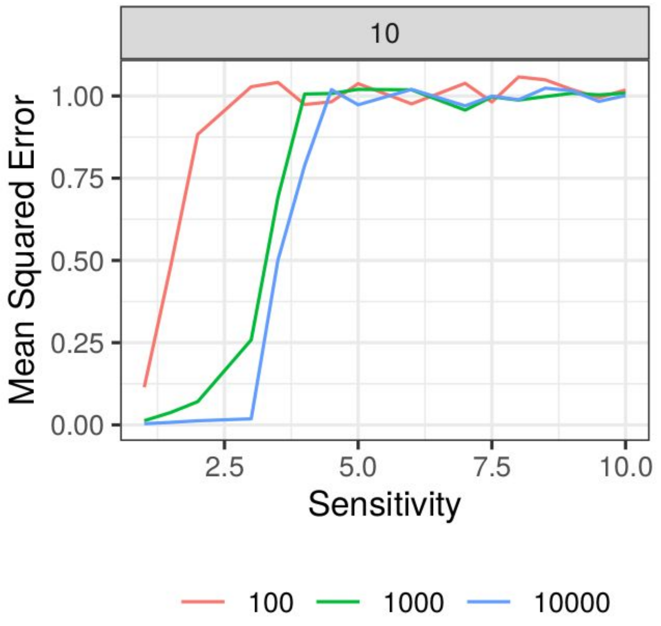


Low-sensitivity functions are more learnable

With a transformer
(4 layers, 4 heads, 32 units)



8 layers

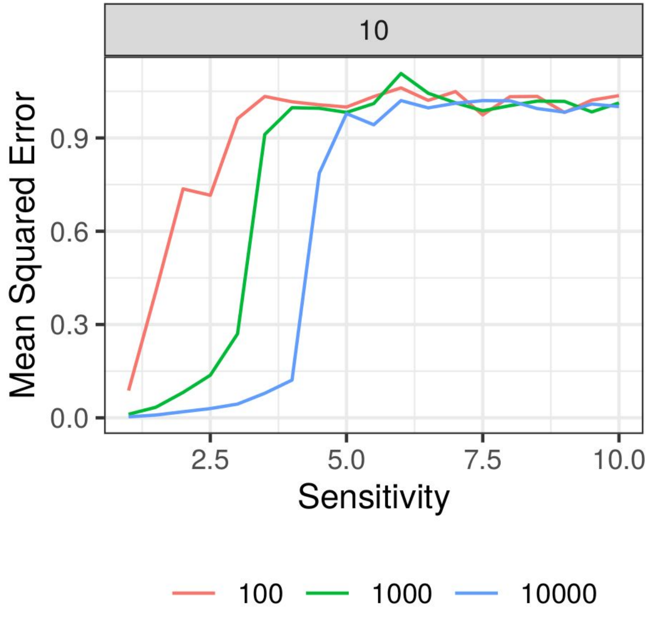


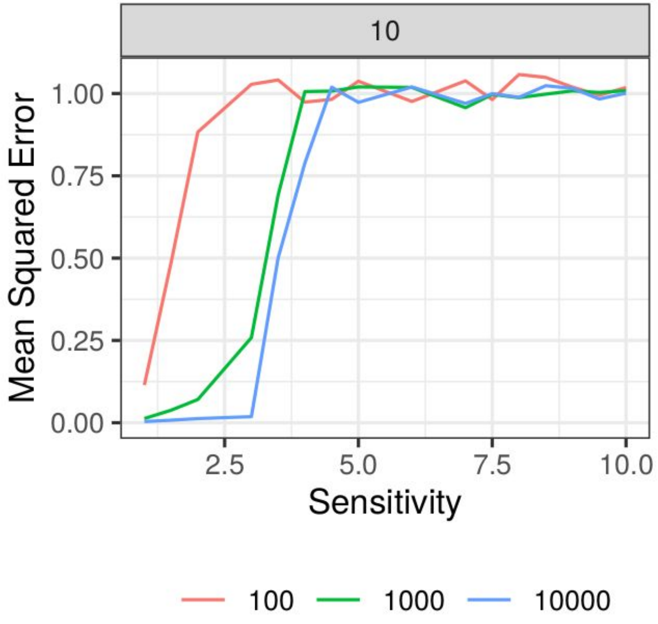
Low-sensitivity functions are more learnable

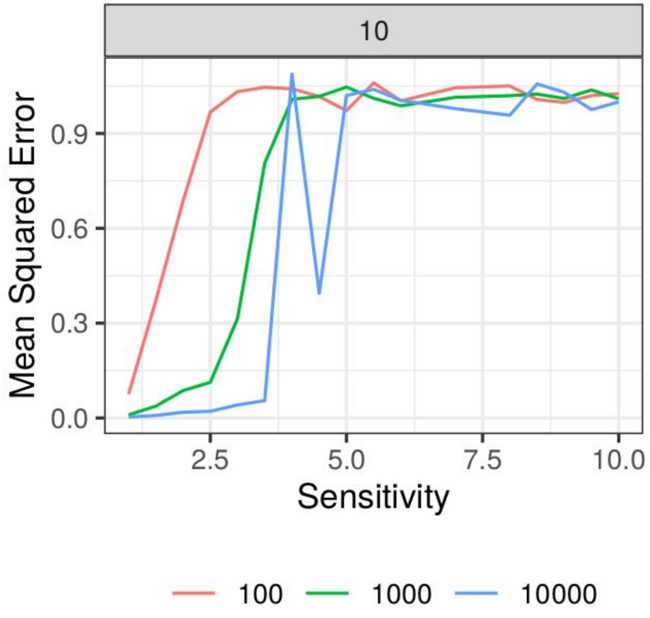
With a transformer (4 layers, 4 heads, 32 units)

8 layers

512 units







	Can represent f_{PARITY}?	Can practically learn high-sensitivity functions?
Lexical Classifier, CNN, ...	No	Strict Bound
LSTM	Yes	Hard
Transformer	No	Hard

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

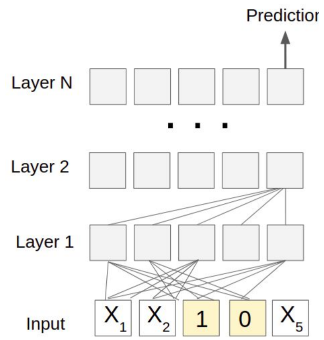
Sensitivity for Sequence Classification

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

Sensitivity can be used for rigorous analysis of the power of ML models

$$bs(f,x) \leq 2L^2C^2k^2$$



Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

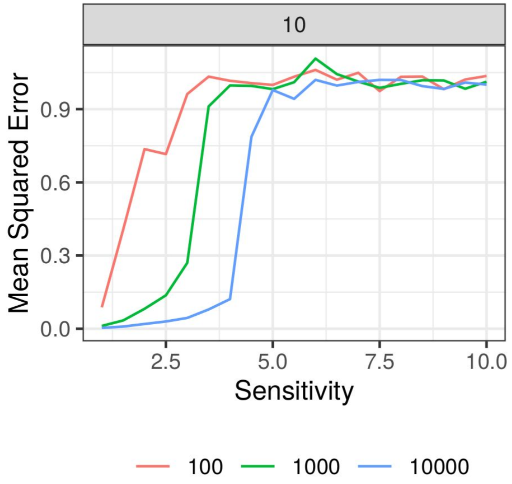
Sensitivity for Sequence Classification

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

Sensitivity can be used for rigorous analysis of the power of ML models

Sensitivity predicts learnability



Sensitivity and Difficulty of NLP Tasks

Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Estimating Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Estimating Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

$$s(f,x,P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

Estimating Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

$$s(f,x,P) := \text{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

Estimate f using a strong model of the task.

Estimating Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

$$s(f,x,P) := \mathrm{Var}\left(f(X) | X \in x^{\oplus P}\right)$$

Inputs that agree with x outside of positions in P.

Sampled using XLNet (Yang et al 2019) and u-PMLM (Liao et al 2020).

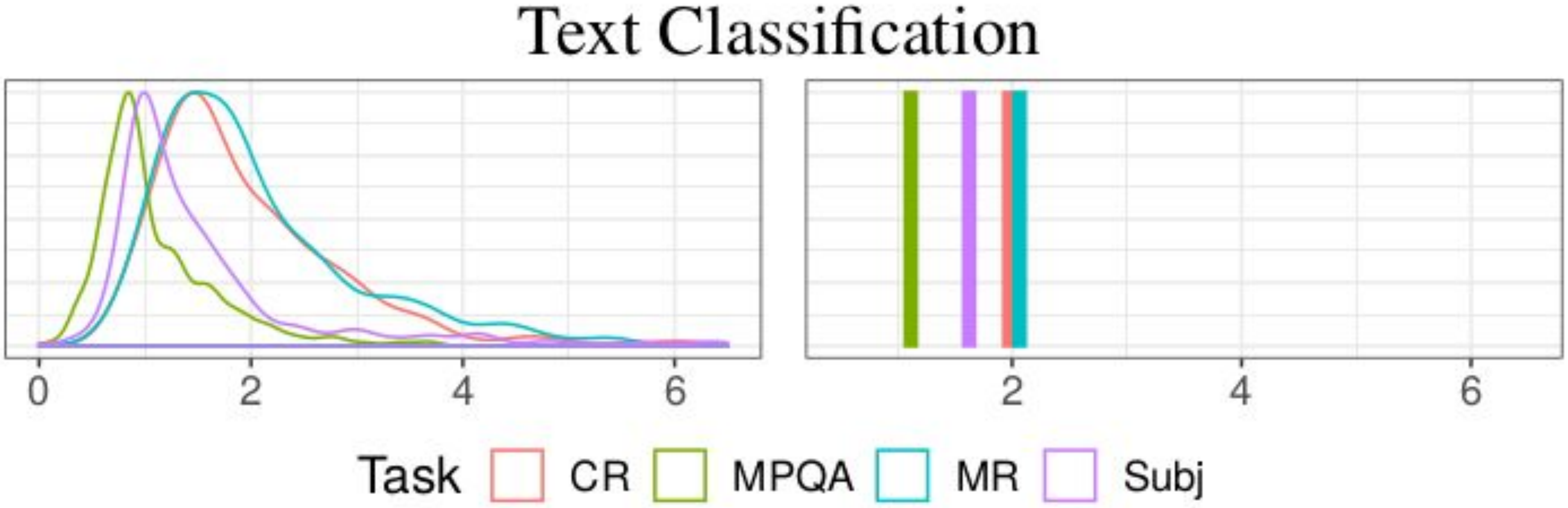
Estimating Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

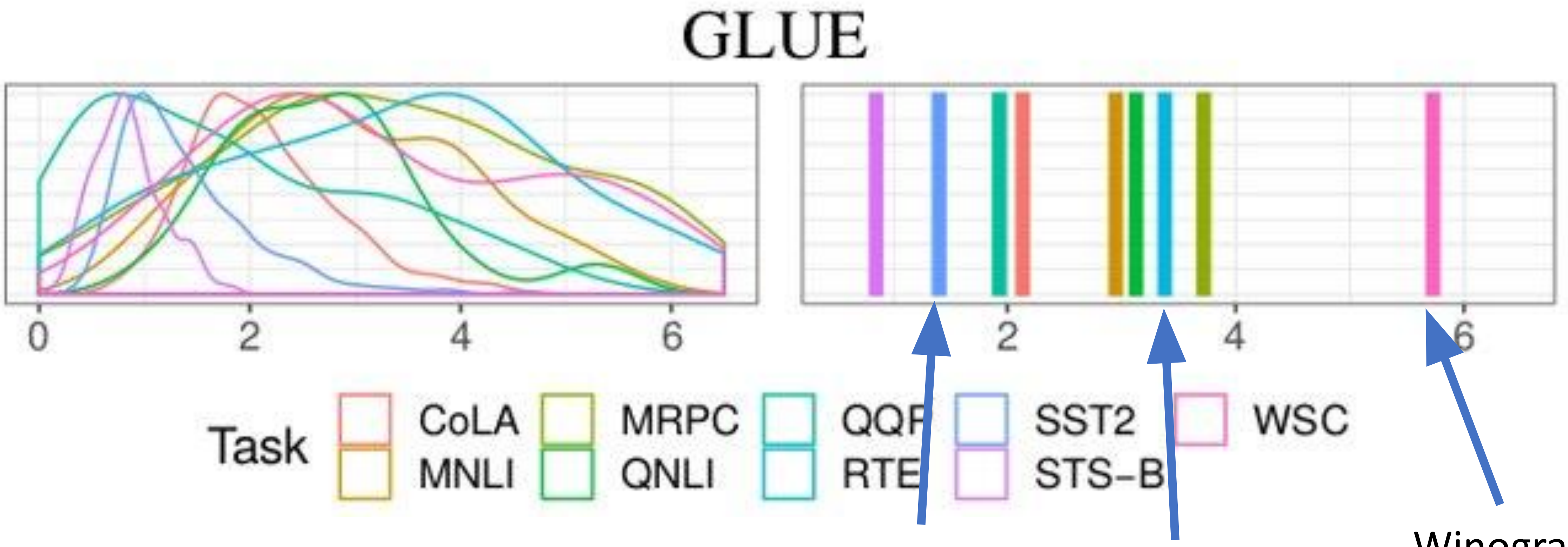
Exponential number of subsets!

Restrict to polynomial number of subsets

Results in lower bound



- CR, MR: review sentiment (Hu and Liu, 2004; Pang and Lee, 2005)
- MPQA: question type (Wiebe et al., 2005)
- Subj: subjectivity (Pang and Lee, 2005)
- *f* estimated using finetuned RoBERTa (Liu et al., 2019)



Stanford Sentiment Treebank (Socher et al 2013)

Recognizing Textual Entailment (Dagan et al 2009)

Winograd Schema Challenge (Levesque et al 2012)

- GLUE challenge suite (Wang et al., 2019)
- f estimated using finetuned RoBERTa

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

Hypothesis:

Steve Jobs worked for Apple.

From Recognizing Textual Entailment (GLUE)

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

Hypothesis:

Steve Jobs worked for Apple.

From Recognizing Textual Entailment (GLUE)

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
2. Steve Jobs was attacked by Sculley and the other executives [...] and resigned from the company a few weeks later.

Hypothesis:

Steve Jobs worked for Apple.

From Recognizing Textual Entailment (GLUE)

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
2. Steve Jobs was attacked by Sculley and the other executives [...] and resigned from the company a few weeks later.

Hypothesis:

Steve Jobs worked for Apple.

3. Jobs later worked for Apple

From Recognizing Textual Entailment (GLUE)

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
2. Steve Jobs was attacked by Sculley and the other executives [...] and resigned from the company a few weeks later.

Hypothesis:

Steve Jobs worked for Apple.

3. Jobs later worked for Apple
4. Steve Jobs returned to Apple

From Recognizing Textual Entailment (GLUE)

Premise:

Steve Jobs was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.

1. Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
2. Steve Jobs was attacked by Sculley and the other executives [...] and resigned from the company a few weeks later.

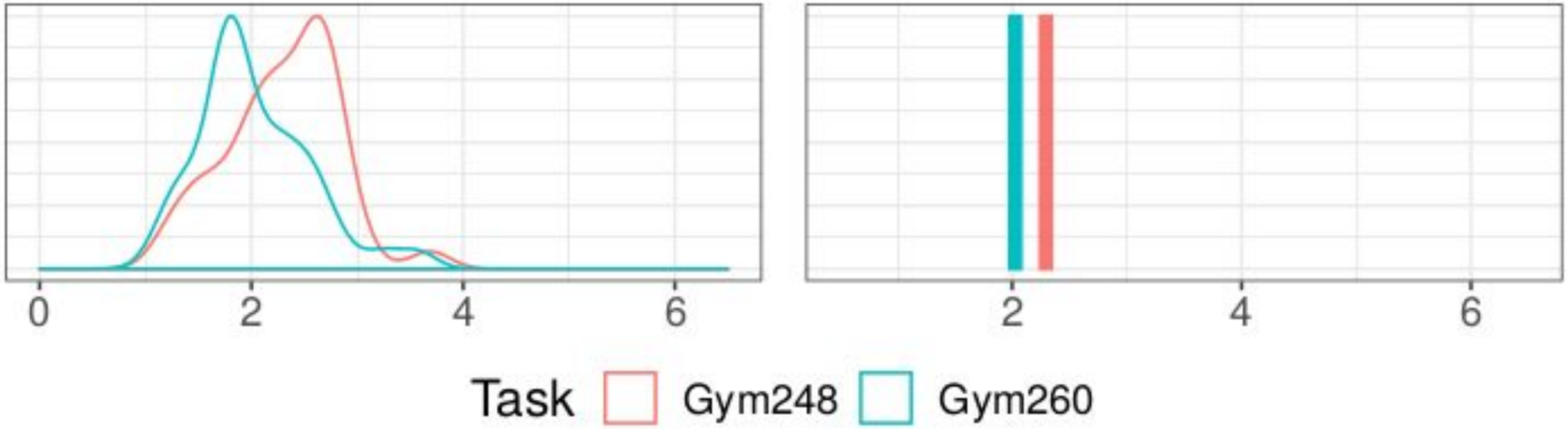
Hypothesis:

Steve Jobs worked for Apple.

3. Jobs later worked for Apple
4. Steve Jobs returned to Apple
5. Steve Jobs worked for Google

From Recognizing Textual Entailment (GLUE)

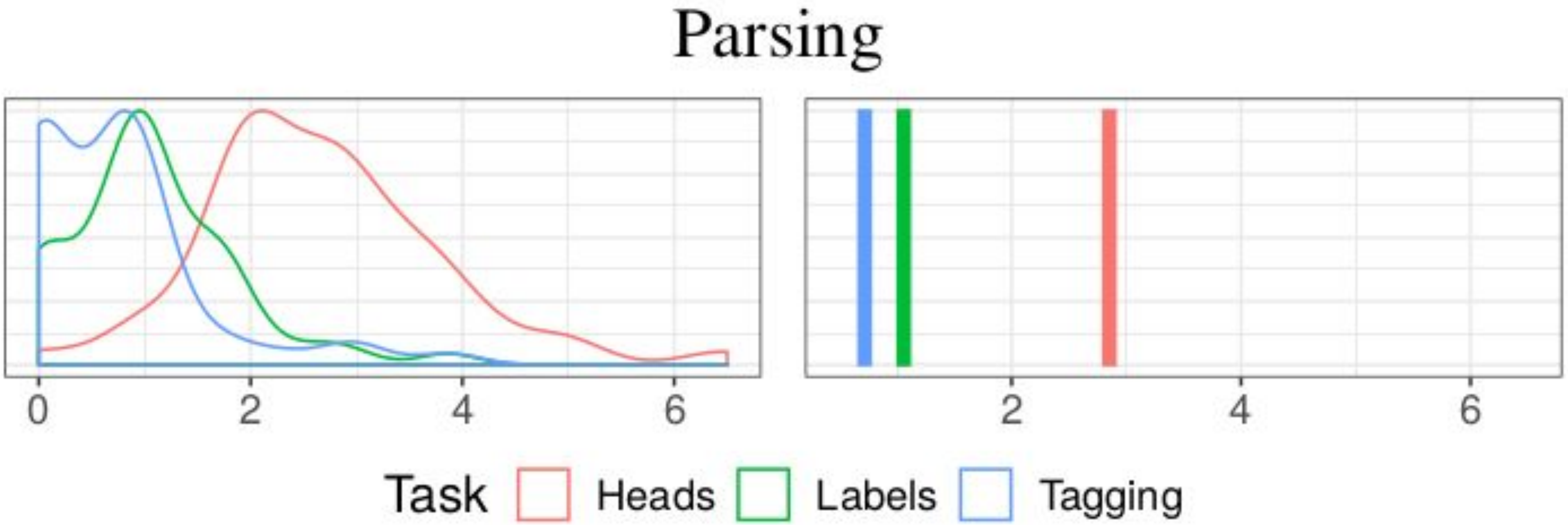
Syntax



- Anaphor licensing (Marvin and Linzen, 2018; Hu et al., 2020)

"The author next to the senators hurt {himself, themselves}."
"The author that liked the senators hurt {himself, themselves}."

- Estimated using medium-size GPT-2

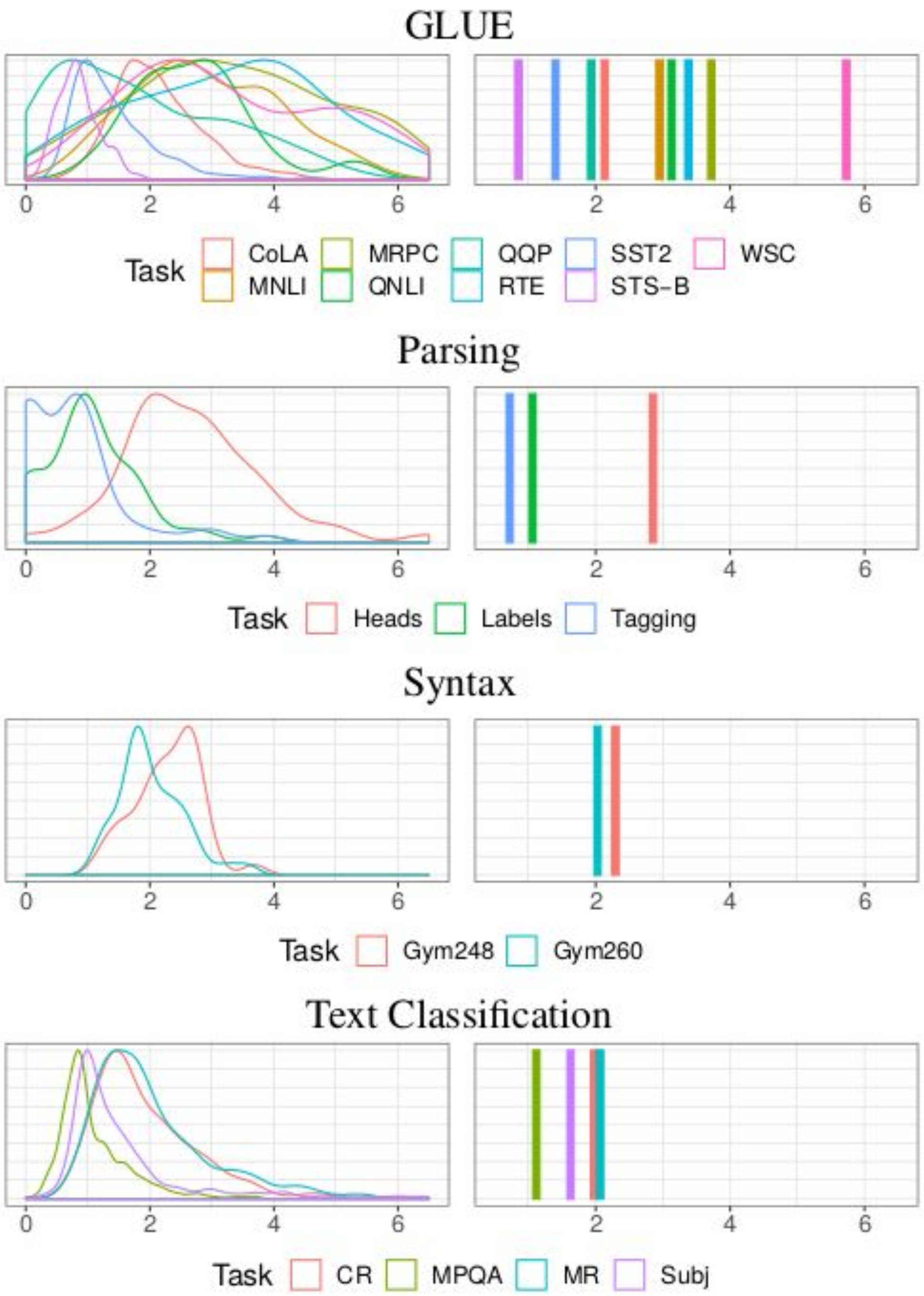


Identifying the relative position of the head (an integer)

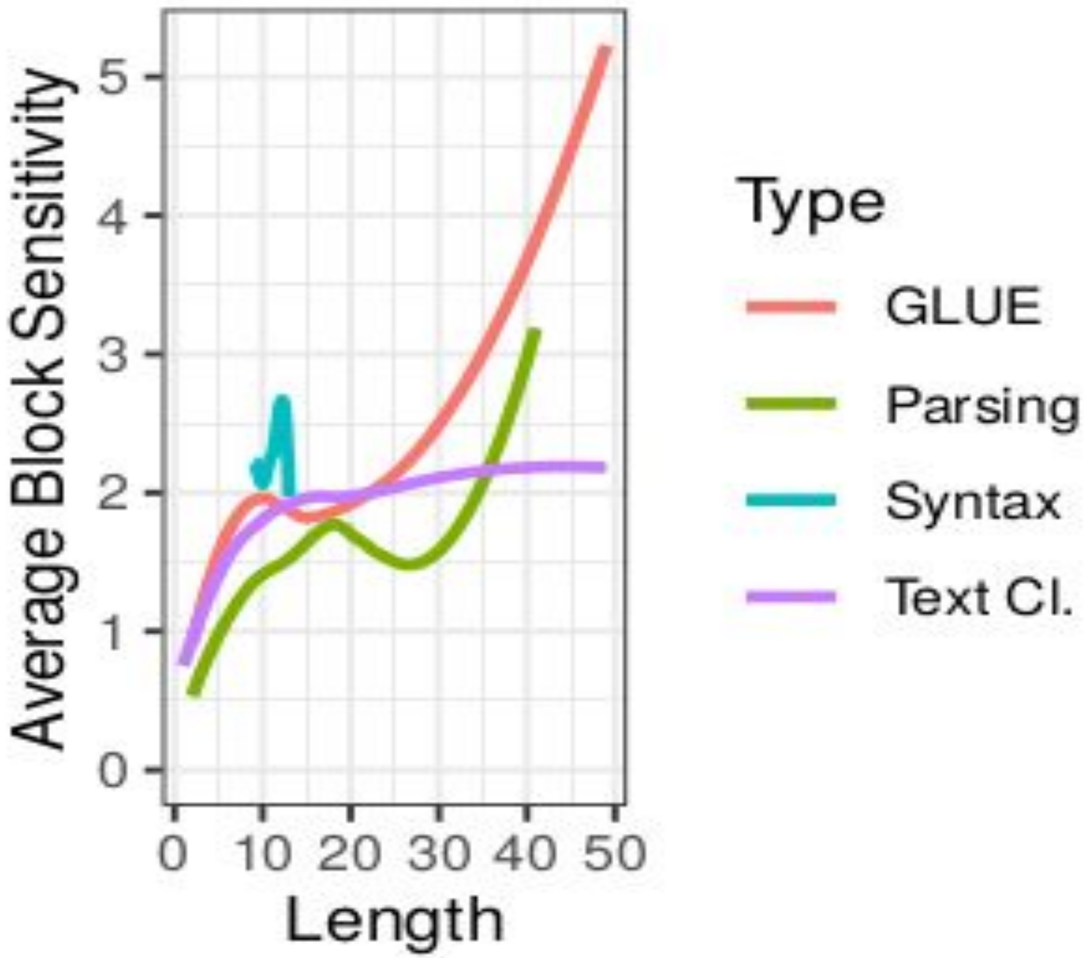
Identify the dependency label

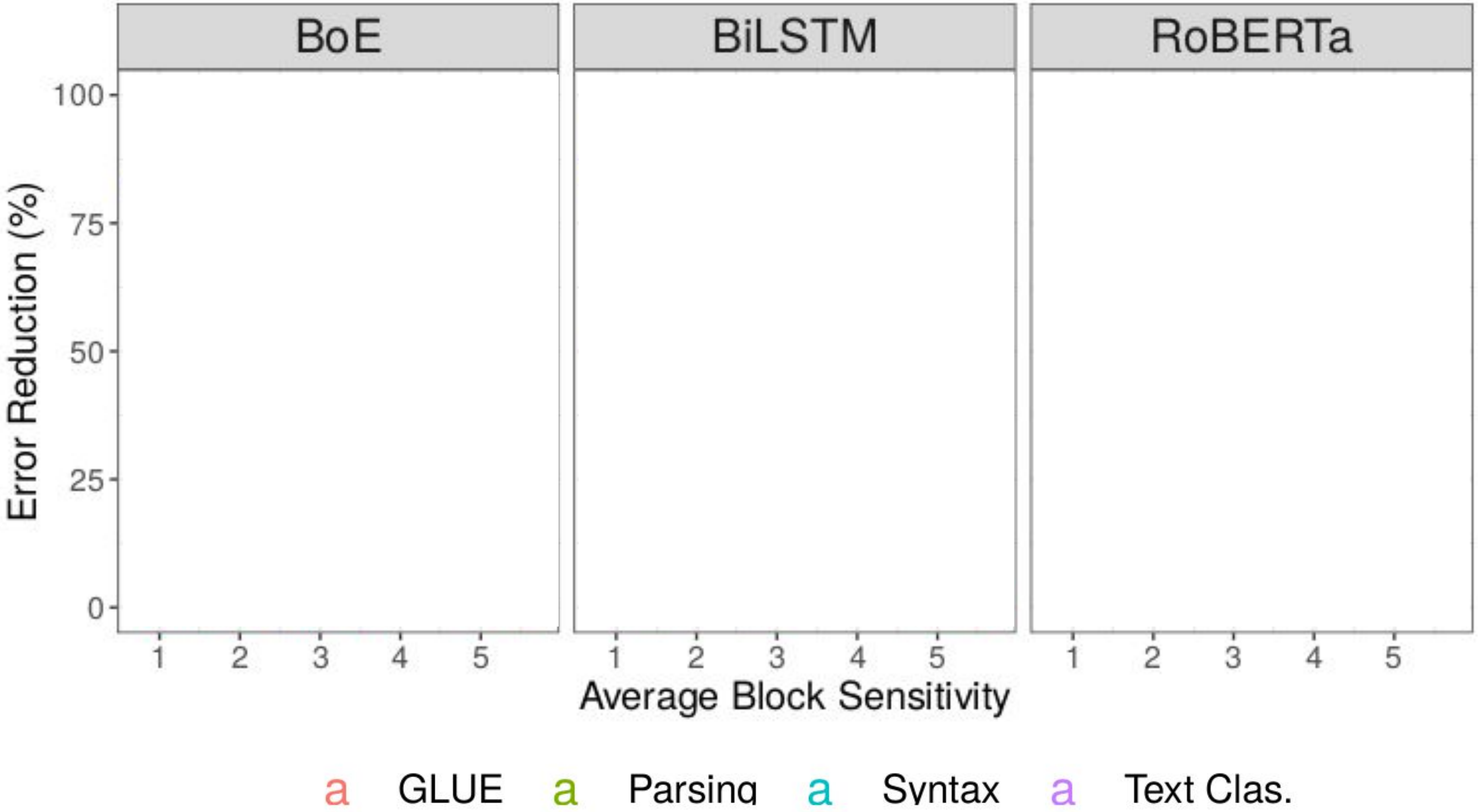
Identify POS tag

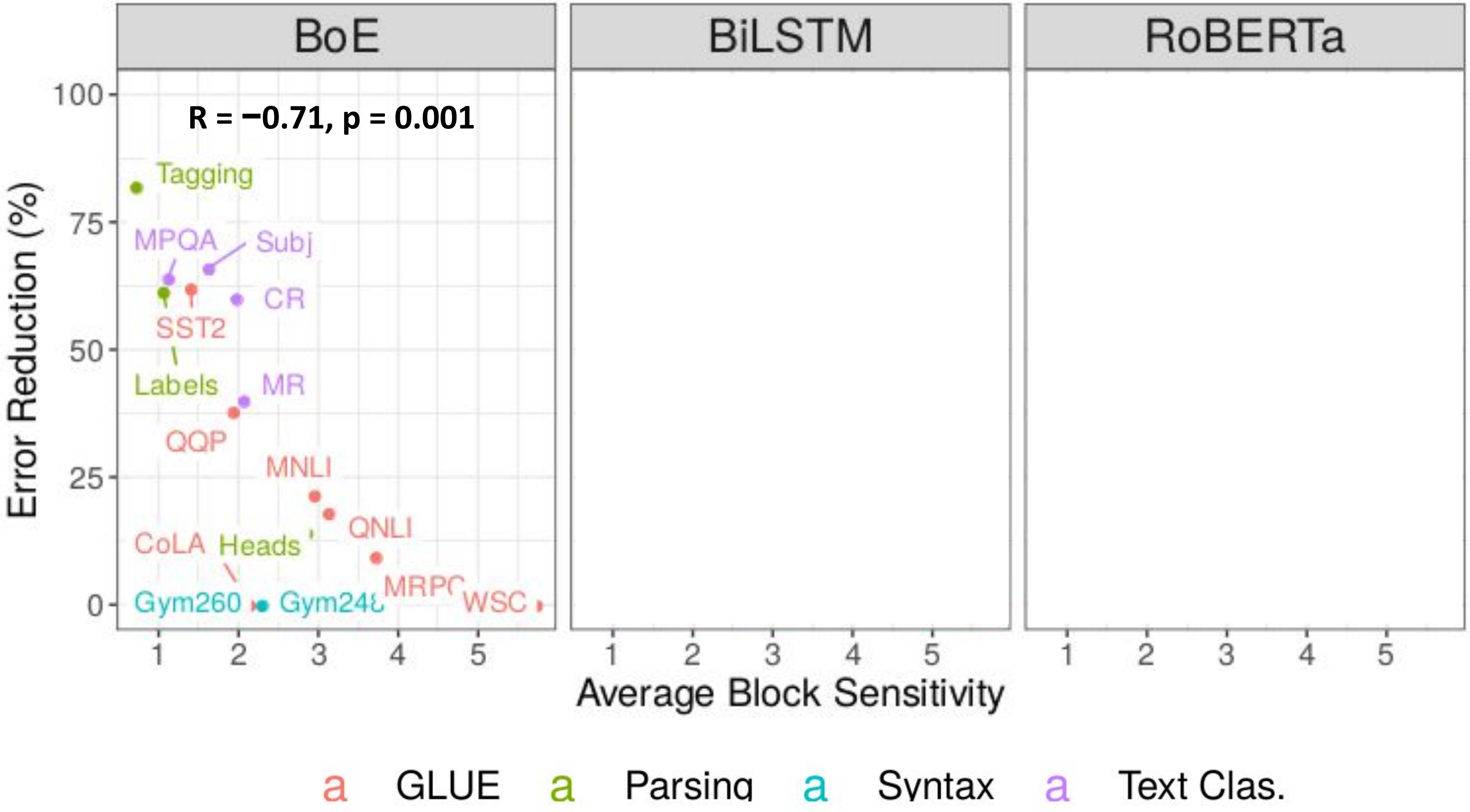
- *f* estimated using off-the-shelf parser (Qi et al., 2018, 2020) on English Web Treebank


GLUE
0
2
4
6
2
4
6
Task
CoLA
MNLI
MRPC
QNLI
QQP
RTE
SST2
STS-B
WSC
Parsing
0
2
4
6
2
4
6
Task
Heads
Labels
Tagging
Syntax
0
2
4
6
2
4
6
Task
Gym248
Gym260
Text Classification
0
2
4
6
2
4
6
Task
CR
MPQA
MR
Subj

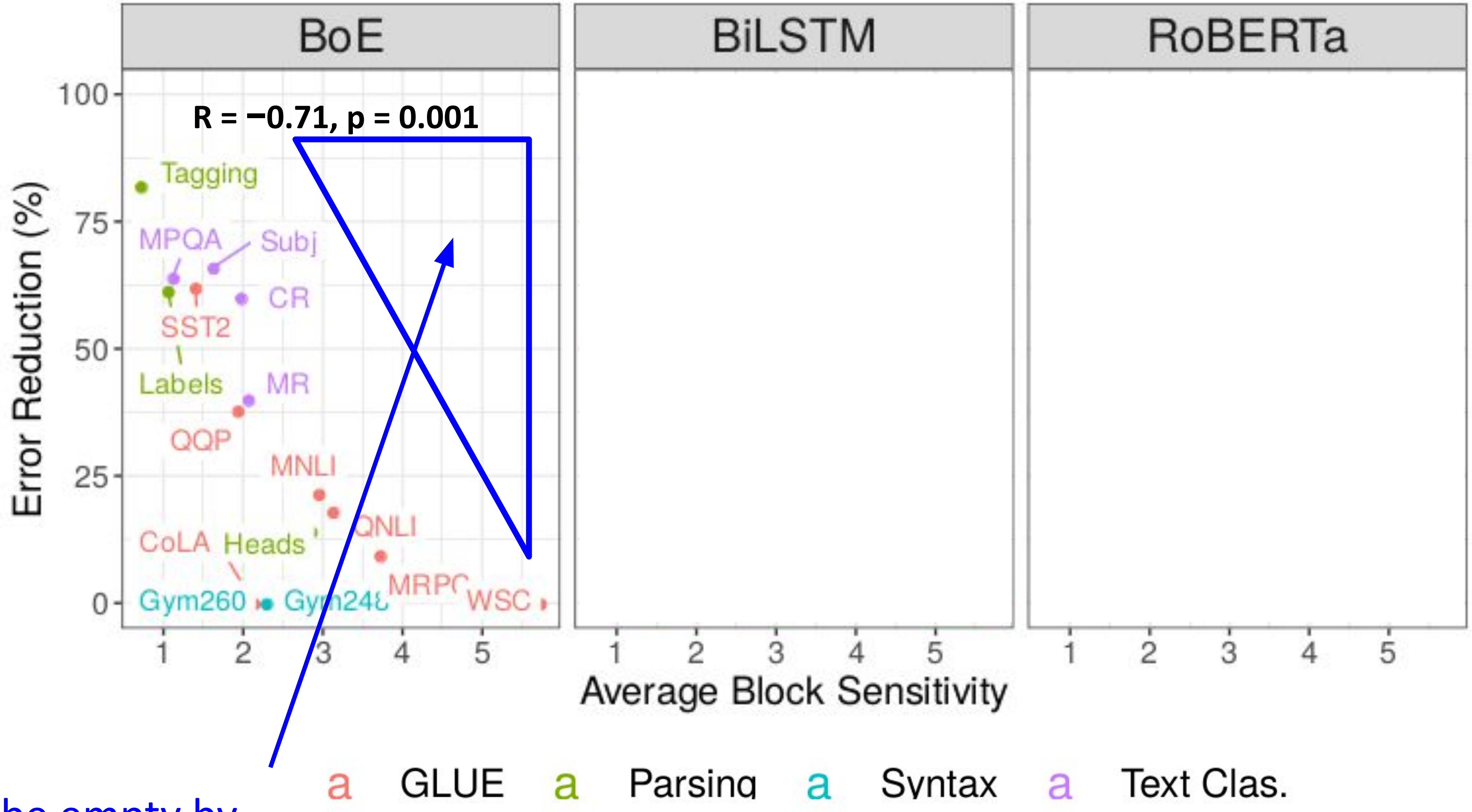
Input length doesn't explain away sensitivity differences



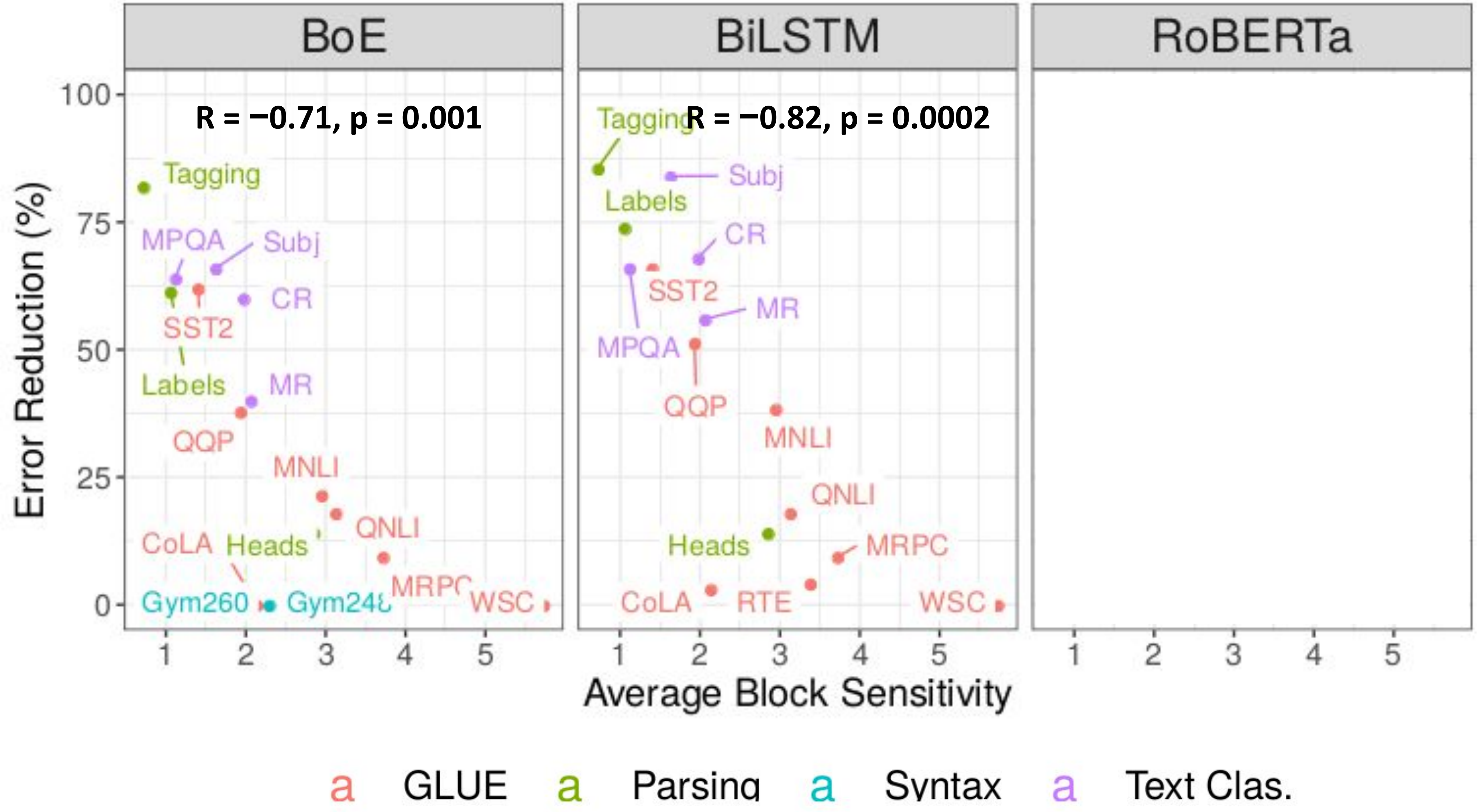

BoE
BiLSTM
RoBERTa
Error Reduction (%)
100
75
50
25
0
1
2
3
4
5
Average Block Sensitivity
GLUE
Parsing
Syntax
Text Clas.



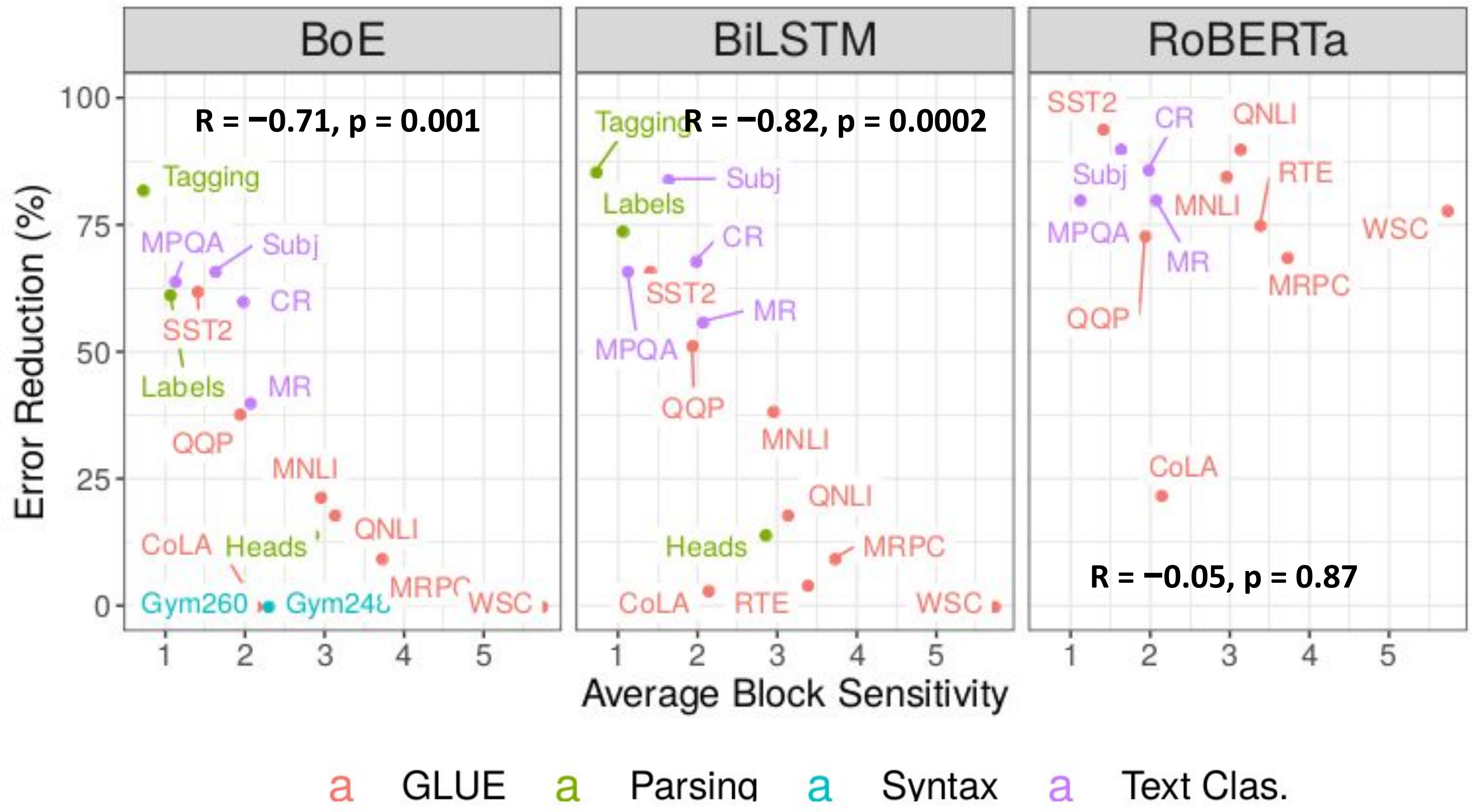
BoE
BiLSTM
RoBERTa
R = −0.71, p = 0.001
Tagging
MPQA
Subj
CR
SST2
Labels
MR
QQP
MNLI
QNLI
CoLA
Heads
MRPC
WSC
Gym260
Gym248
Error Reduction (%)
Average Block Sensitivity
GLUE
Parsing
Syntax
Text Clas.



Must be empty by theoretical bound!

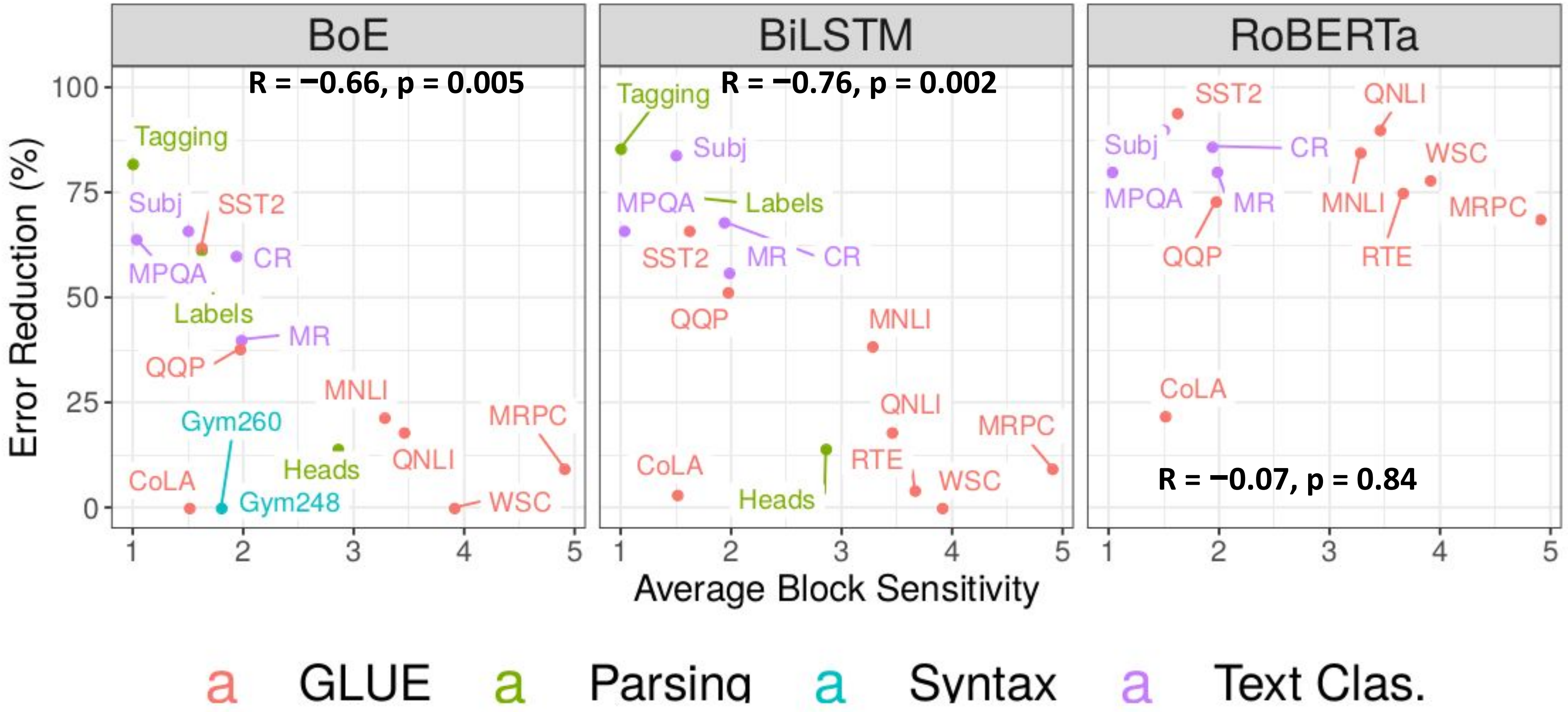


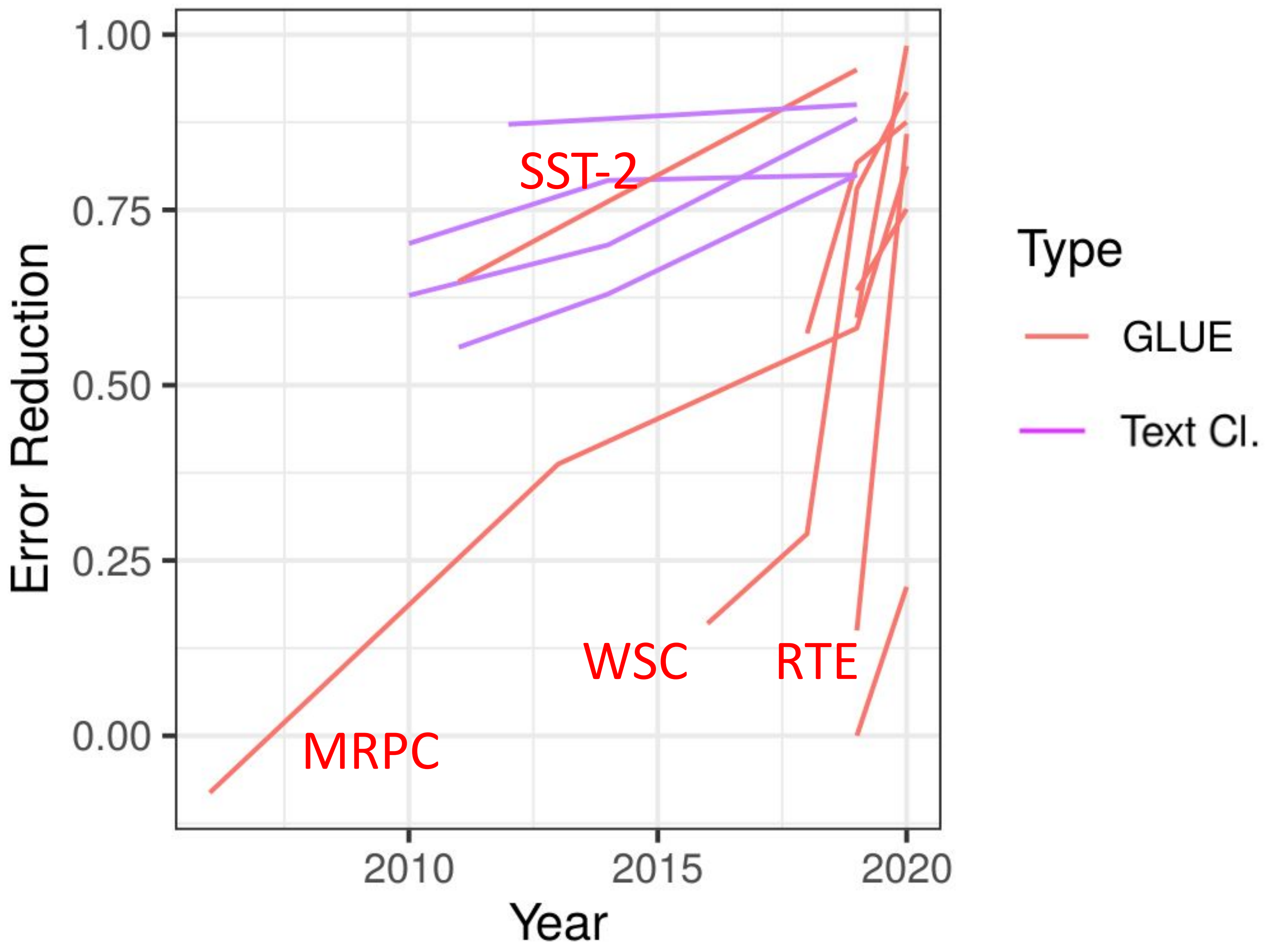
BoE
BiLSTM
RoBERTa
R = −0.71, p = 0.001
R = −0.82, p = 0.0002
Error Reduction (%)
Average Block Sensitivity
Tagging
MPQA
Subj
SST2
CR
Labels
MR
QQP
MNLI
QNLI
CoLA
Heads
MRPC
WSC
Gym260
RTE
GLUE
Parsing
Syntax
Text Clas.



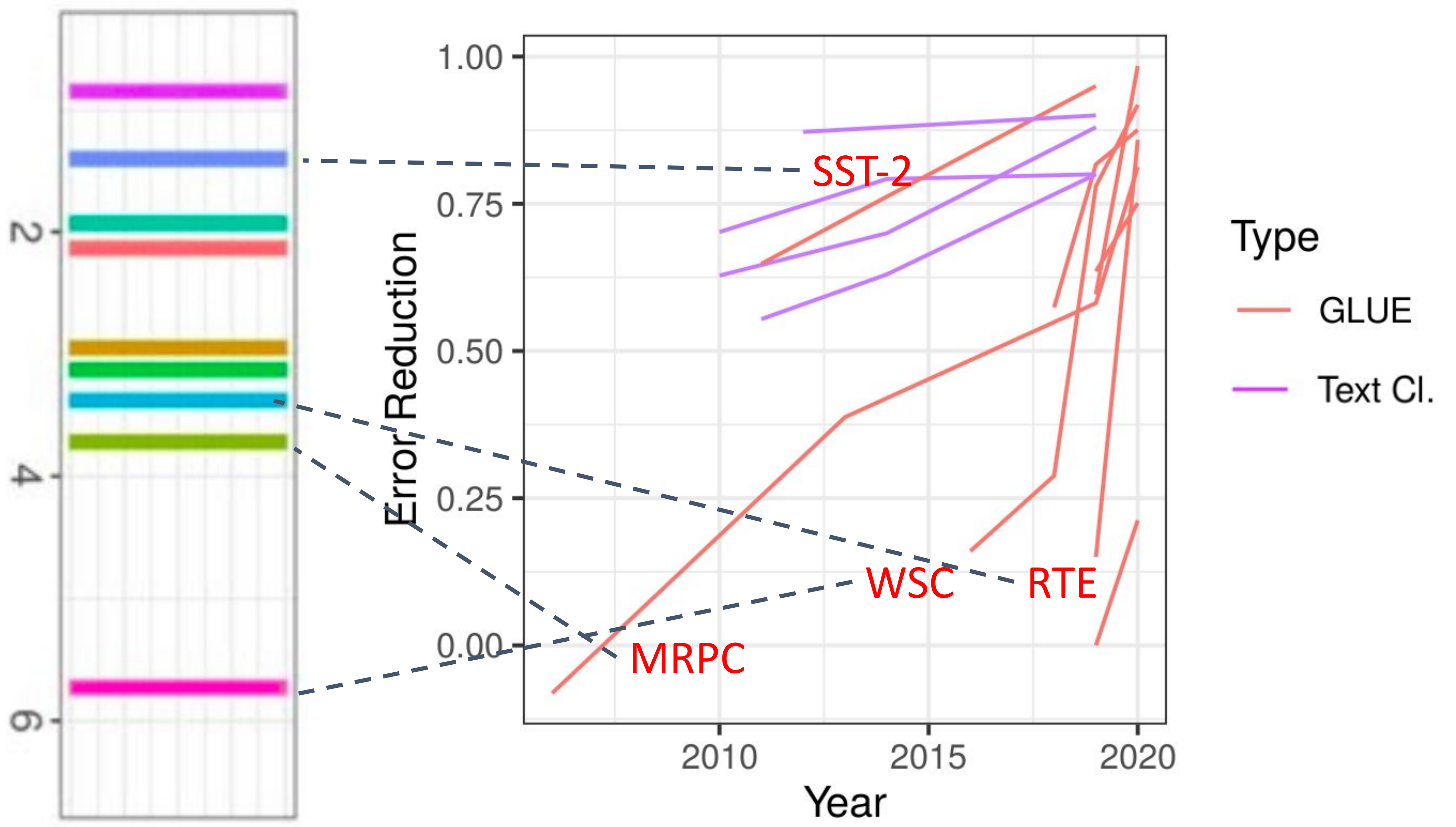
BoE
BiLSTM
RoBERTa
R = −0.71, p = 0.001
R = −0.82, p = 0.0002
R = −0.05, p = 0.87
Error Reduction (%)
Average Block Sensitivity
Tagging
MPQA
Subj
SST2
CR
Labels
MR
QQP
MNLI
QNLI
CoLA
Heads
MRPC
WSC
RTE
Gym260
GLUE
Parsing
Syntax
Text Clas.

Using u-PMLM (Liao et al., 2020) instead of XLNet:





1.00
0.75
0.50
0.25
0.00
Error Reduction
2010
2015
2020
Year
SST-2
WSC
RTE
MRPC
Type
GLUE
Text Cl.



2
4
6
1.00
0.75
0.50
0.25
0.00
Error Reduction
2010
2015
2020
Year
SST-2
WSC
RTE
MRPC
Type
GLUE
Text Cl.

Sensitivity Identifies Difficult Inputs

Low Sensitivity:

a gorgeous , witty , seductive movie .

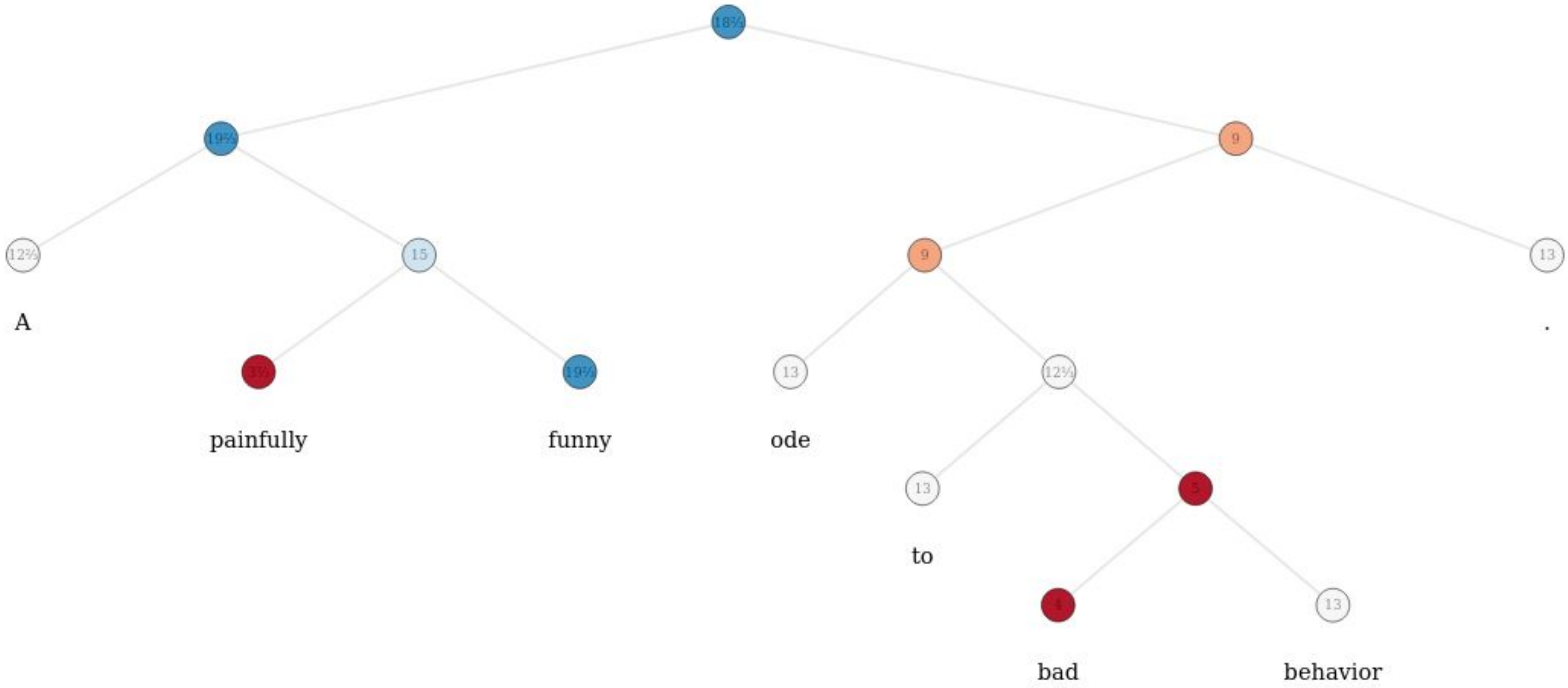
1. a farce of ideas squanders this movie .

High Sensitivity:

a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior .
2. a painfully bleak ode to bad behavior .
3. a painfully funny ode to bad movies .

Sensitivity Identifies Difficult Inputs

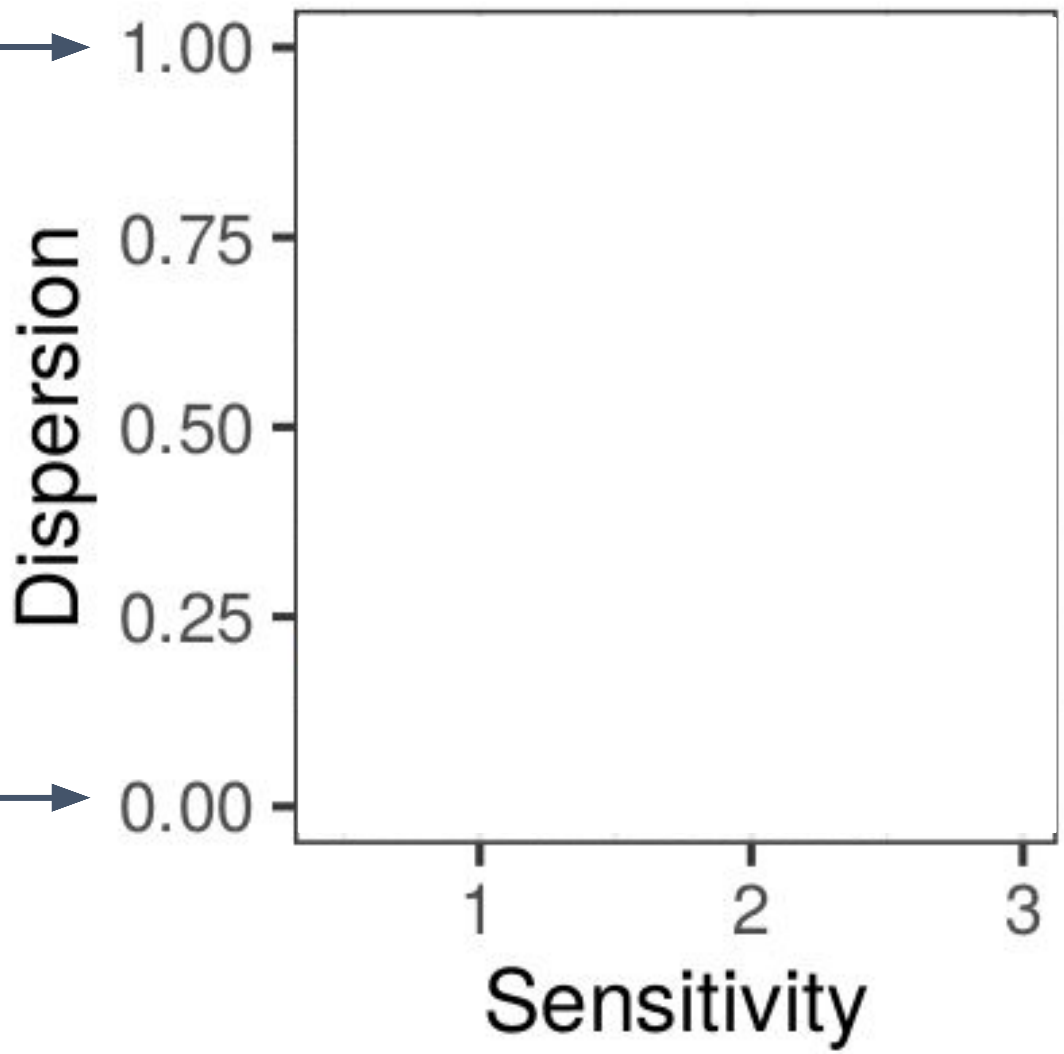


https://nlp.stanford.edu/sentiment/treebank.html?w=ode%2Cbad

Sensitivity Identifies Difficult Inputs

Same number of positive and negative constituents

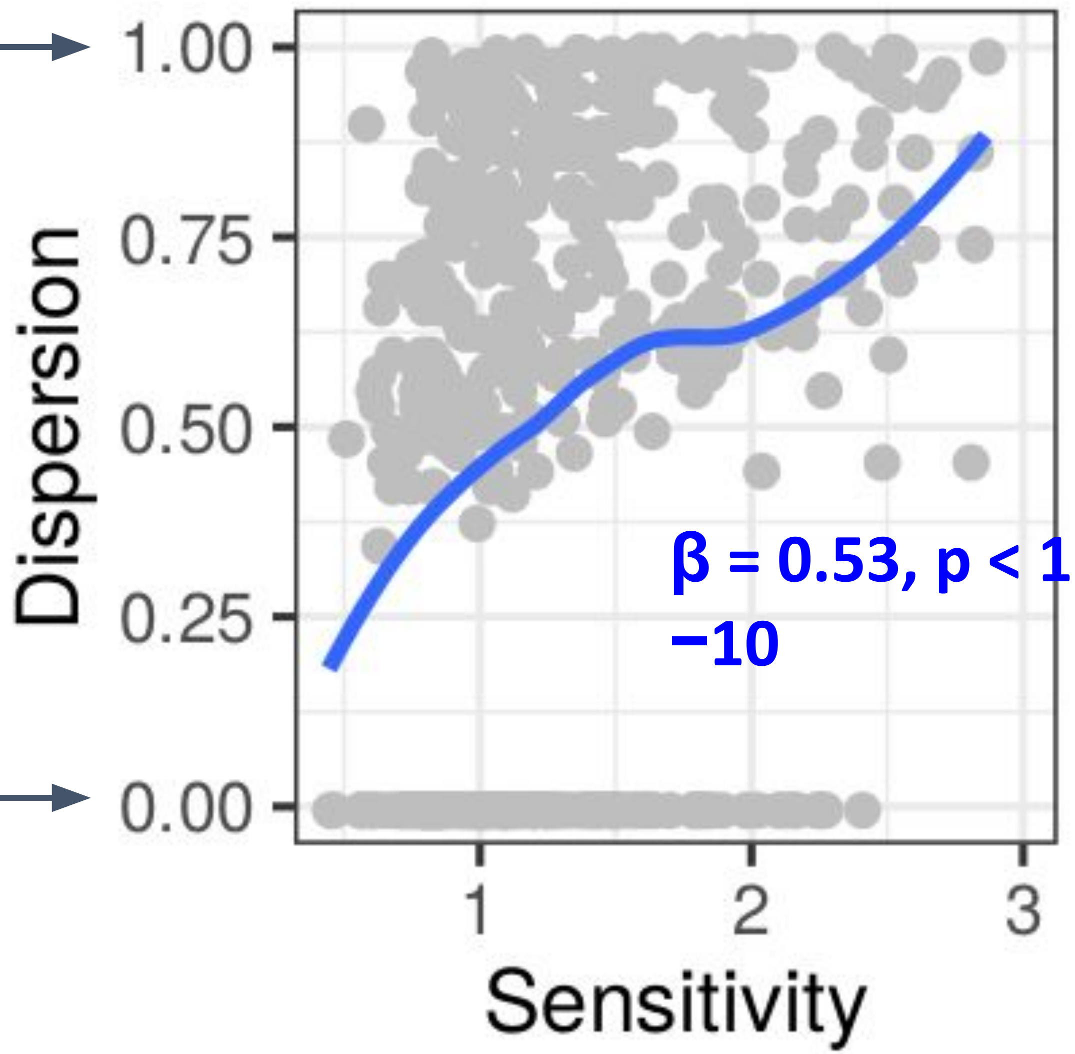
All constituents have the same sentiment.



Sensitivity Identifies Difficult Inputs

Same number of positive and negative constituents

All constituents have the same sentiment.



$\beta = 0.53$, $p < 1.95 \cdot 10^{-10}$

Sentence length not significant beyond dispersion ($\beta = 0$, $p = 0.49$)

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

Only positive constituents

Dispersion 0

Block Sensitivity 0.93

Low Sensitivity:

a gorgeous , witty , seductive movie .

1. a farce of ideas squanders this movie .

High Sensitivity:

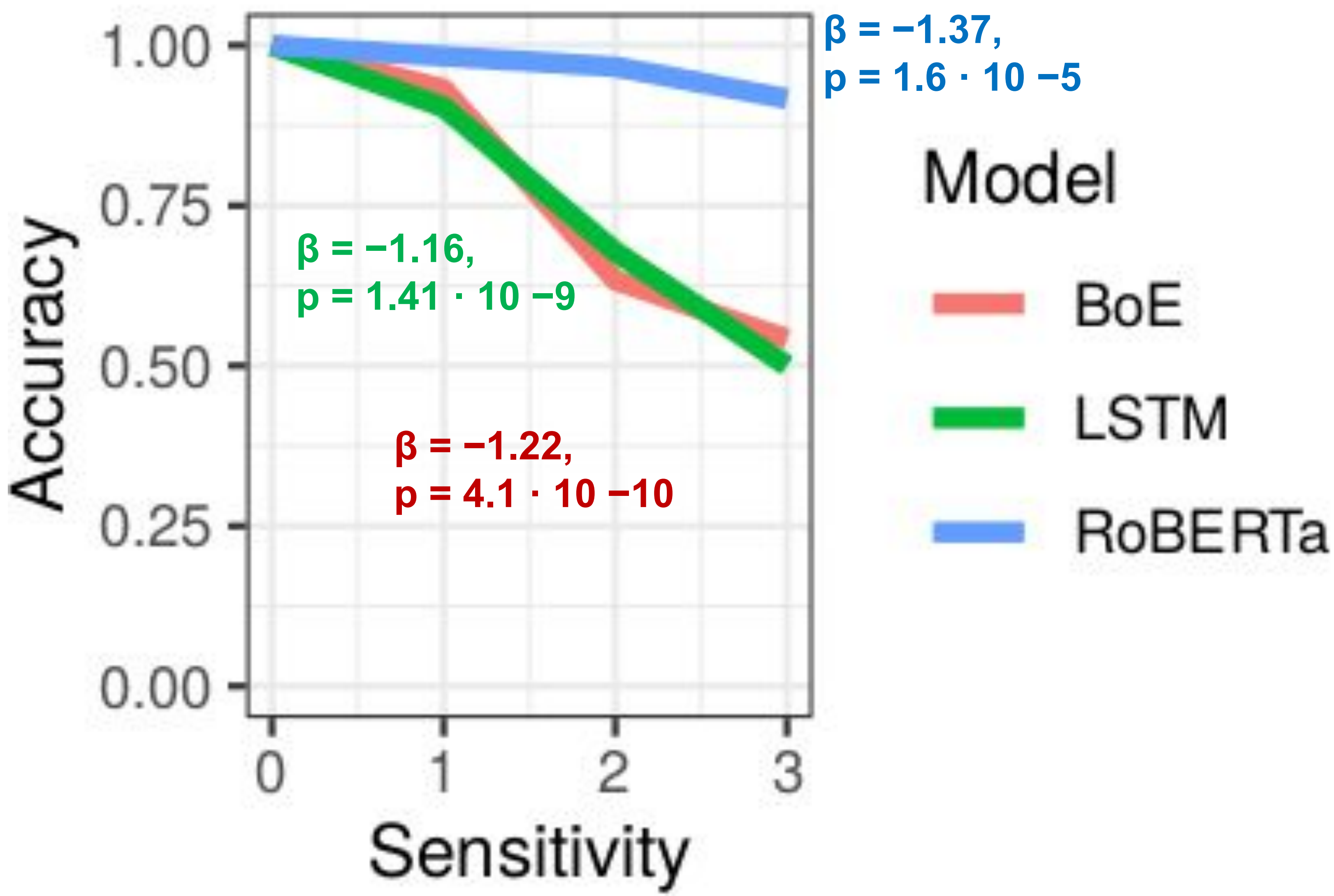
a painfully funny ode to bad behavior .

1. Not a funny story, just bad behavior .
2. a painfully bleak ode to bad behavior .
3. a painfully funny ode to bad movies .

Only positive constituents
Dispersion 0
Block Sensitivity 0.93

3 positive constituents
5 negative constituents
Dispersion 0.97
Block Sensitivity 1.88

Sensitivity Identifies Difficult Inputs



Sentence length not significant beyond sensitivity ($p > 0.05$)

Sensitivity as a Complexity Measure for Sequence Classification Tasks

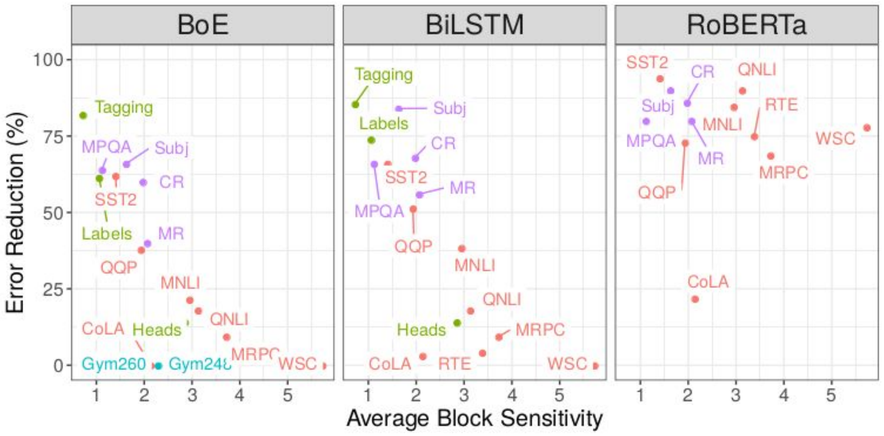
Sensitivity for Sequence Classification

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

Sensitivity and Difficulty of NLP Tasks

Sensitivity predicts **task difficulty**



Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

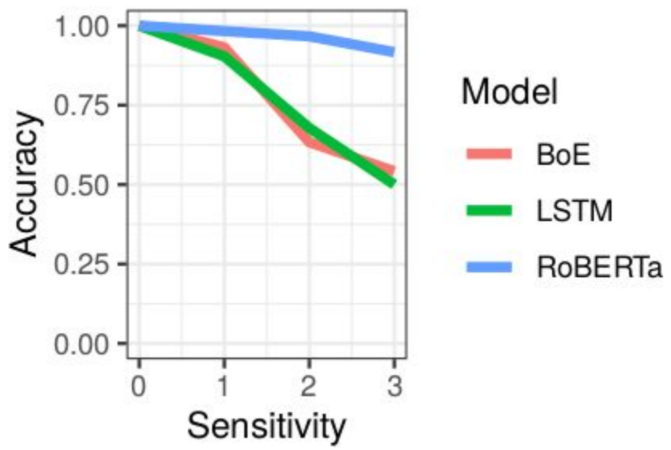
$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Sensitivity Bounds for ML Methods

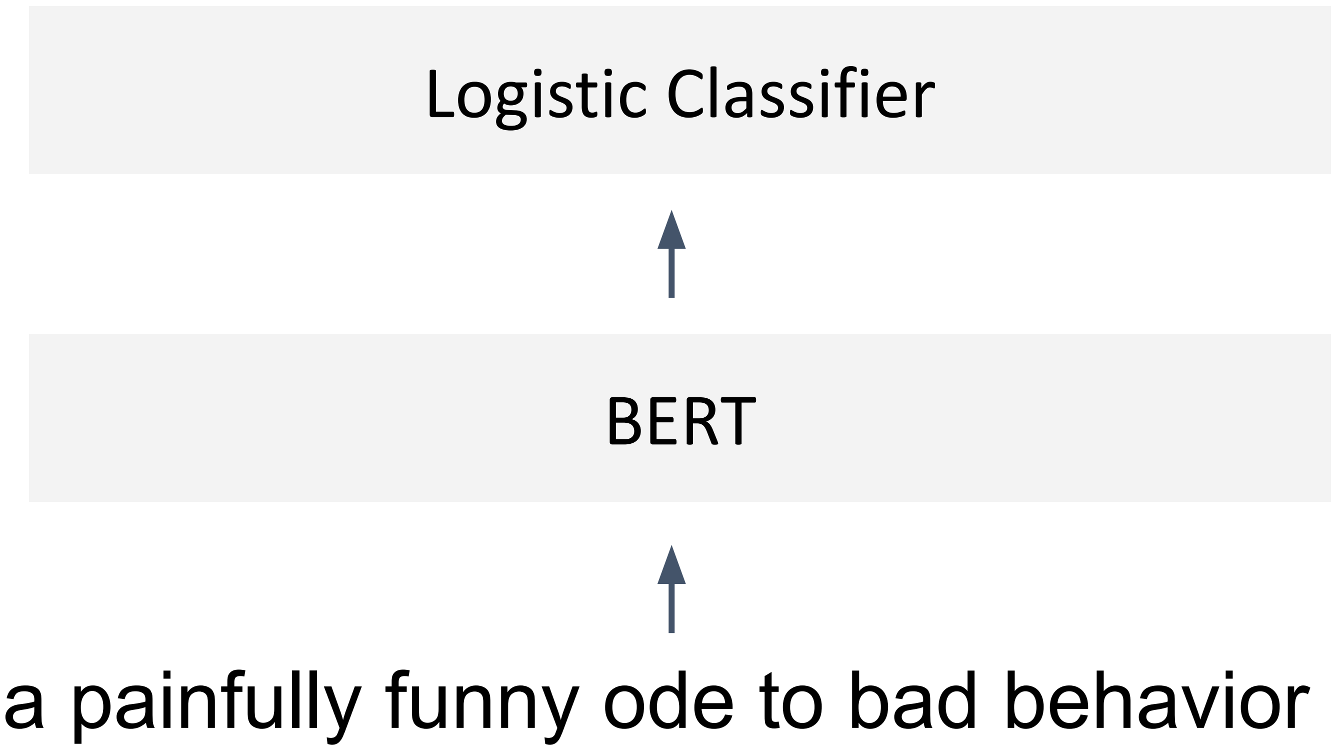
Sensitivity and Difficulty of NLP Tasks

Sensitivity predicts task difficulty

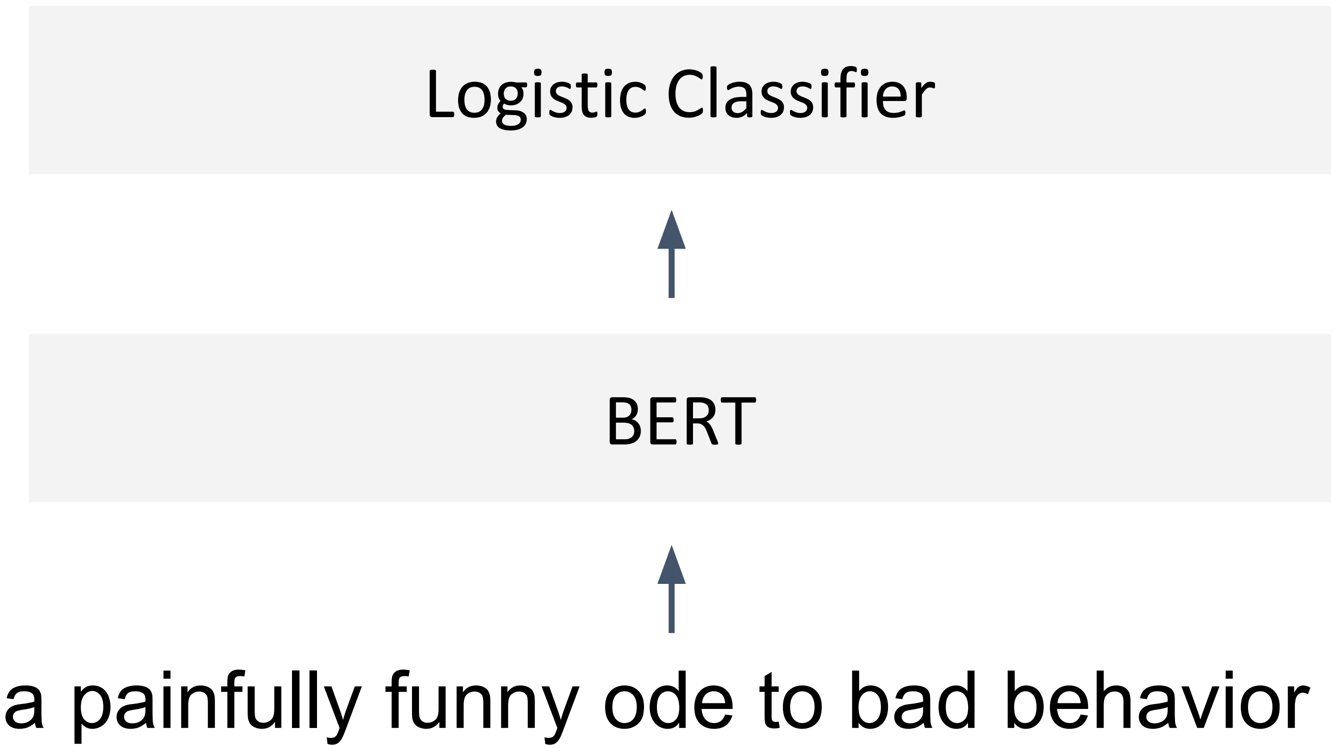
Sensitivity identifies difficult inputs



Why is Pretraining so Successful?

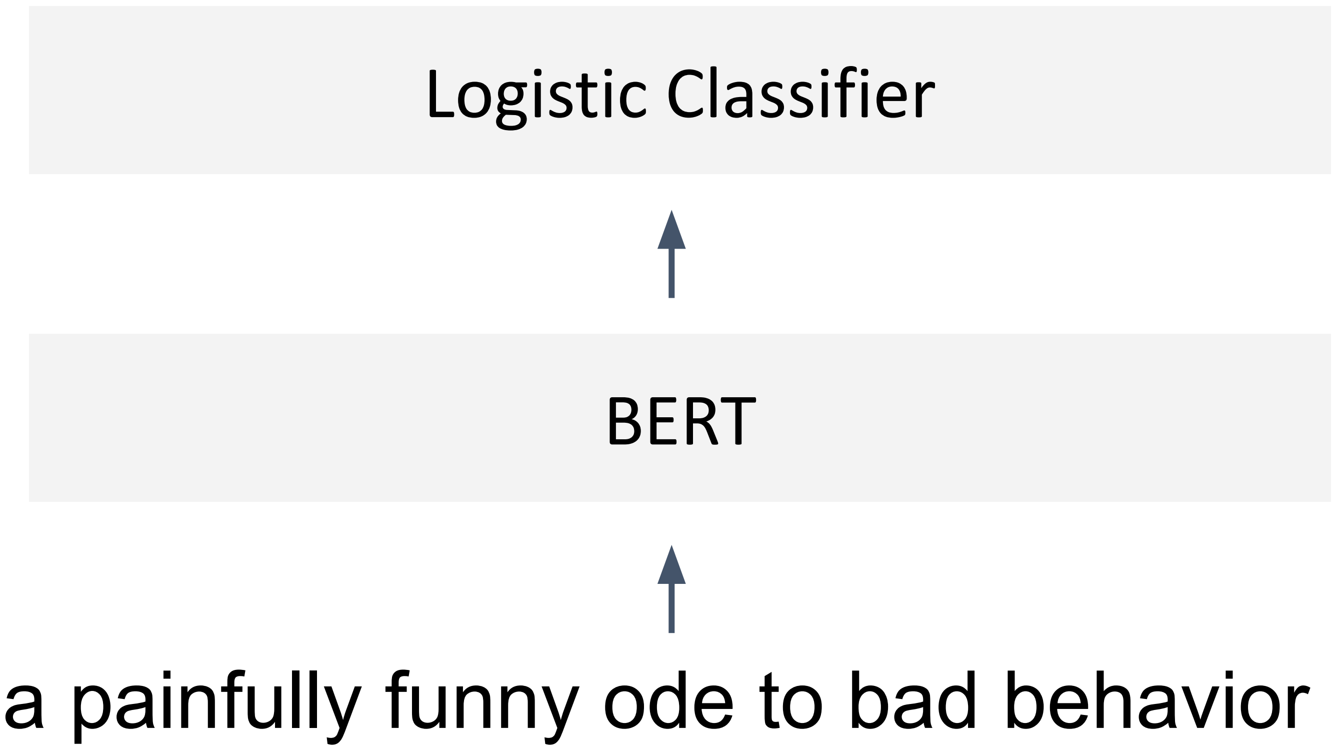


Why is Pretraining so Successful?



Pretrained models extract features from which label can be decoded with simple classifiers

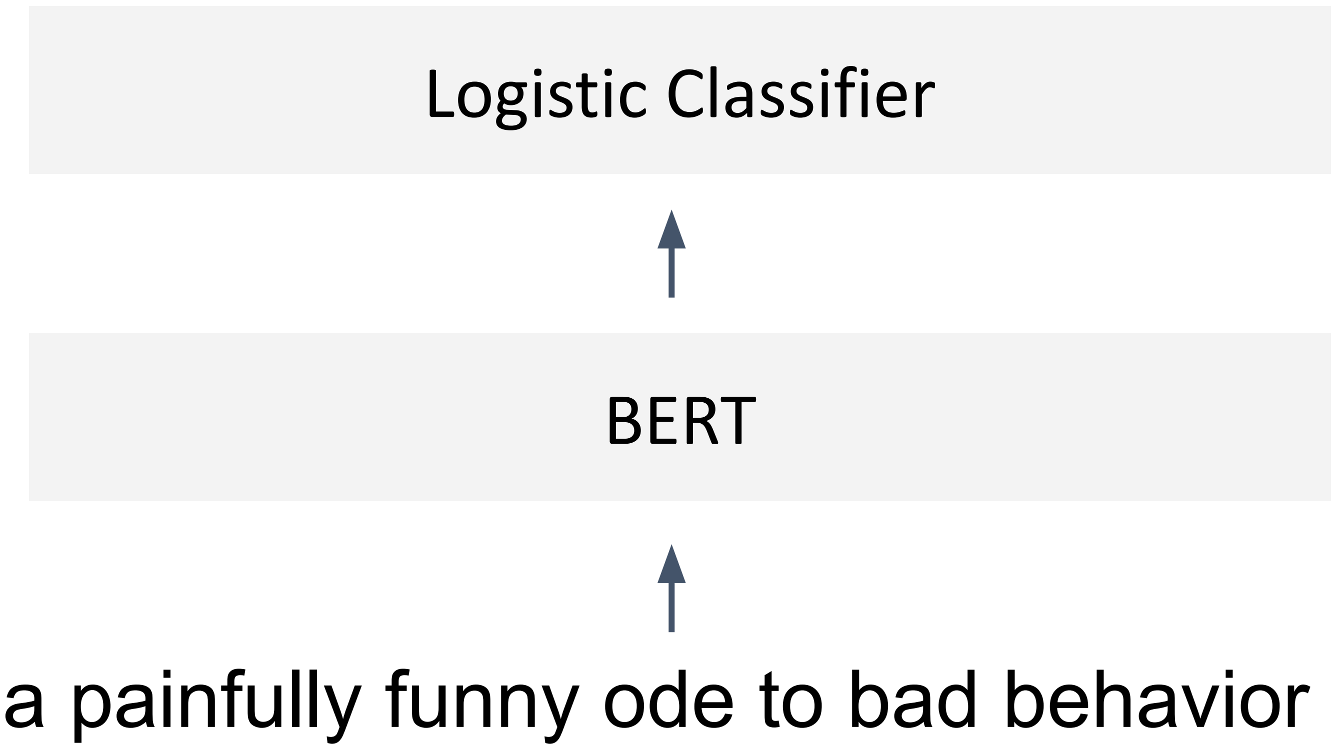
Why is Pretraining so Successful?



Pretrained models extract features from which label can be decoded with simple classifiers

In effect, they reduce task sensitivity!

Why is Pretraining so Successful?



Pretrained models extract features from which label can be decoded with simple classifiers

In effect, they reduce task sensitivity!

<u>Future work:</u>
How do they achieve this?

Sensitivity and Spurious Correlations

Models often rely on
spurious statistical patterns

Sensitivity and Spurious Correlations

Models often rely on
spurious statistical patterns

Simple lexical correlates of the label in Reading Comprehension
(e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018)

Sensitivity and Spurious Correlations

Models often rely on
spurious statistical patterns

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018)

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018)

Sensitivity and Spurious Correlations

Models often rely on spurious statistical patterns

Make the decision boundary `simpler' (e.g., Gardner et al. 2019).

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018)

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018)

Sensitivity and Spurious Correlations

Models often rely on spurious statistical patterns

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018)

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018)

Make the decision boundary `simpler' (e.g., Gardner et al. 2019).

Conjecture: They reduce sensitivity.

Sensitivity and Spurious Correlations

Models often rely on spurious statistical patterns

Make the decision boundary `simpler' (e.g., Gardner et al. 2019).

Conjecture: They reduce sensitivity.

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018)

→ Labels are correlated with output of simple lexical classifiers

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018)

Sensitivity and Spurious Correlations

Models often rely on spurious statistical patterns

Make the decision boundary `simpler' (e.g., Gardner et al. 2019).

Conjecture: They reduce sensitivity.

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018) → Labels are correlated with output of simple lexical classifiers

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018) → Changing the premise while staying within the task distribution less likely to flip the label

Sensitivity and Spurious Correlations

Models often rely on spurious statistical patterns

Make the decision boundary `simpler' (e.g., Gardner et al. 2019).

Conjecture: They reduce sensitivity.

Simple lexical correlates of the label in Reading Comprehension (e.g. Kaushik and Lipton, 2018; Gururangan et al., 2018) → Labels are correlated with output of simple lexical classifiers

Hypothesis alone predictive of the label in entailment tasks (Poliak et al., 2018) → Changing the premise while staying within the task distribution less likely to flip the label

Future work: Can sensitivity help automatically mitigate spurious patterns?

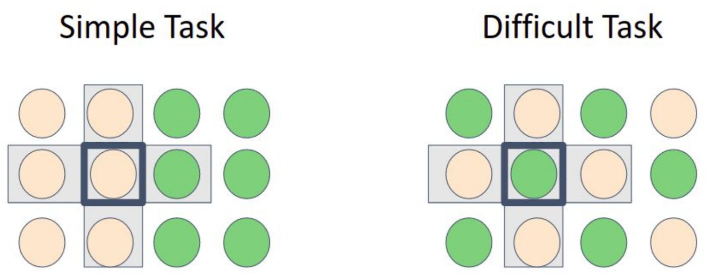
Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Conclusion

Introduced sensitivity as a complexity measure for sequence classification

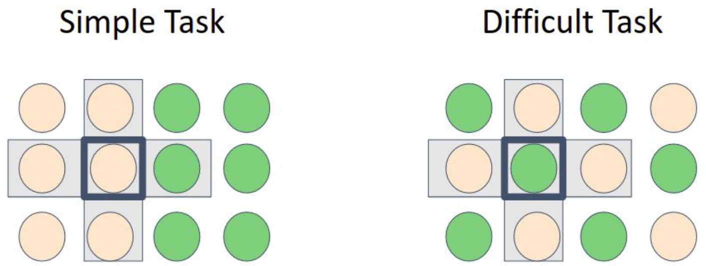
Measures complexity of decision boundary



Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Measures complexity of decision boundary



Generalizes well-studied theory from Boolean functions to general sequence classification

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Sensitivity predicts what functions are difficult for ML models

Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Sensitivity predicts what functions are difficult for ML models

Simple lexical classifiers cannot express high-sensitivity functions

$$bs(f,x) \leq 2L^2C^2k^2$$

Conclusion

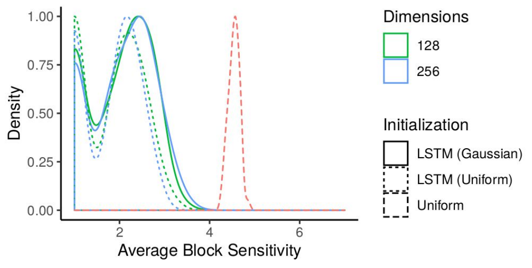
Introduced sensitivity as a complexity measure for sequence classification

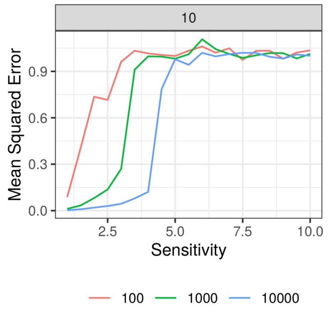
Sensitivity predicts what functions are difficult for ML models

Simple lexical classifiers cannot express high-sensitivity functions

$$bs(f,x) \leq 2L^2C^2k^2$$

Even LSTMs & Transformers are biased towards low sensitivity





Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Sensitivity predicts what functions are difficult for ML models

Sensitivity predicts difficulty of NLP tasks

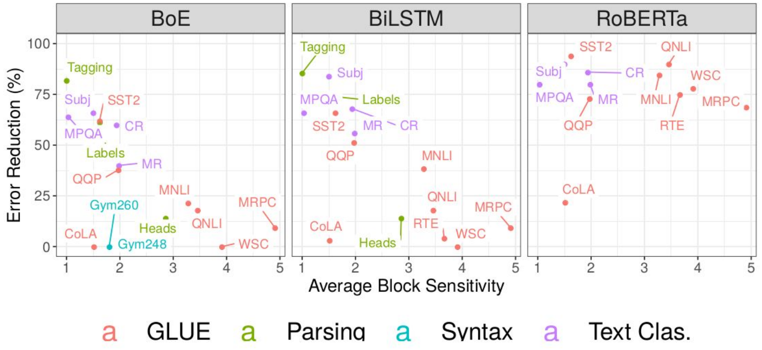
Conclusion

Introduced sensitivity as a complexity measure for sequence classification

Sensitivity predicts what functions are difficult for ML models

Sensitivity predicts difficulty of NLP tasks

Characterizes which tasks require pretrained models

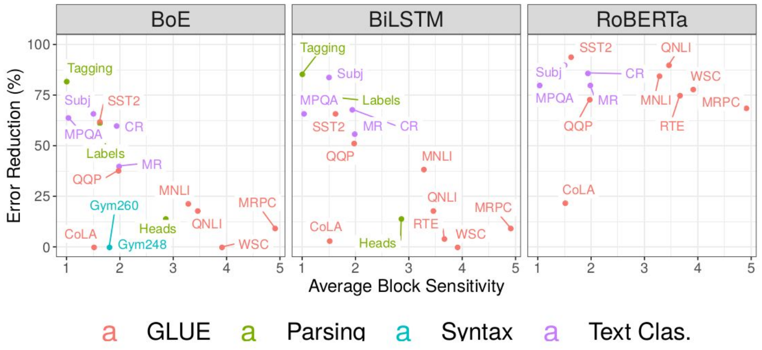


Conclusion

Introduced sensitivity as a complexity measure for sequence classification

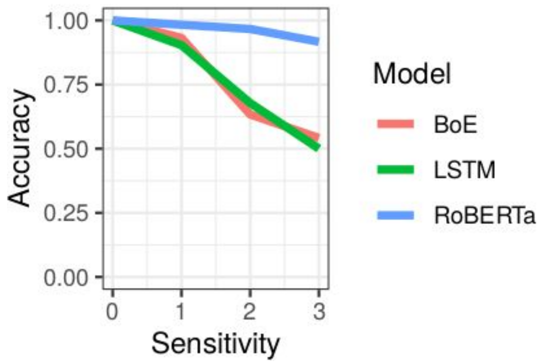
Sensitivity predicts what functions are difficult for ML models

Sensitivity predicts difficulty of NLP tasks



Characterizes which tasks require pretrained models

Predicts difficulty of individual inputs



Thanks!

Why Subsets?

Why Subsets instead of Individual Words?

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

(1) Words are composed into phrases. Changing a phrase can change meaning when changing any word cannot.

a gorgeous , witty , seductive movie .
a farce of ideas squanders this movie .

Why Subsets instead of Individual Words?

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

(1) Words are composed into phrases. Changing a phrase can change meaning when changing any word cannot.

(2) There are distributions $\prod$ where we would always get 0 with singletons.

Why Subsets instead of Individual Words?

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

(1) Words are composed into phrases. Changing a phrase can change meaning when changing any word cannot.

(2) There are distributions $\prod$ where we would always get 0 with singletons.

(3) Block sensitivity can only increase with finer tokenization.

Why Subsets instead of Individual Words?

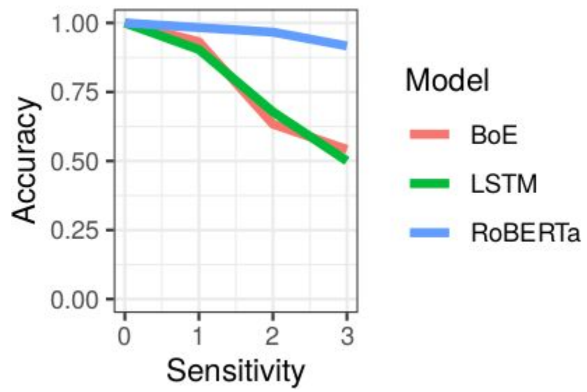
$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

(1) Words are composed into phrases. Changing a phrase can change meaning when changing any word cannot.

(2) There are distributions $\prod$ where we would always get 0 with singletons.

(3) Block sensitivity can only increase with finer tokenization.

(4) Model fit predicting accuracy on SST-2 is stronger with block sensitivity



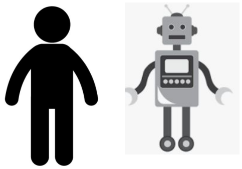
Relation to Adversarial Examples

Relation to Adversarial Examples

Adversarial Brittleness

(Szegedy et al., 2013; Jia and Liang, 2017)

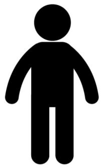
Neighboring inputs on which the model output changes erroneously.



a painfully funny ode to bad behavior	+1	+1
a painfully funny ode to terrible behavior	+1	−1

High Sensitivity

Neighboring inputs within the data distribution on which the true label changes.



a painfully funny ode to bad behavior	+1
not really funny, just bad behavior	−1

Measuring Sensitivity without Models

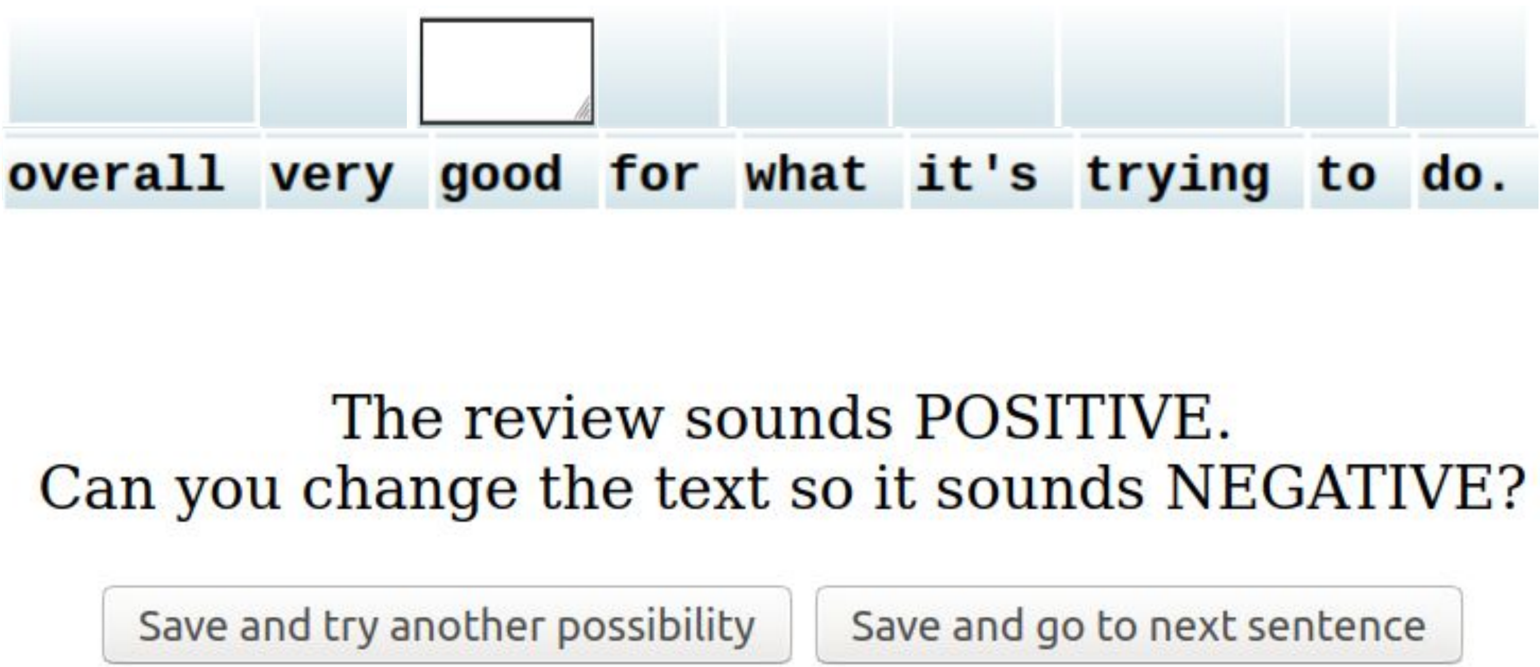
Approximating Sensitivity without Models

overall very good for what it's trying to do.

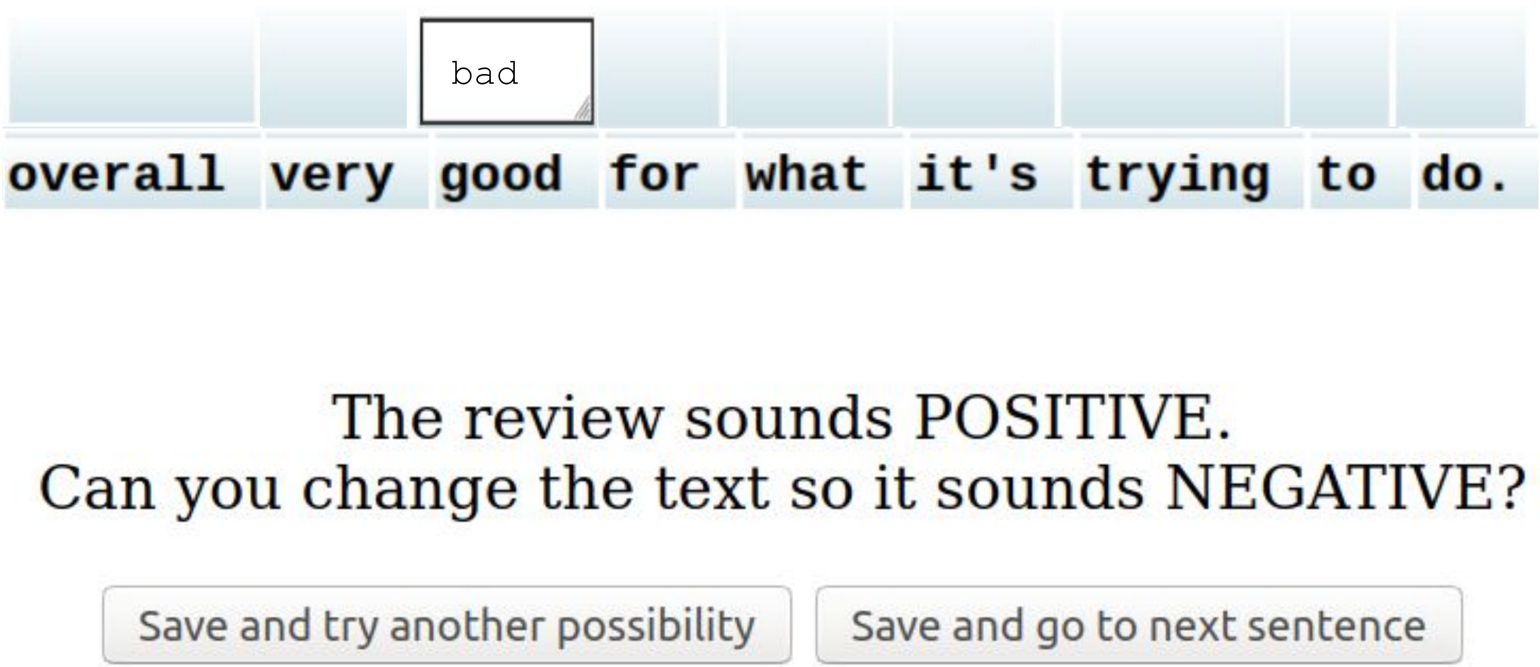
The review sounds POSITIVE.
Can you change the text so it sounds NEGATIVE?

Save and try another possibility | Save and go to next sentence

Approximating Sensitivity without Models



Approximating Sensitivity without Models



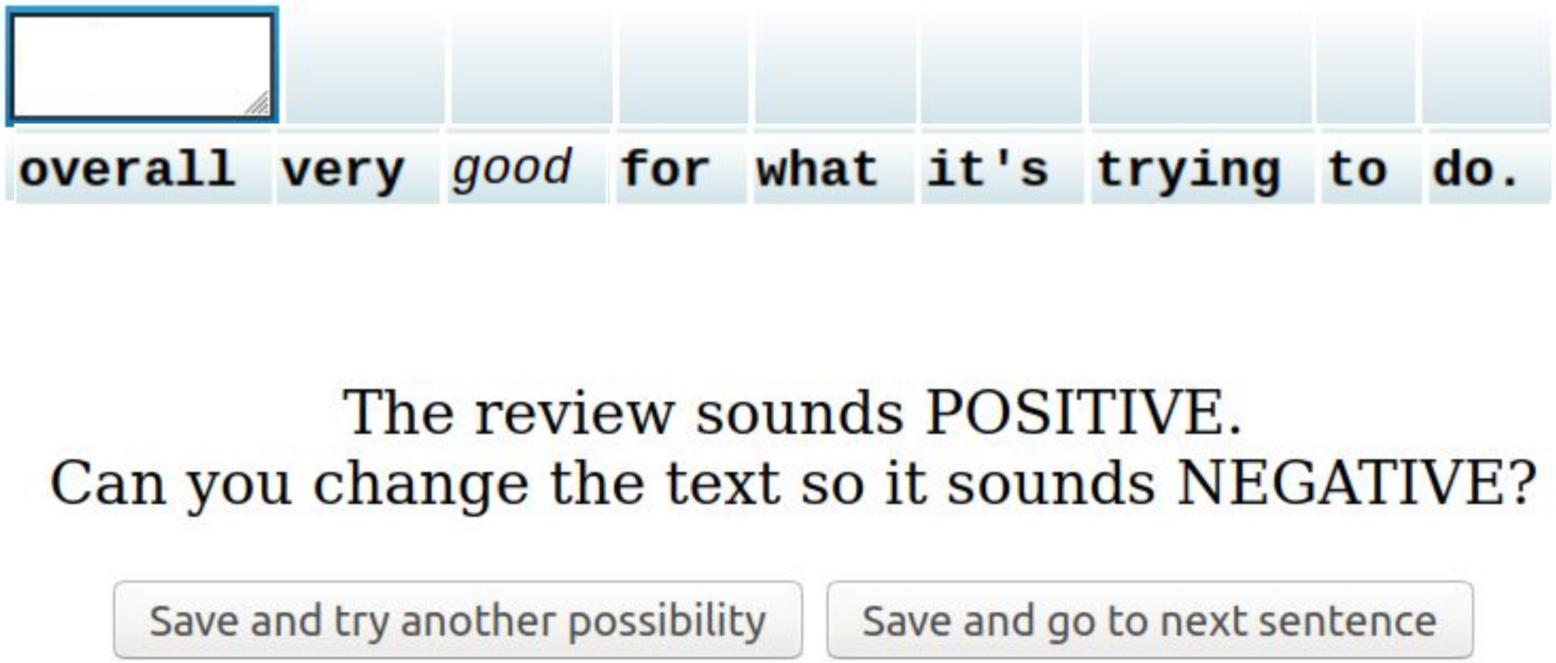
Approximating Sensitivity without Models

overall very *good* for what it's trying to do.

The review sounds POSITIVE.
Can you change the text so it sounds NEGATIVE?

Save and try another possibility | Save and go to next sentence

Approximating Sensitivity without Models



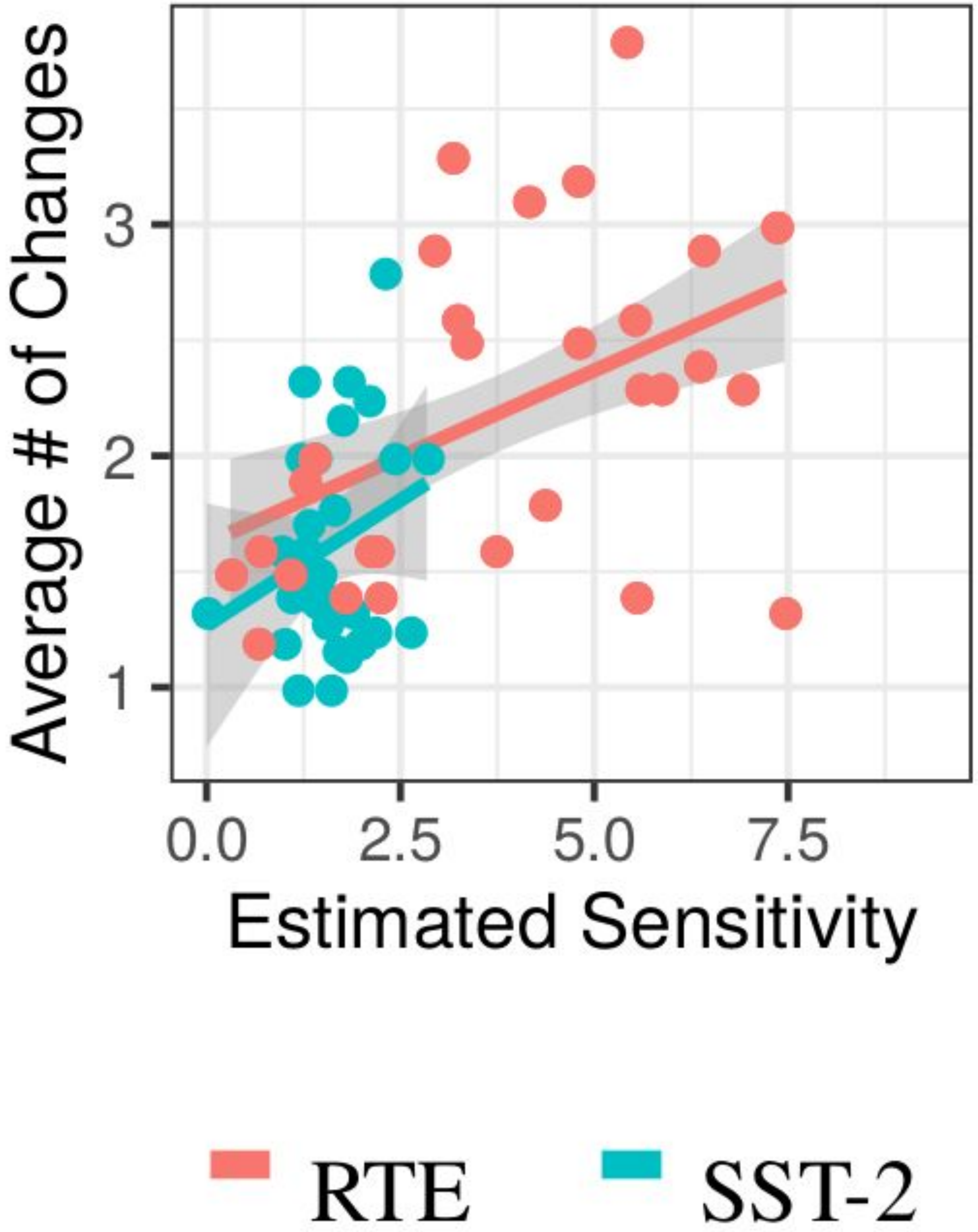
Approximating Sensitivity without Models

not

overall very *good* for what it's trying to do.

The review sounds POSITIVE.
Can you change the text so it sounds NEGATIVE?

Save and try another possibility | Save and go to next sentence



Poisson regression:

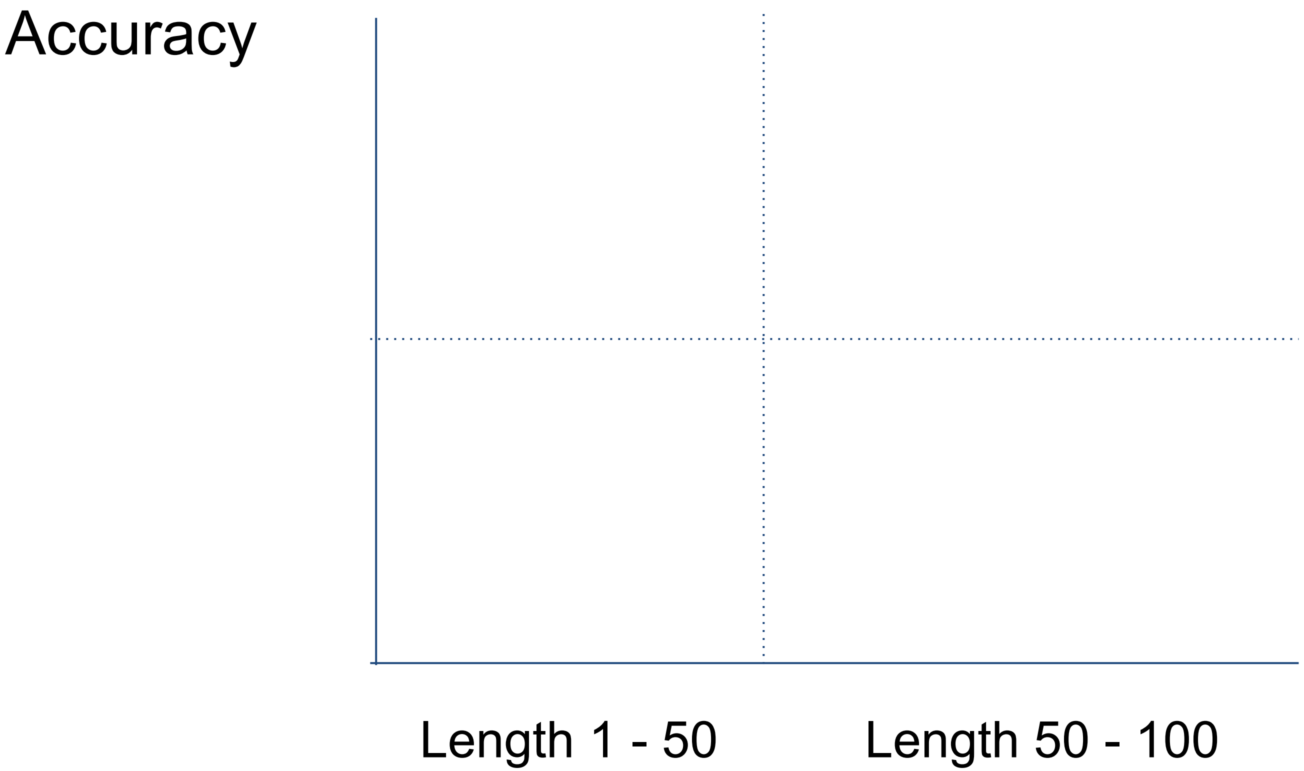
β = 0.061,

p = 0.0023

controlling for task, sentence length, and random variation between sentences and annotators.

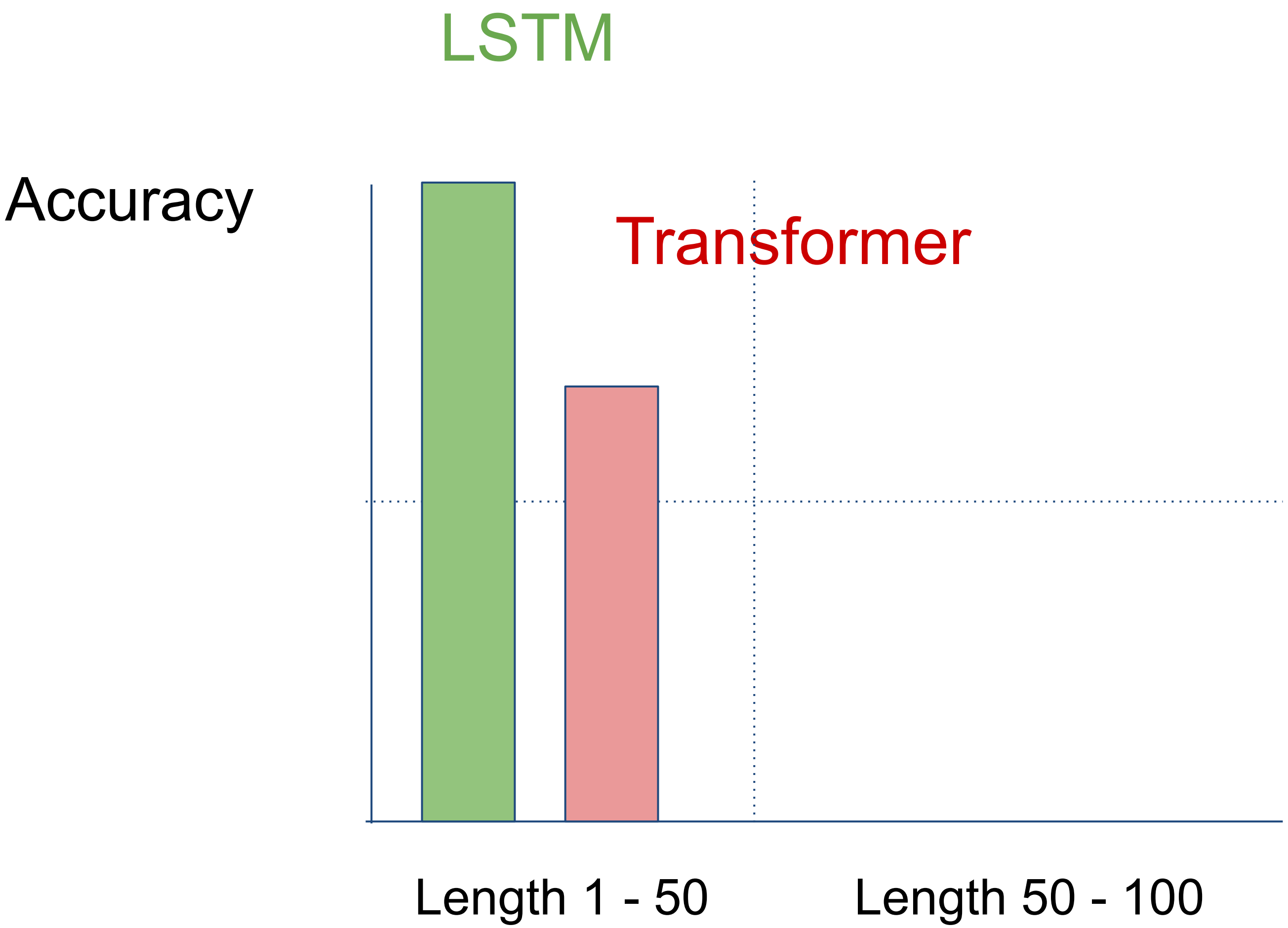
Empirical Learnability of PARITY

Empirical Learnability Results:



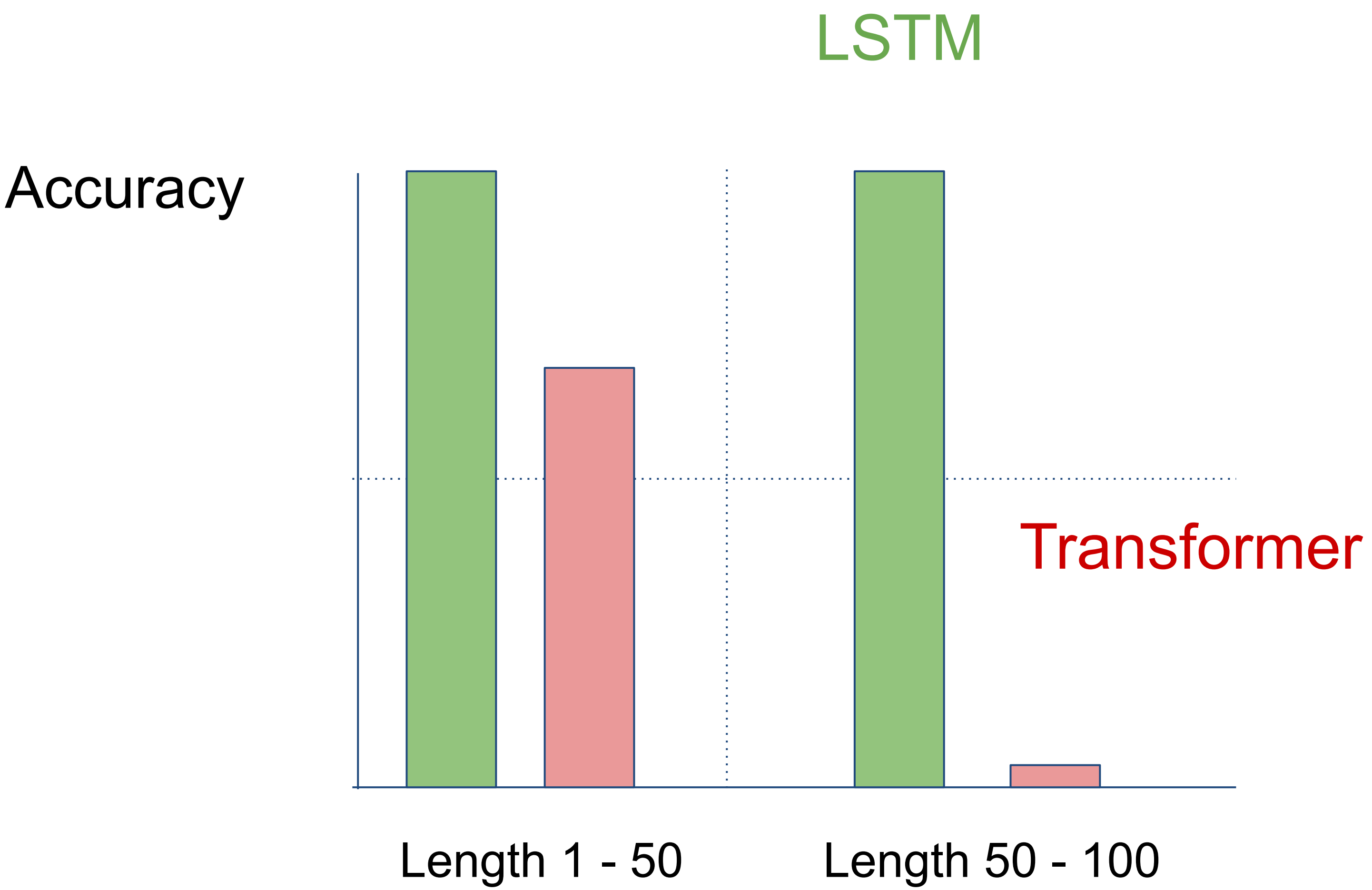
Bhattamishra, Ahuja, Goyal (2020, EMNLP)

Empirical Learnability Results:



Bhattamishra, Ahuja, Goyal (2020, EMNLP)

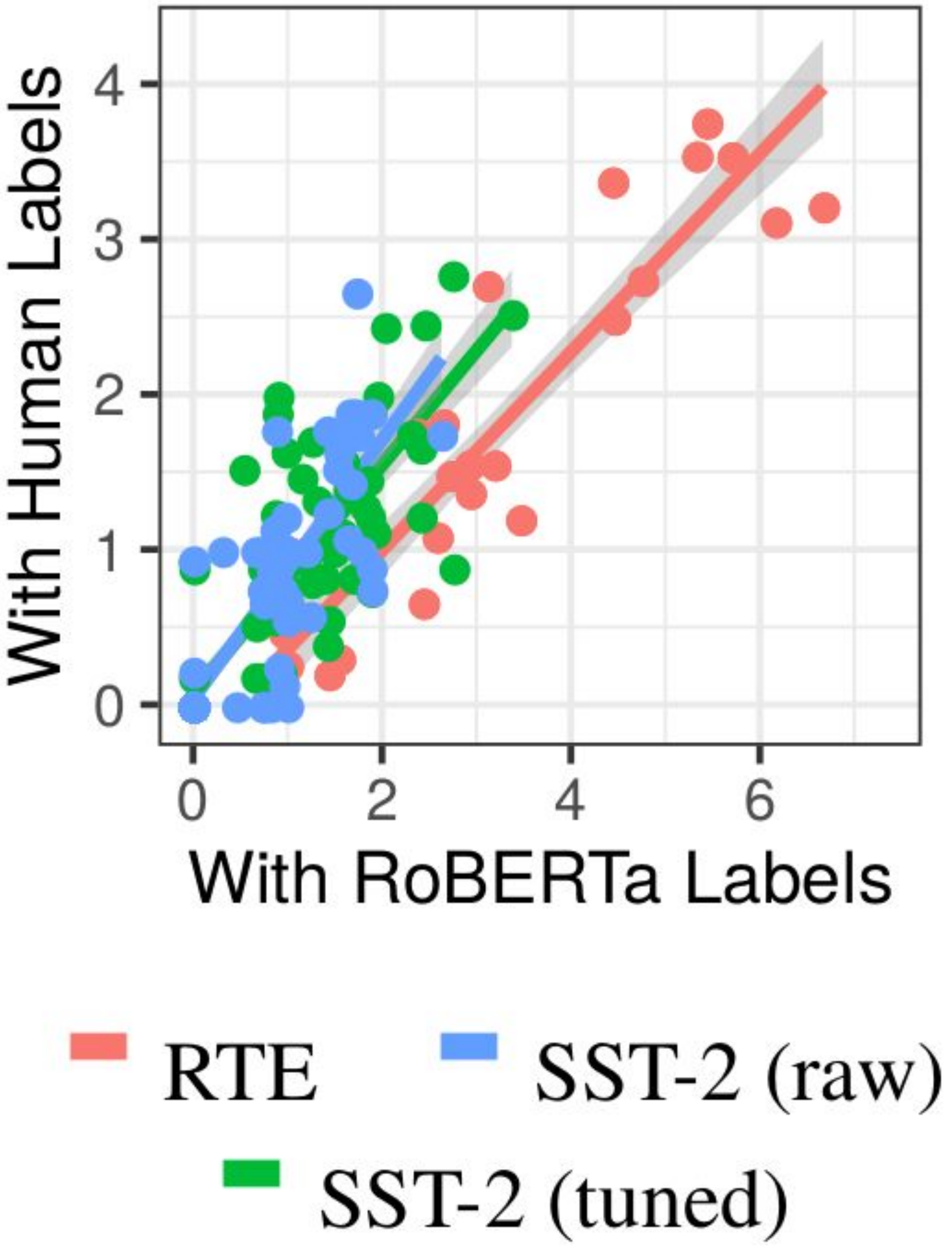
Empirical Learnability Results:



Bhattamishra, Ahuja, Goyal (2020, EMNLP)

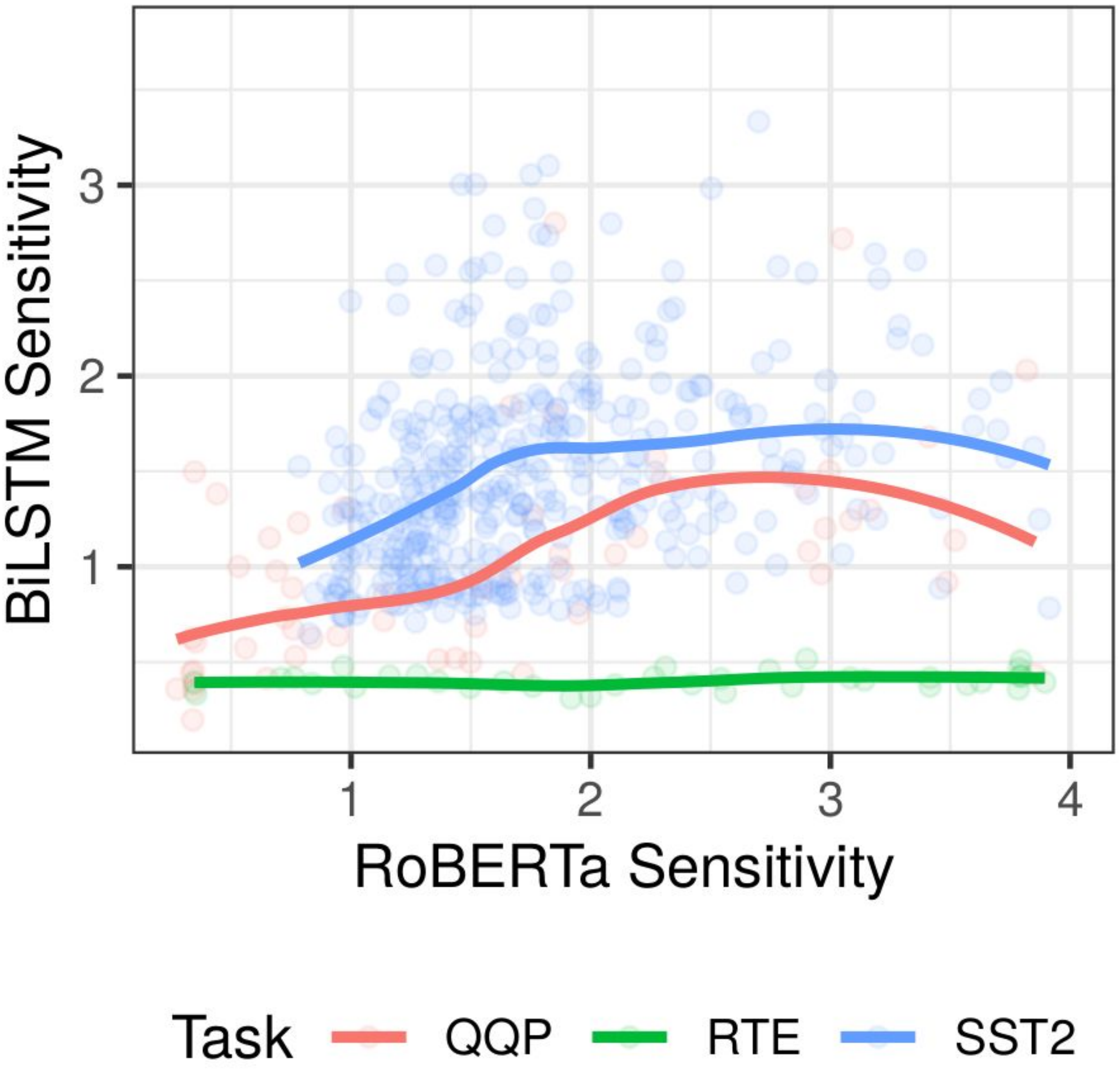
Sensitivity using Human Oracle Labels

Sensitivity using Human Oracle Labels



Role of Task Model

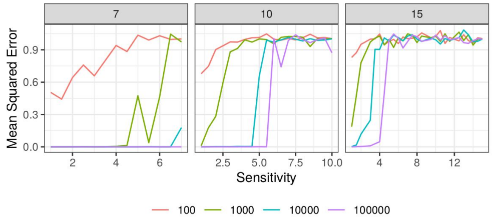
Role of Task Model



Inductive Biases in DL

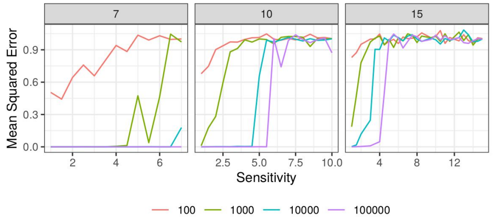
Inductive Biases in DL

Even powerful neural models are biased towards low sensitivity



Inductive Biases in DL

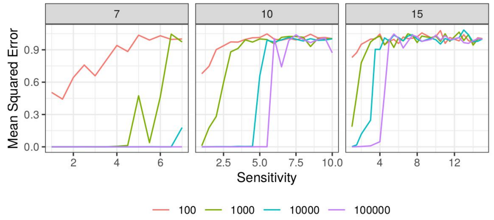
Even powerful neural models are biased towards low sensitivity



Empirical and theoretical evidence that neural networks generalize because they are biased towards “simple” functions (De Palma et al., 2018).

Inductive Biases in DL

Even powerful neural models are biased towards low sensitivity

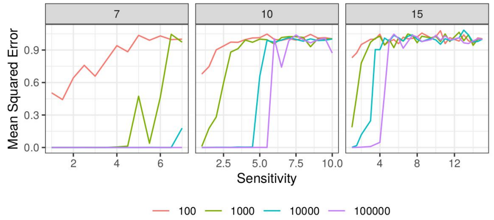


Empirical and theoretical evidence that neural networks generalize because they are biased towards “simple” functions (De Palma et al., 2018).

Some studies propose notions close to sensitivity (Franco, 2006; De Palma et al., 2018, Novak et al., 2018).

Inductive Biases in DL

Even powerful neural models are biased towards low sensitivity



Empirical and theoretical evidence that neural networks generalize because they are biased towards “simple” functions (De Palma et al., 2018).

Some studies propose notions close to sensitivity (Franco, 2006; De Palma et al., 2018, Novak et al., 2018).

Empirically, neural networks learn low Fourier frequencies first

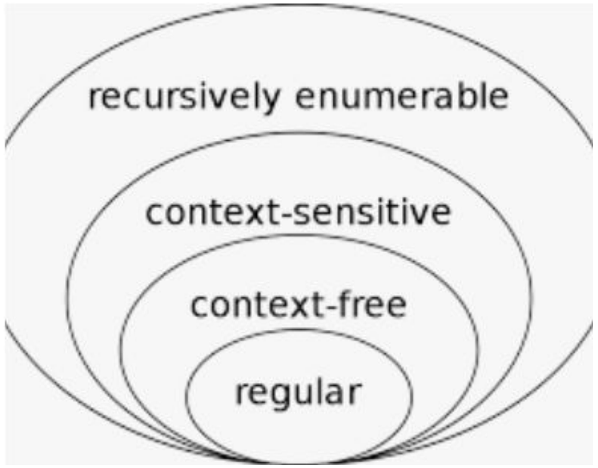
(Rahaman et al., 2019; Xu et al., 2019; Cao et al., 2019).

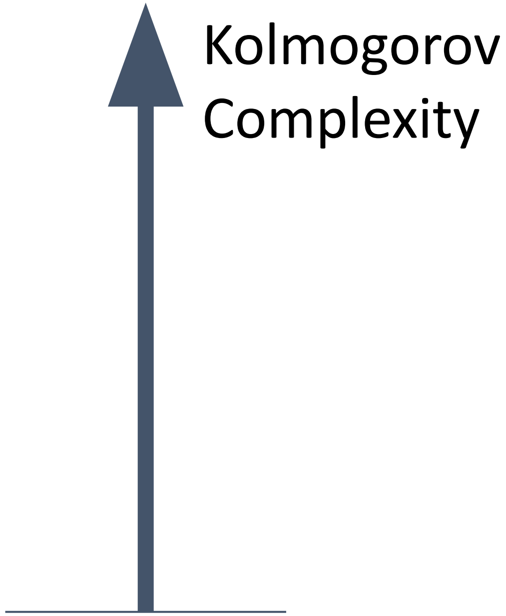
- For Boolean functions: low average sensitivity <=> Fourier spectrum concentrated on low frequencies!

Other Complexity Metrics

Other Complexity Measures

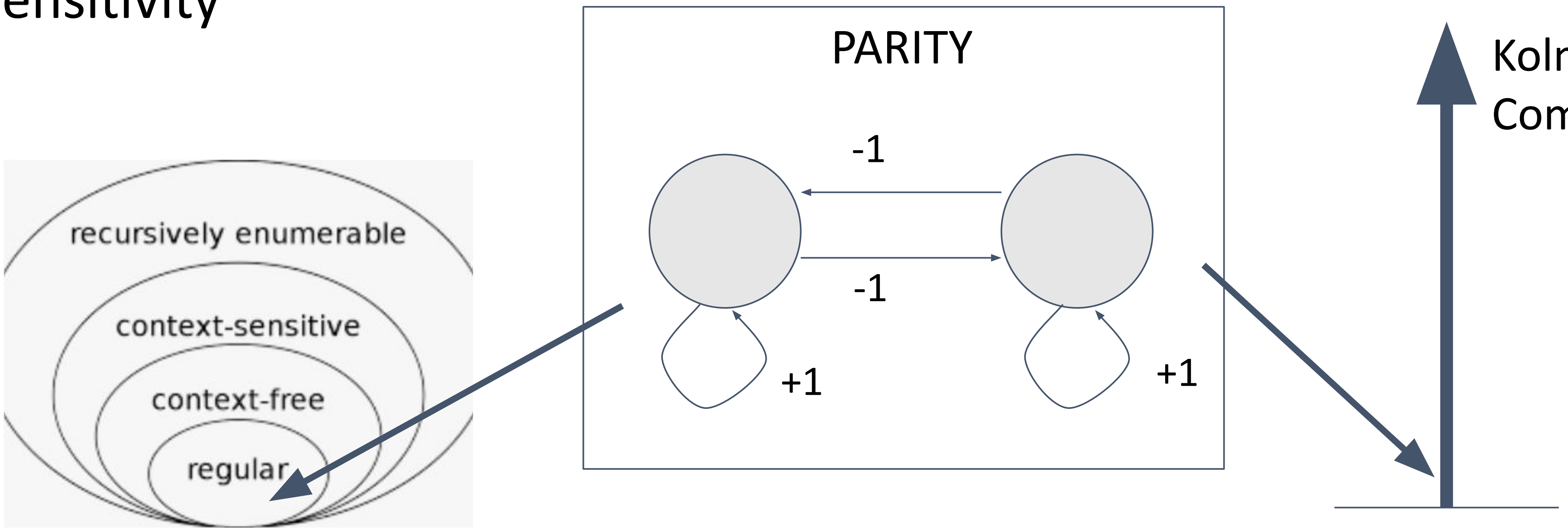
Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity





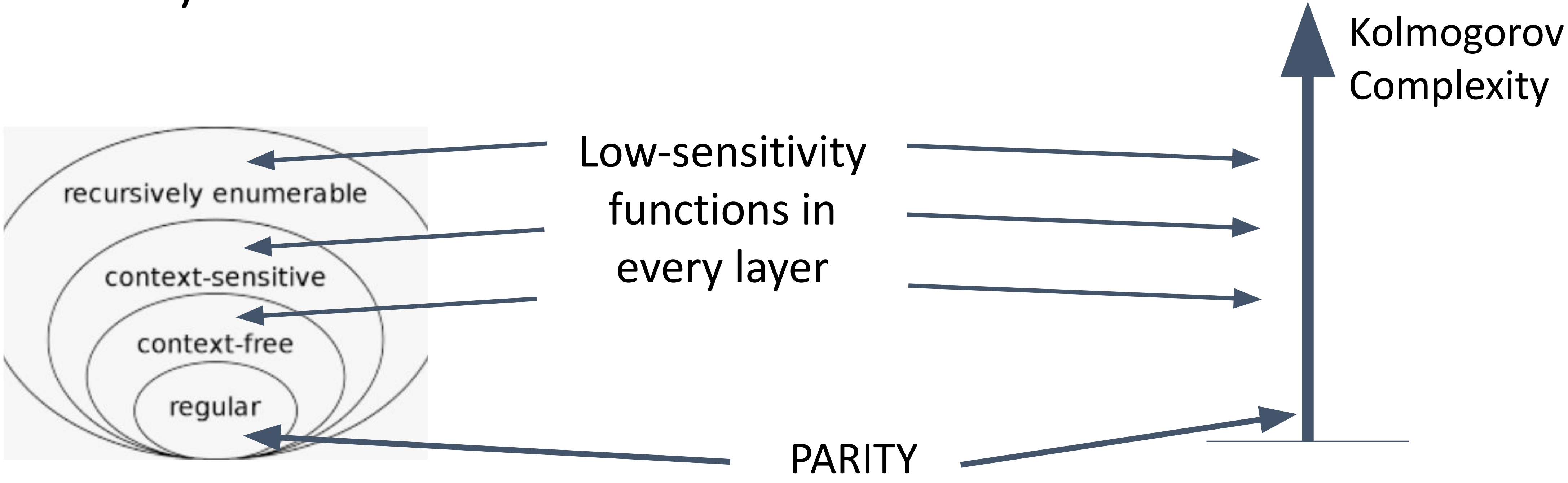
Other Complexity Measures

Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity



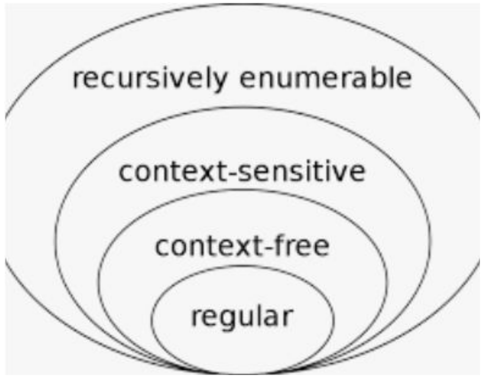
Other Complexity Measures

Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity

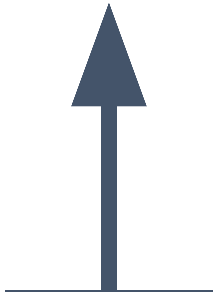


Other Complexity Measures

Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity



Kolmogorov Complexity

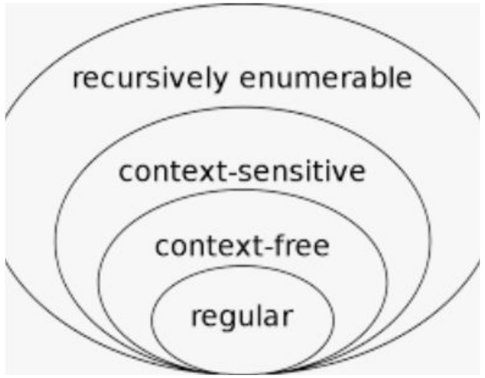


Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Other Complexity Measures

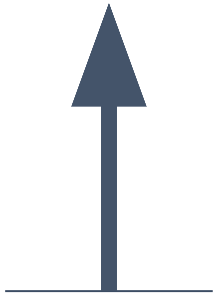
Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity



asymptotic worst-case complexity

not defined for individual inputs

Kolmogorov Complexity

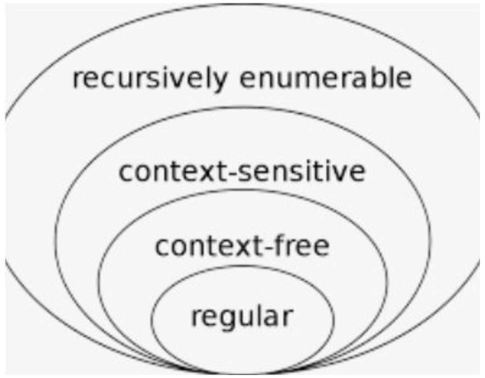


Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot{\cup} \ldots \dot{\cup} P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Other Complexity Measures

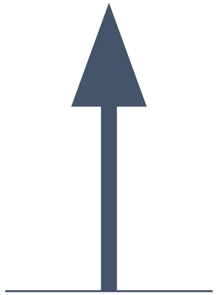
Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity



asymptotic worst-case complexity

not defined for individual inputs

Kolmogorov Complexity



uncomputable

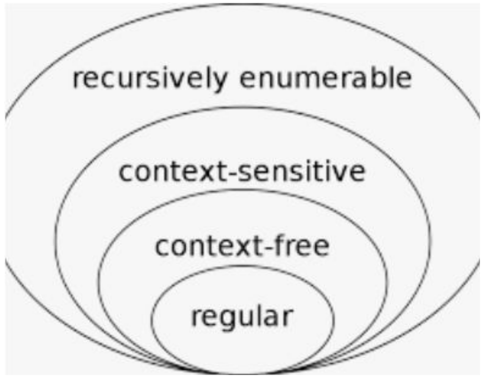
only asymptotically defined

Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

Other Complexity Measures

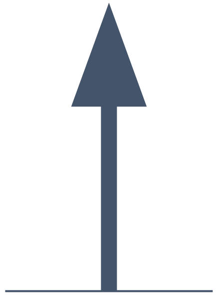
Chomsky Hierarchy and Kolmogorov Complexity are orthogonal to sensitivity



asymptotic worst-case complexity

not defined for individual inputs

Kolmogorov Complexity



uncomputable

only asymptotically defined

Sensitivity

$$bs(f,x) := \max_{k, P_1 \dot\cup \ldots \dot\cup P_k} \sum_{i=1}^{k} s(f,x,P_i)$$

estimated on individual inputs

takes input distribution into account

Conjecture about Degree and Block Sensitivity

Conjecture 2. *Define $bs(f,x,P)$ as in (1).*

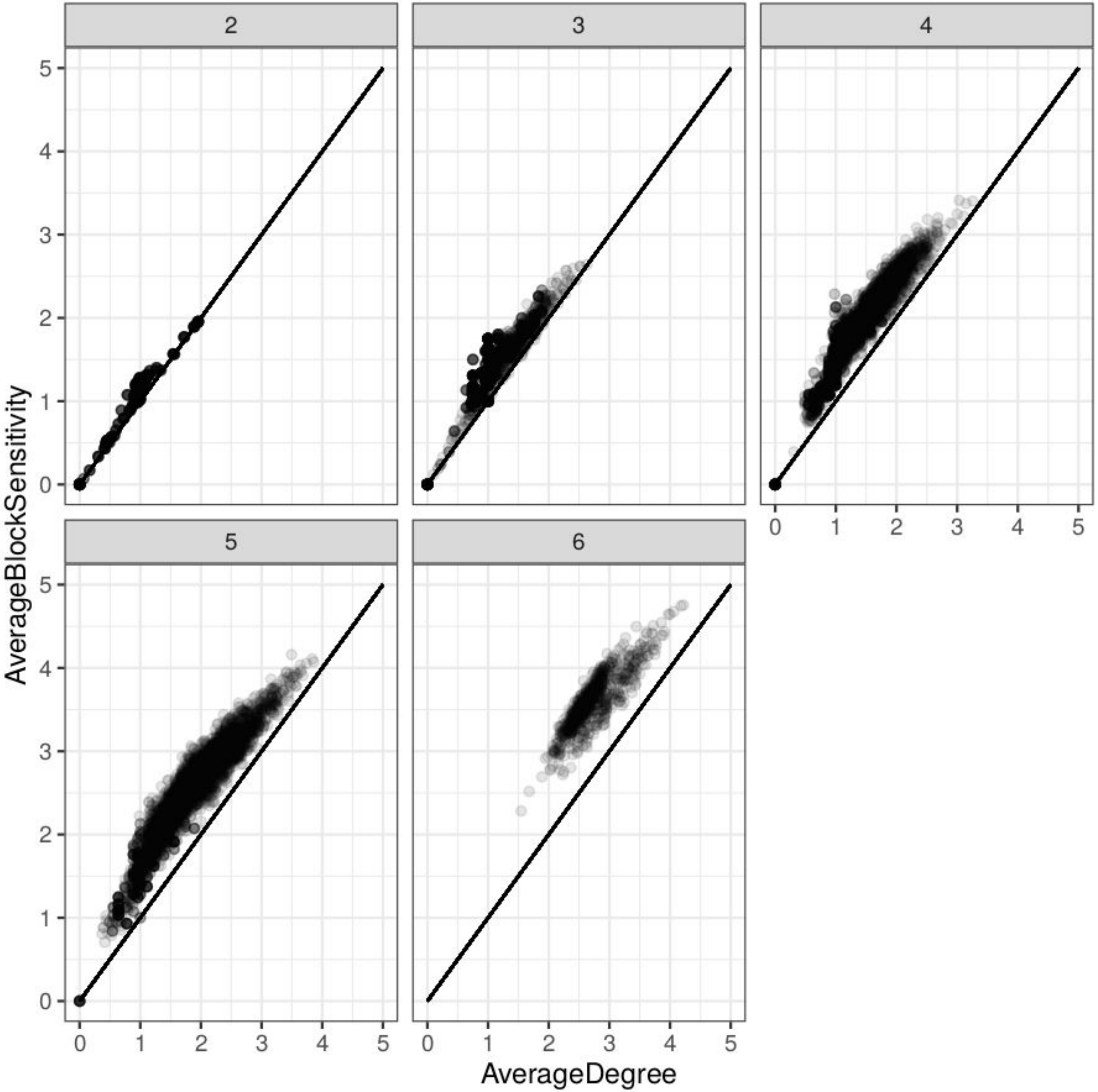
For $d \in [n]$, let $\Pi_d f$ be the orthogonal projection of f onto the subspace of degree-d polynomials in $L^2(\{-1,1\}^n, \mathbb{P})$.

Define the "average degree" to be

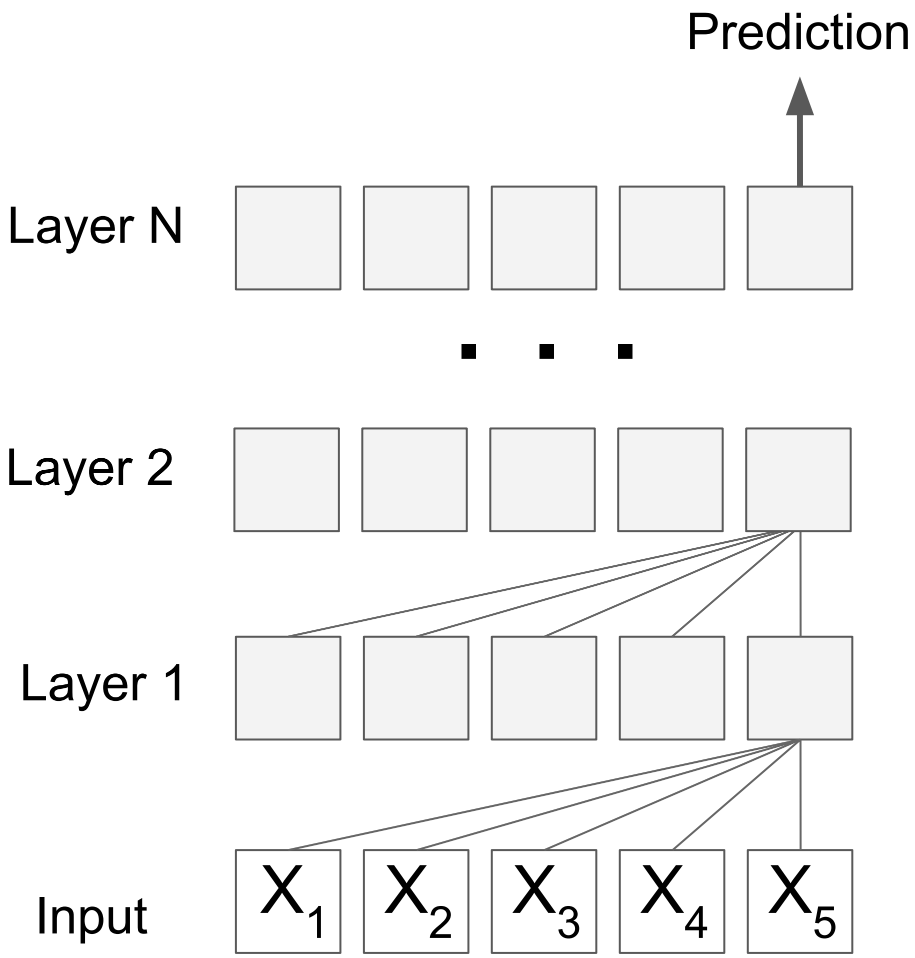
$$adeg(f) := \sum_{d=1}^{n} d \cdot \|\Pi_d f - \Pi_{d-1} f\|_2^2 \tag{3}$$

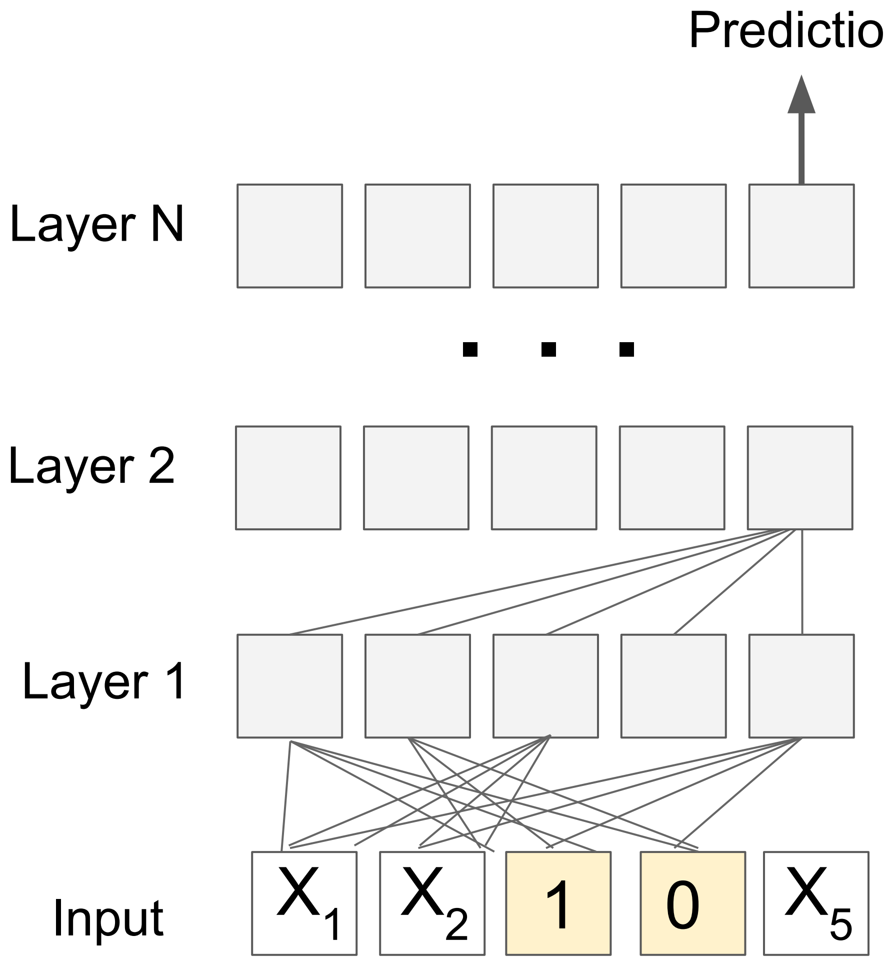
*Then the **conjecture** is:*

$$adeg(f) \leq \mathop{\mathbb{E}}_{x \sim \mathbb{P}} bs(f,x) \tag{4}$$

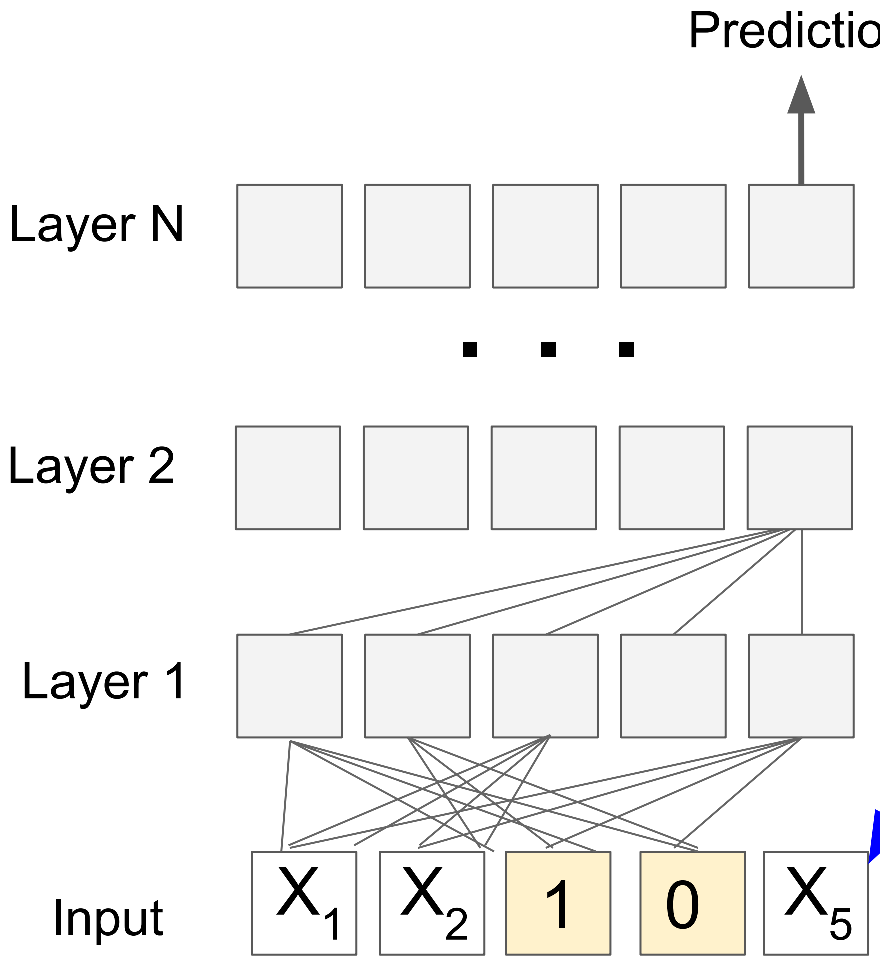

2
3
4
5
6
AverageBlockSensitivity
AverageDegree
0
1
2
3
4
5

Proof for Transformers and PARITY


Prediction
Layer N
. . .
Layer 2
Layer 1
Input
X1
X2
X3
X4
X5



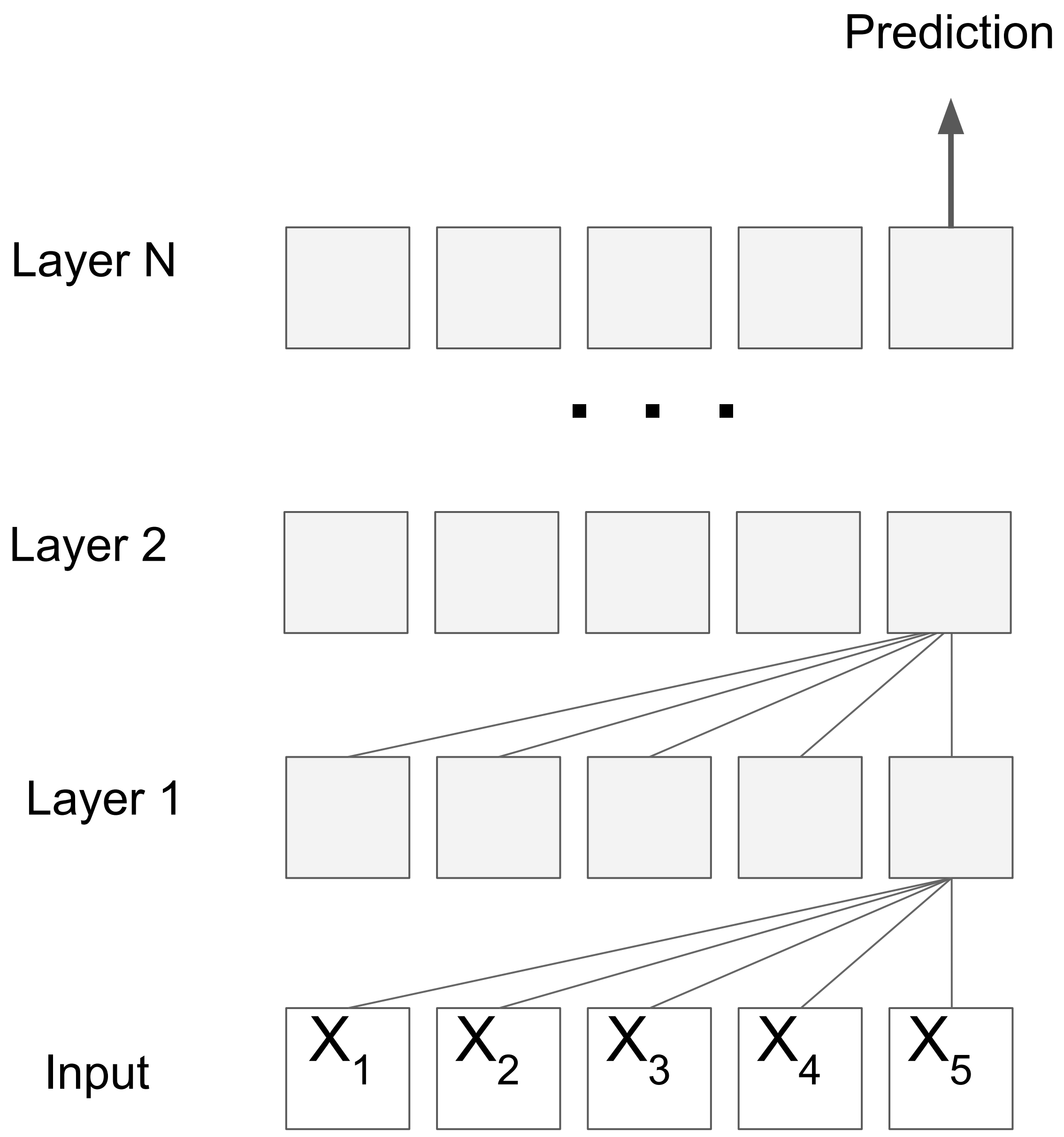
Idea: Some input bits will be ignored, since their attention weights are always smaller than those of the fixed bits.

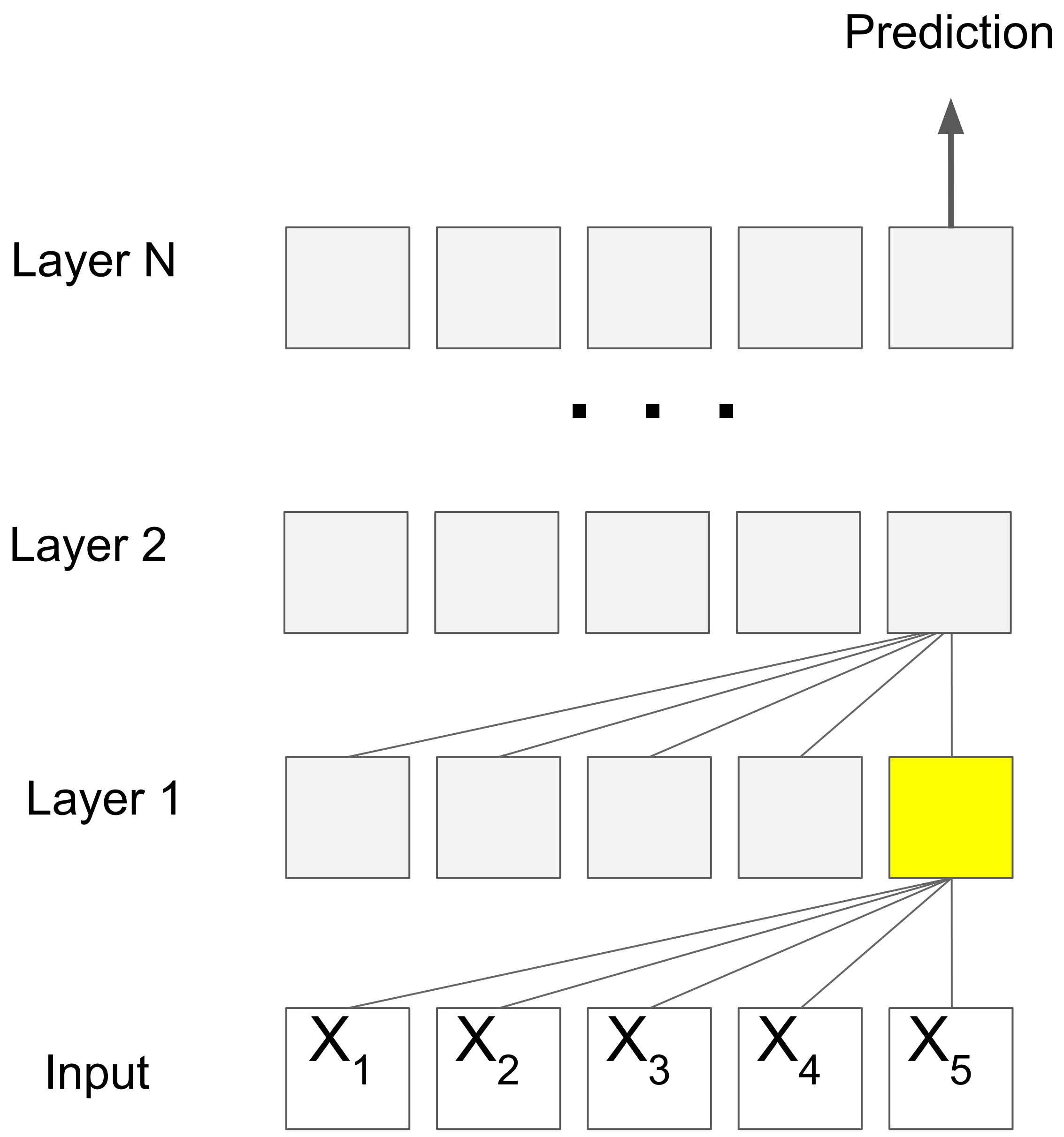


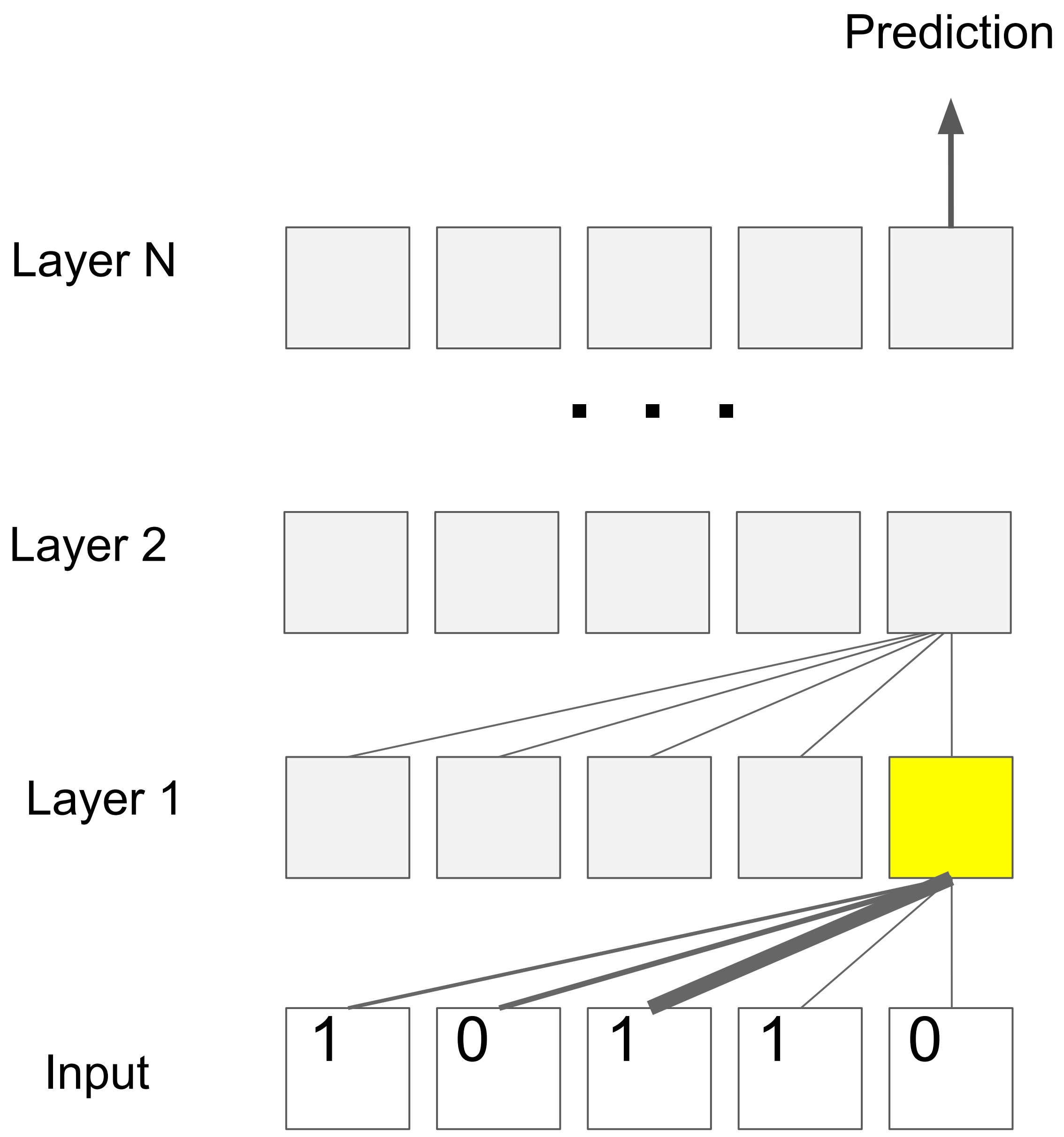
The entire network ignores this bit!

Consequence: It could not have modeled PARITY, since every bit matters for PARITY.

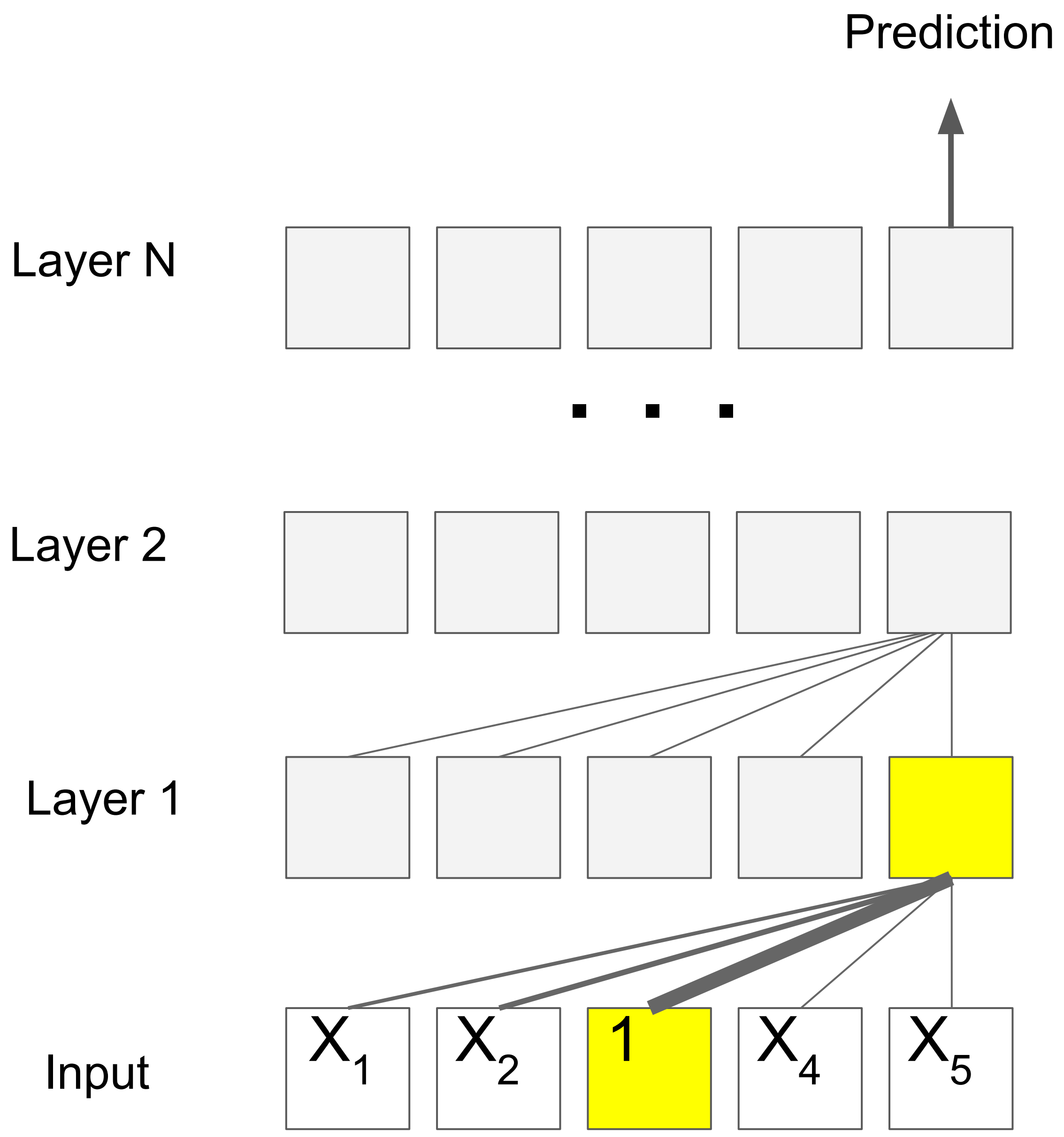
(Similar proof works for $DYCK_2$)


Prediction
Layer N
. . .
Layer 2
Layer 1
Input
X1
X2
X3
X4
X5

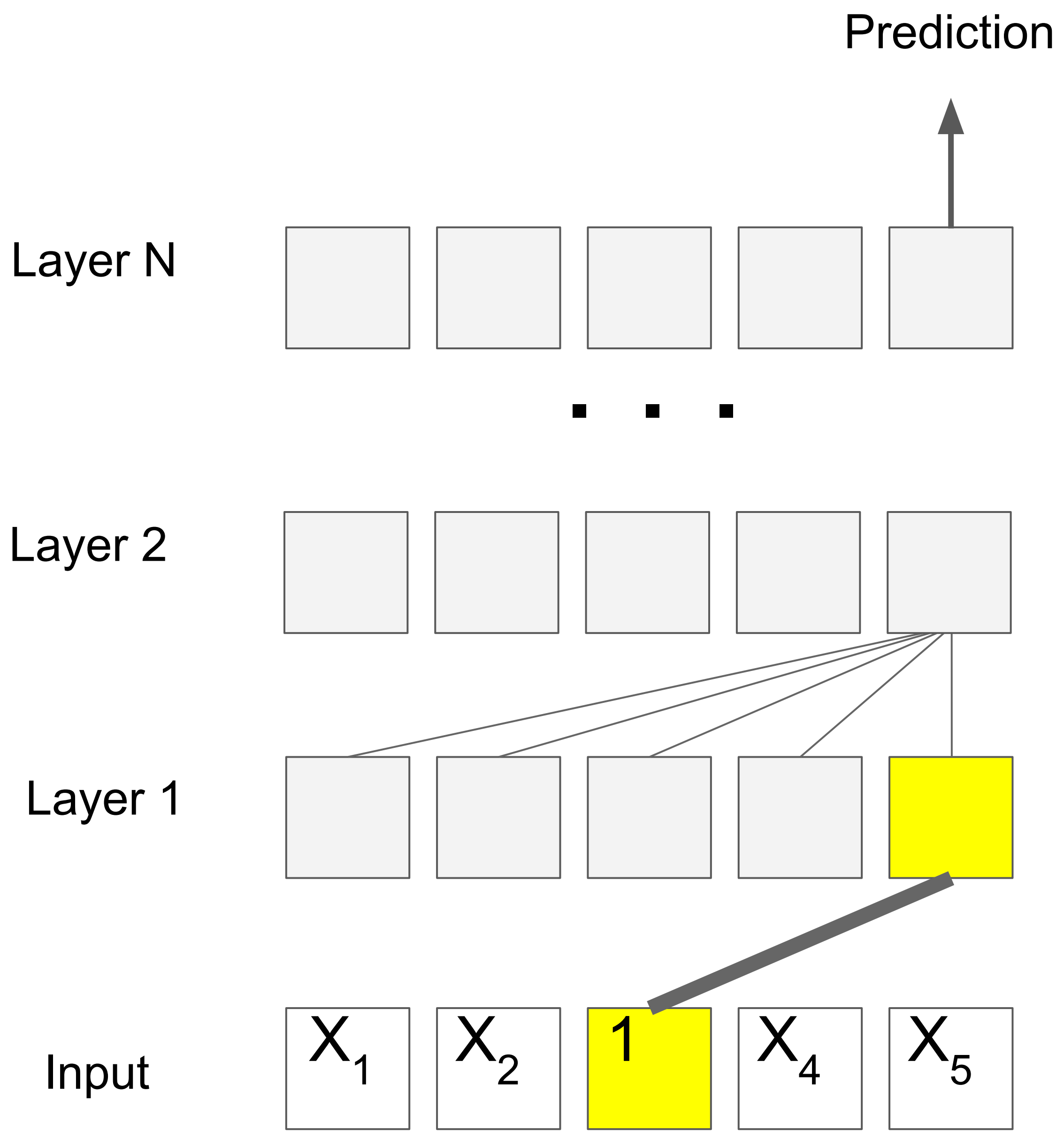

Prediction
Layer N
. . .
Layer 2
Layer 1
Input
X1
X2
X3
X4
X5



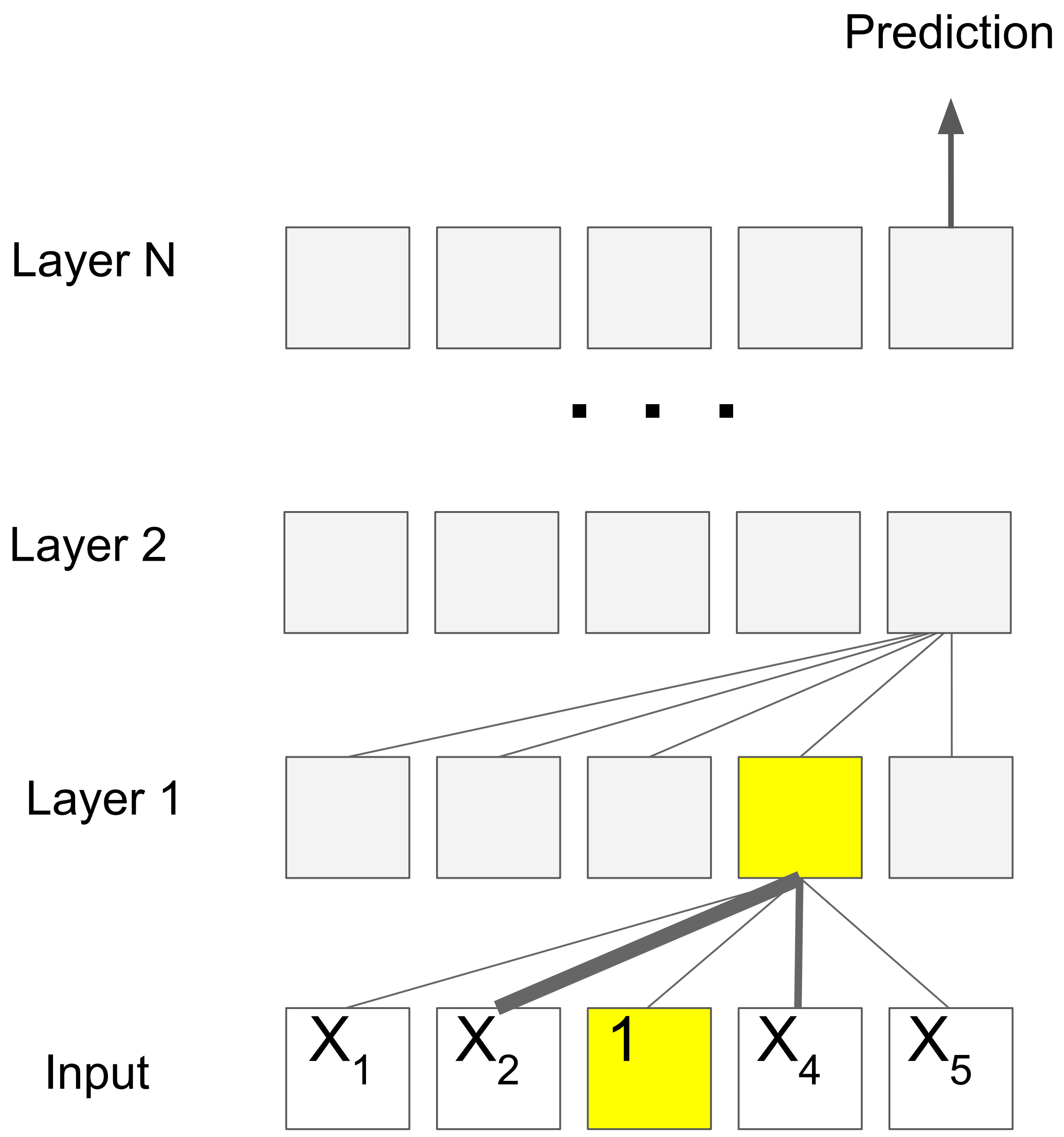
For each input bit, imagine the highest possible attention value.



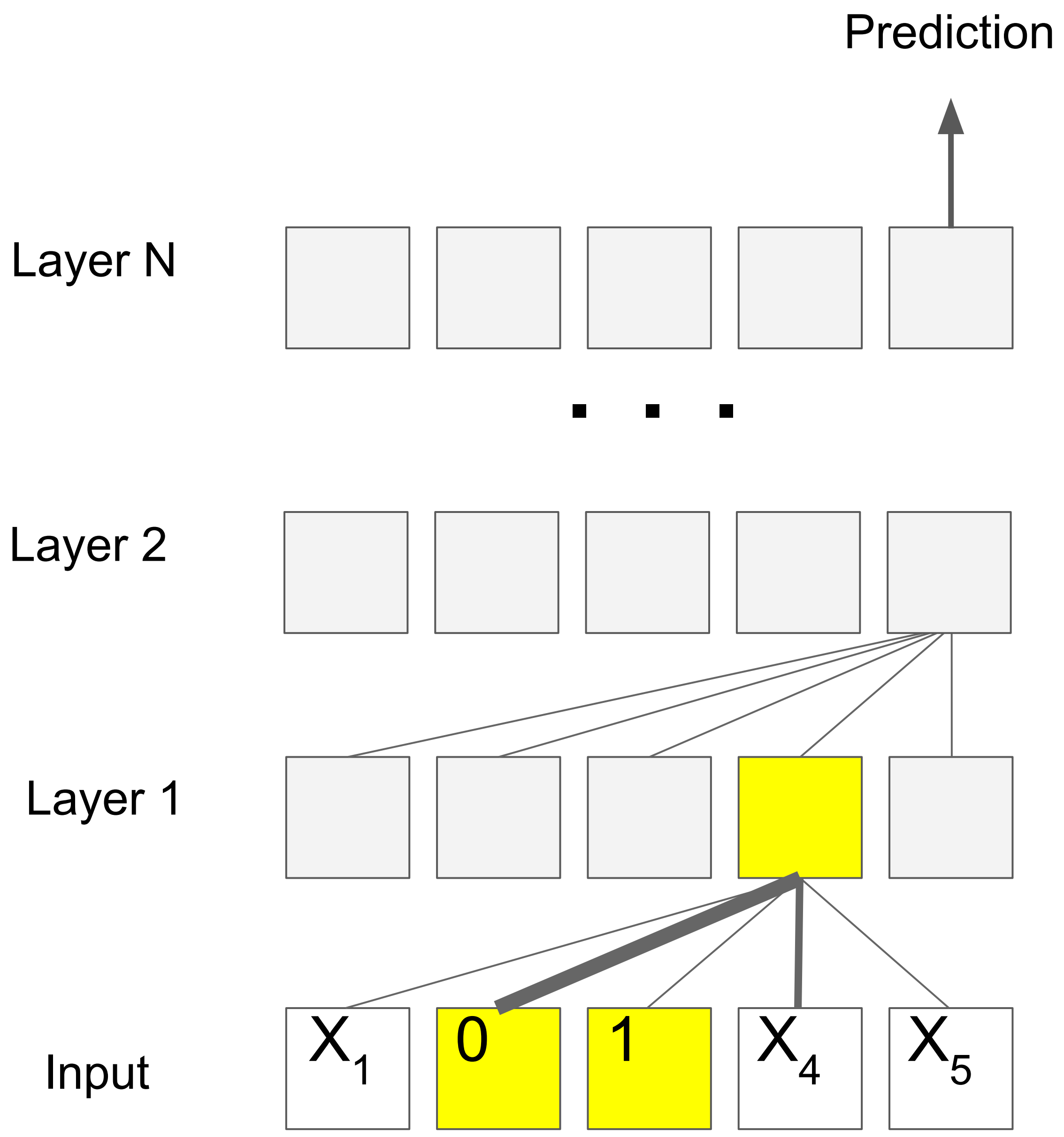
By fixing one input, we can make the head ignore all remaining input bits.



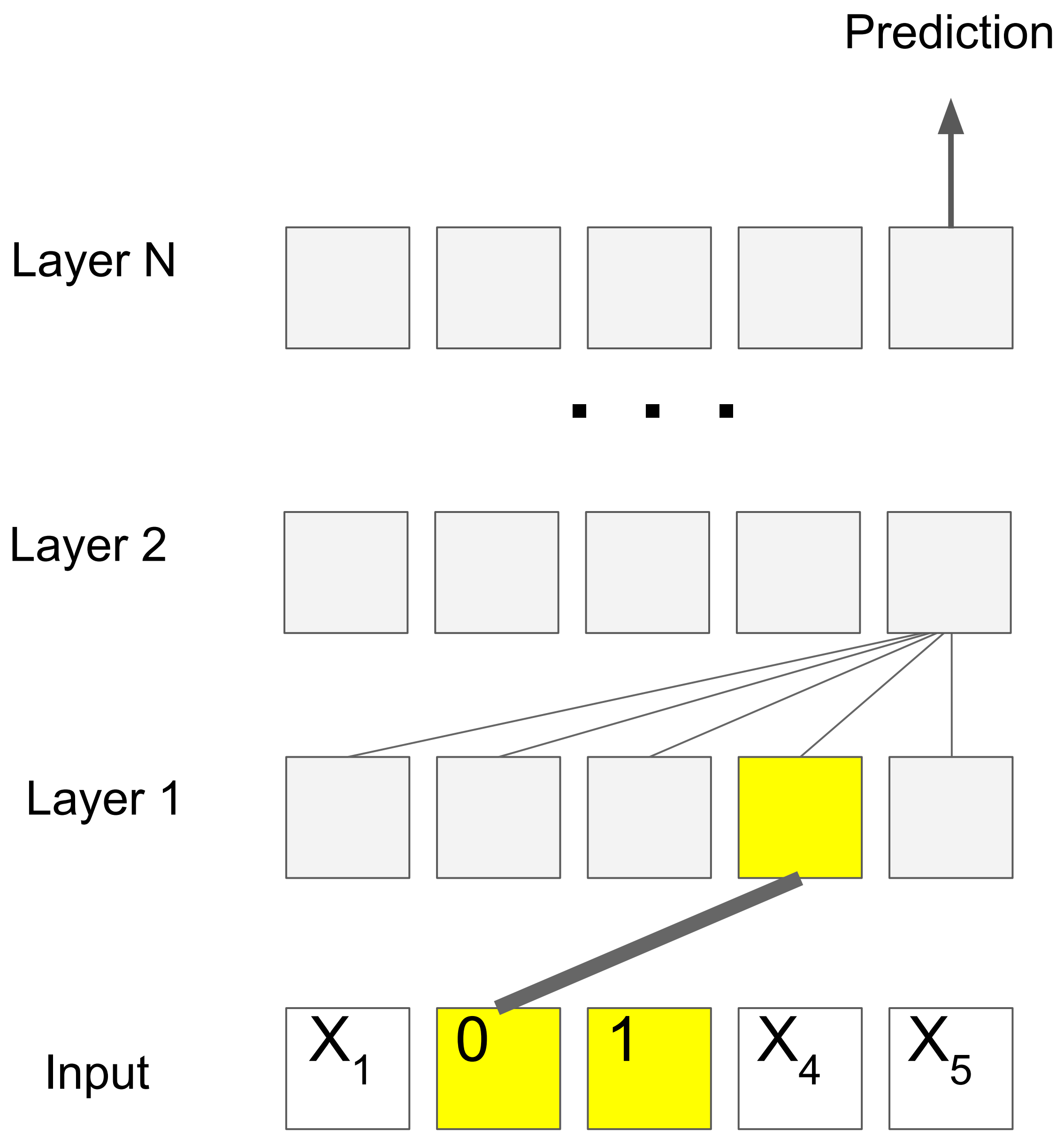
By fixing one input, we can make the head ignore all remaining input bits.



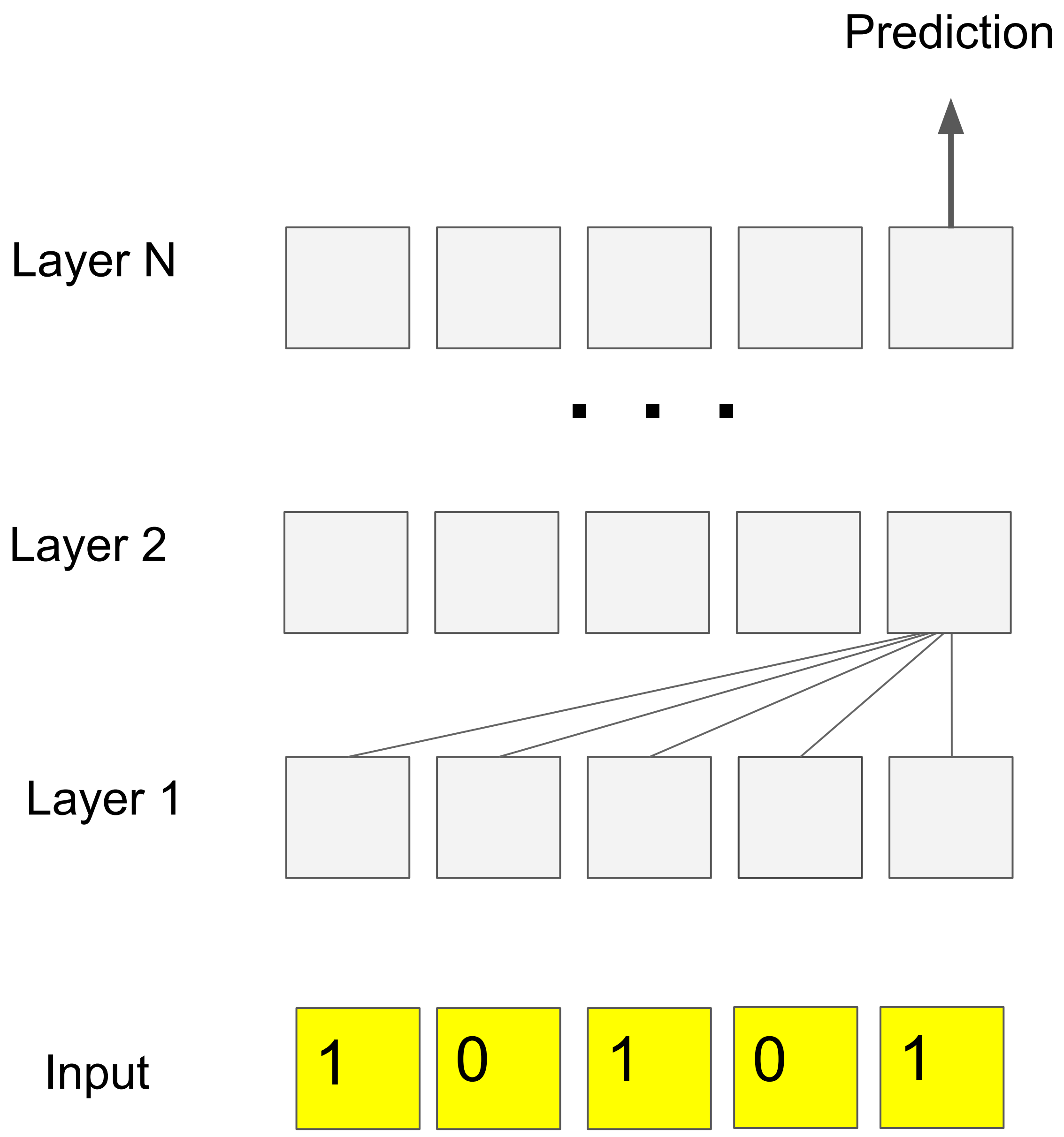
Now let's repeat this for every Layer 1 head.



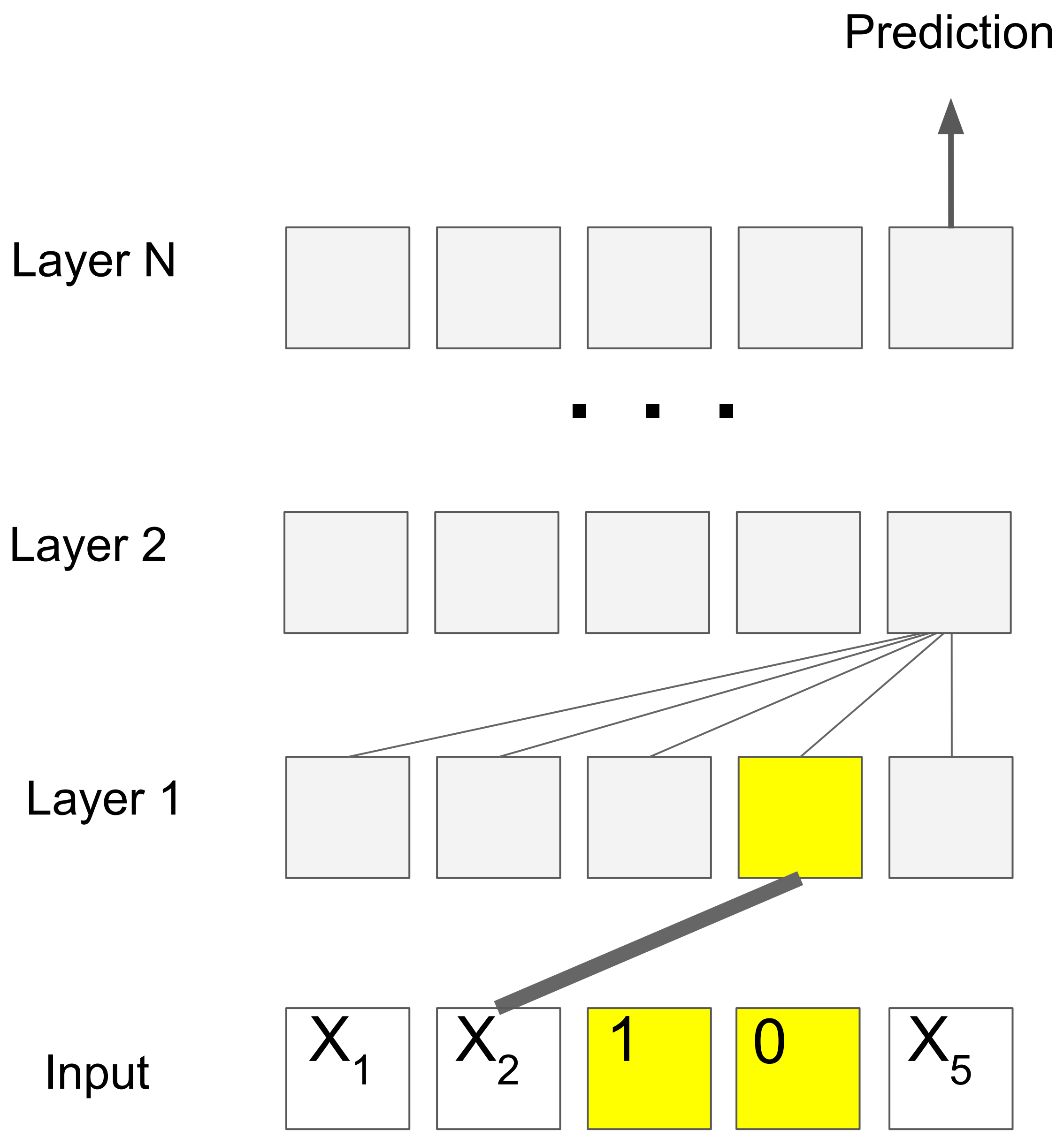
Now let's repeat this for every Layer 1 head.



Now let's repeat this for every Layer 1 head.

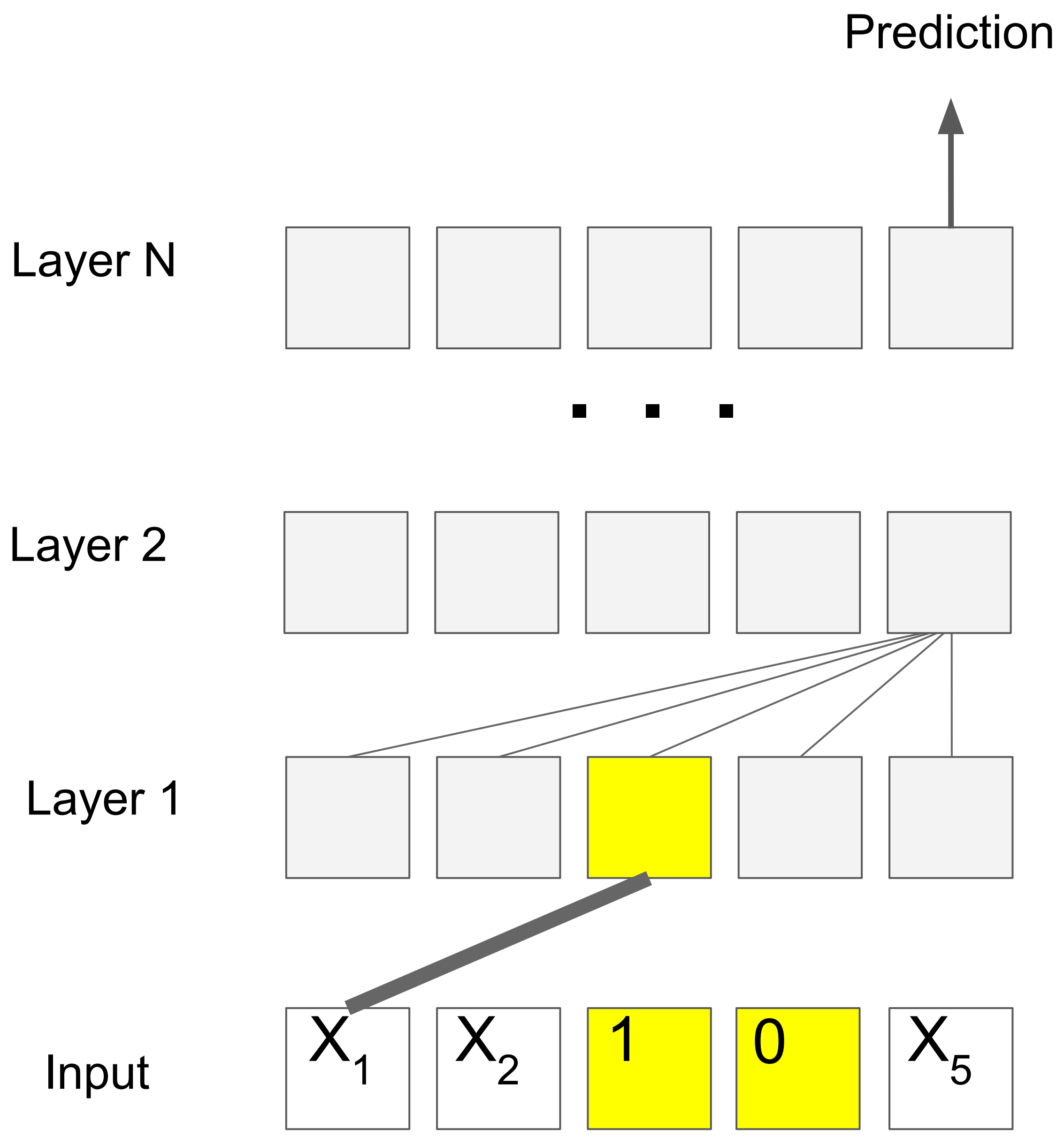


Problem: We might end up fixing all inputs.



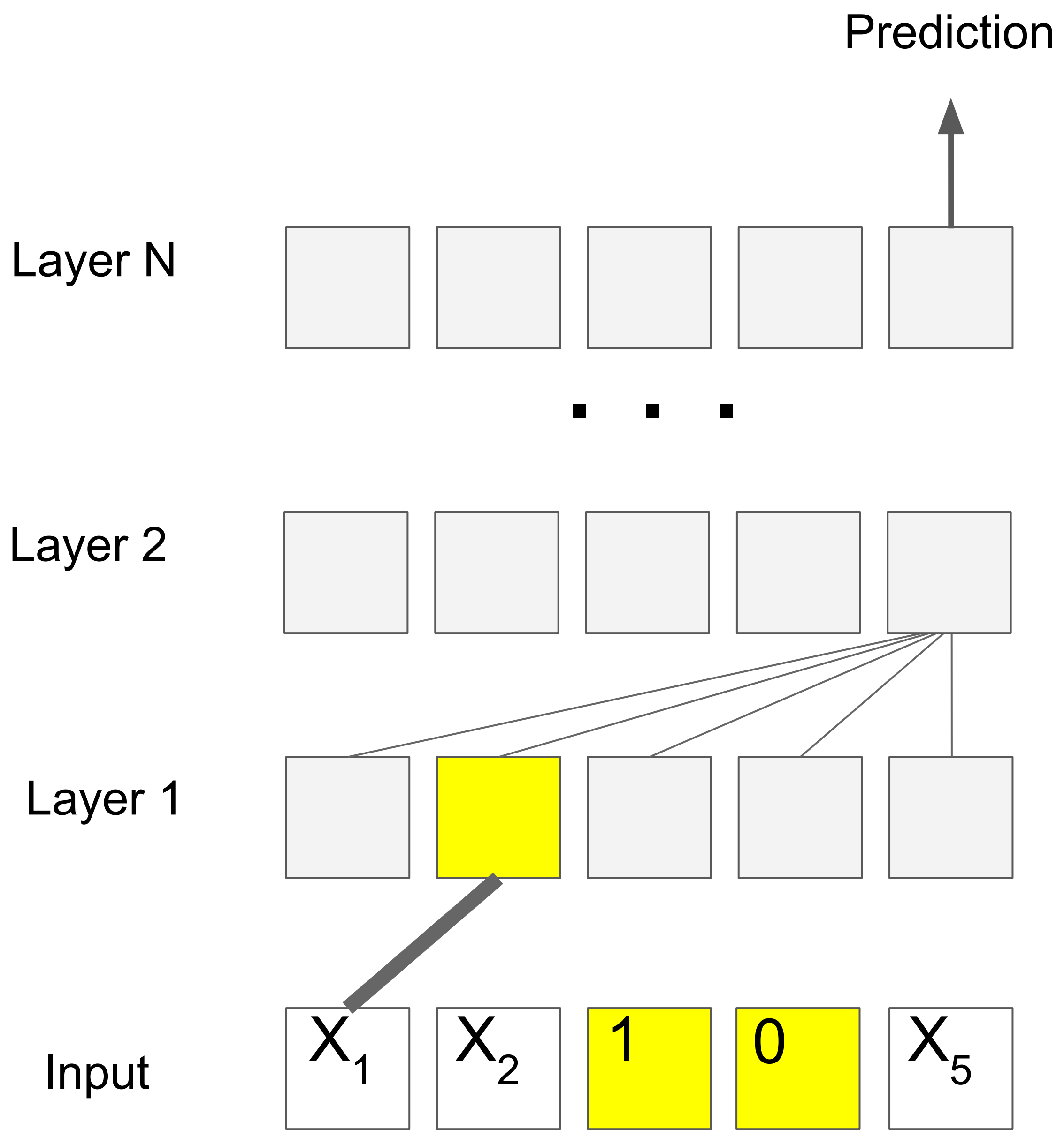
Problem: We might end up fixing all inputs.

Solution: Fix bits in such a way that each head ignores all but *k* input bits (for some constant *k*)



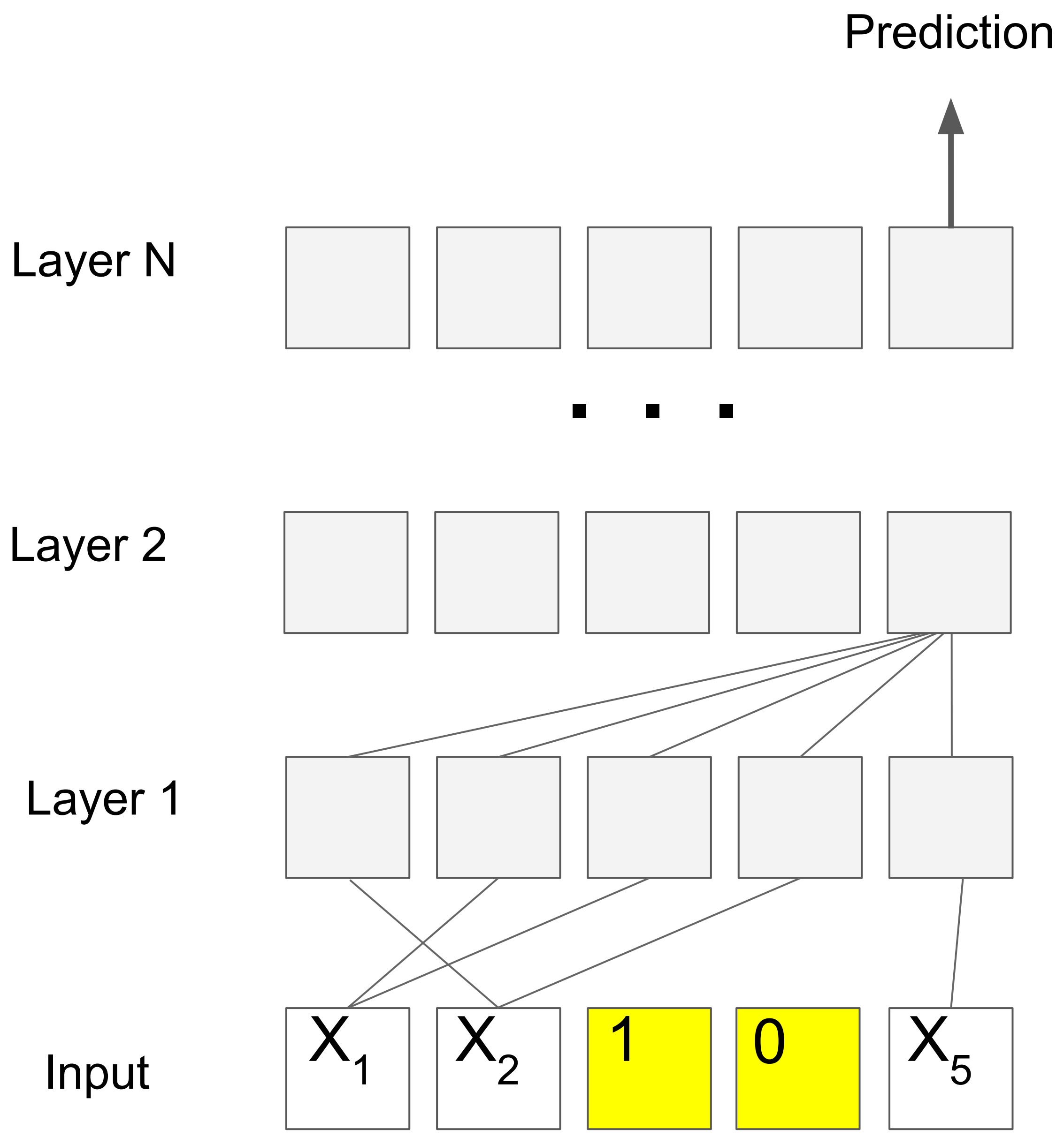
Problem: We might end up fixing all inputs.

Solution: Fix bits in such a way that each head ignores all but *k* input bits (for some constant *k*)



Problem: We might end up fixing all inputs.

Solution: Fix bits in such a way that each head ignores all but *k* input bits (for some constant *k*)

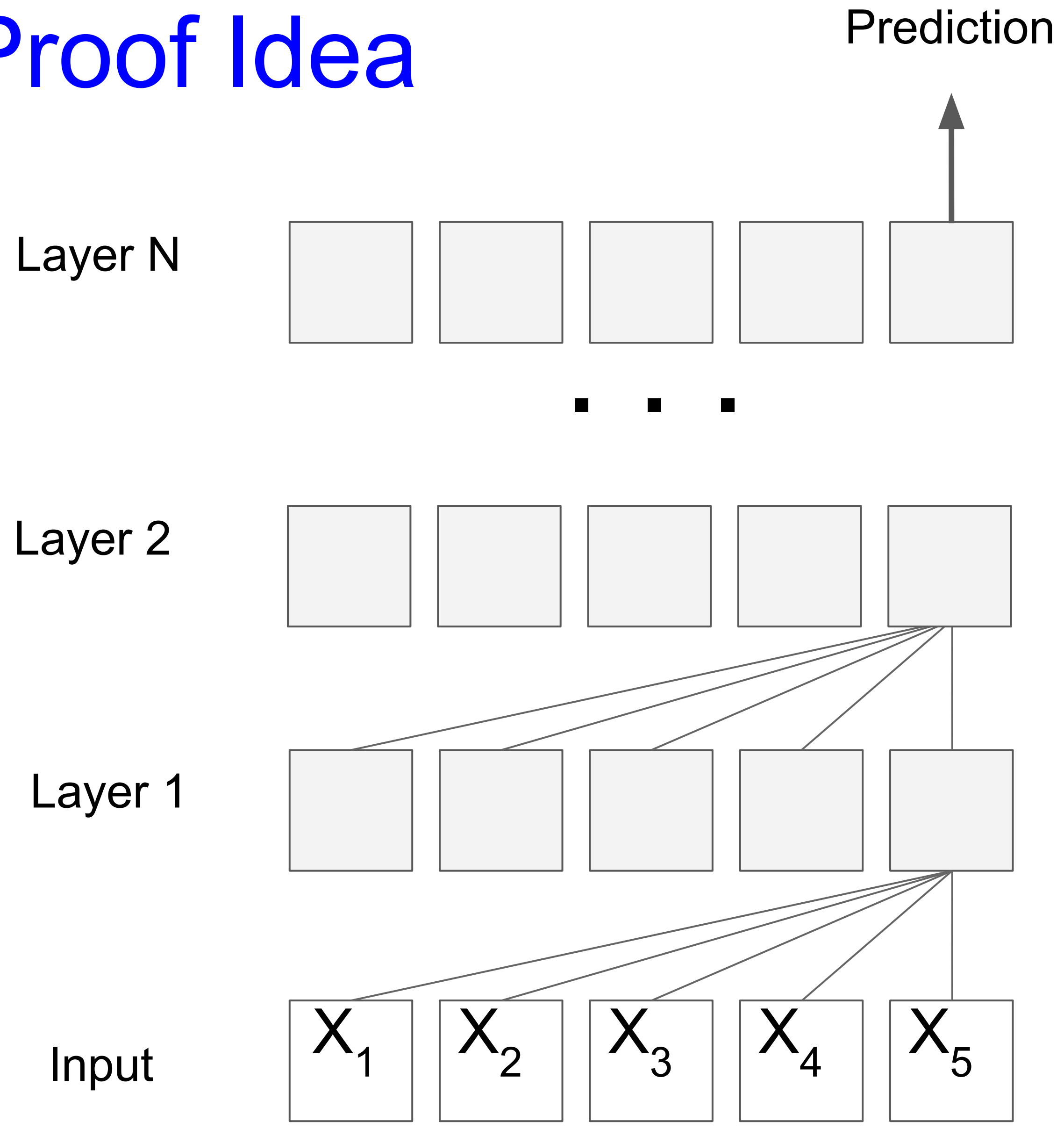


Problem: We might end up fixing all inputs.

Solution: Fix bits in such a way that each head ignores all but k input bits (for some constant k).

Can guarantee that this fixes only < 10% of bits.

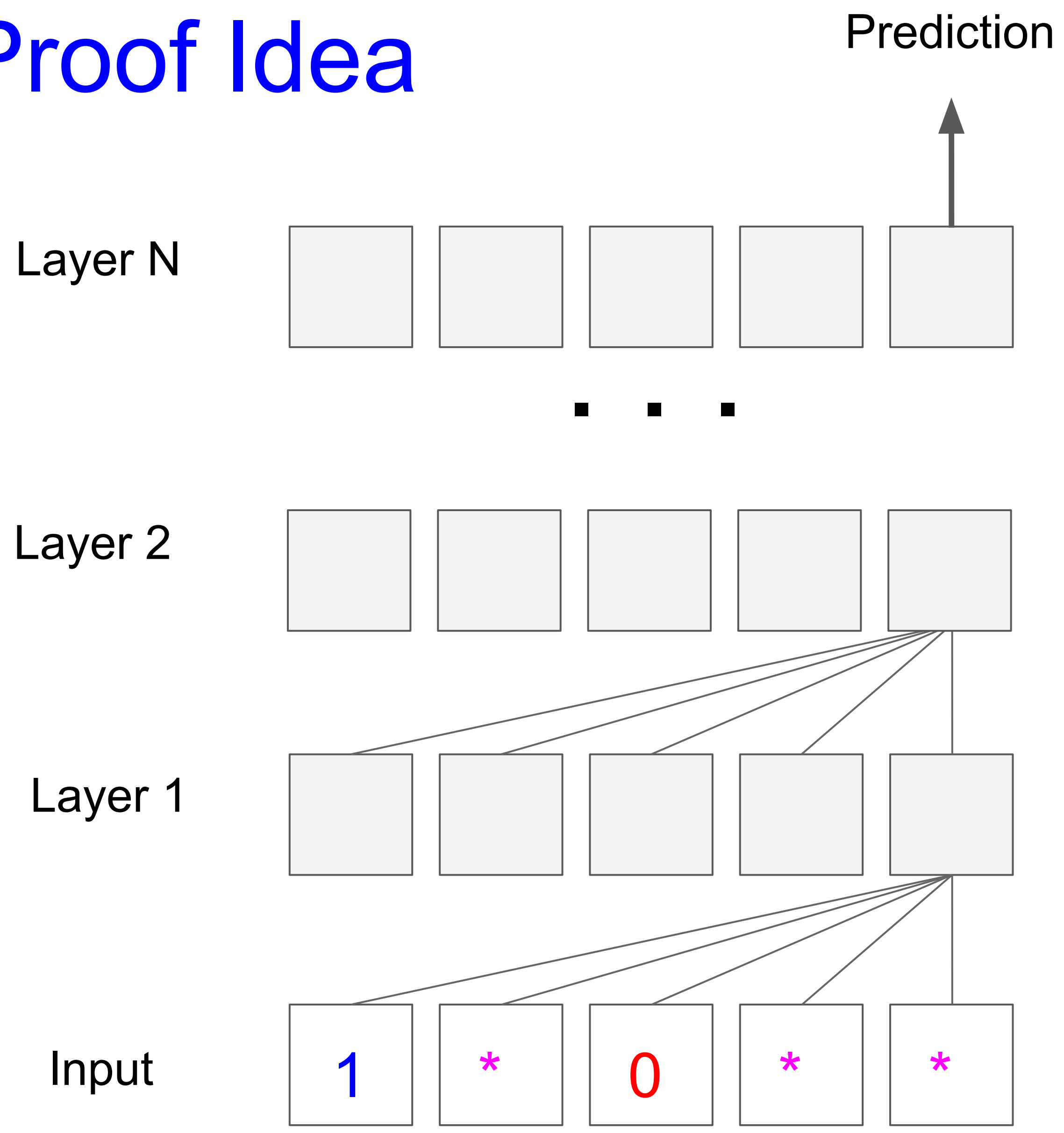
Proof Idea



Set each input i.i.d. to

* with p=95%
0 with p=2.5%
1 with p=2.5%

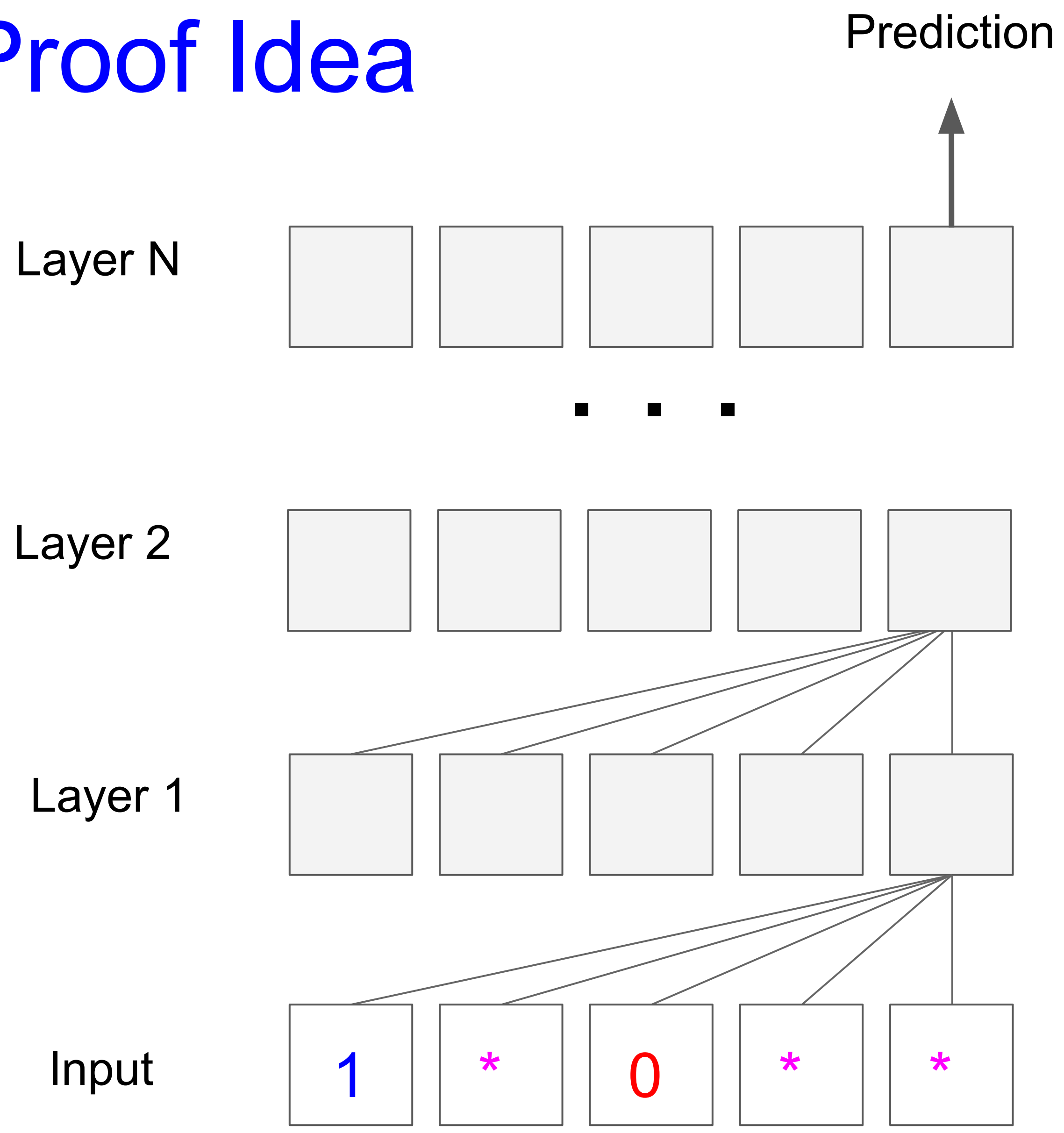
Proof Idea



Set each input i.i.d. to

* with p=95%
0 with p=2.5%
1 with p=2.5%

Proof Idea

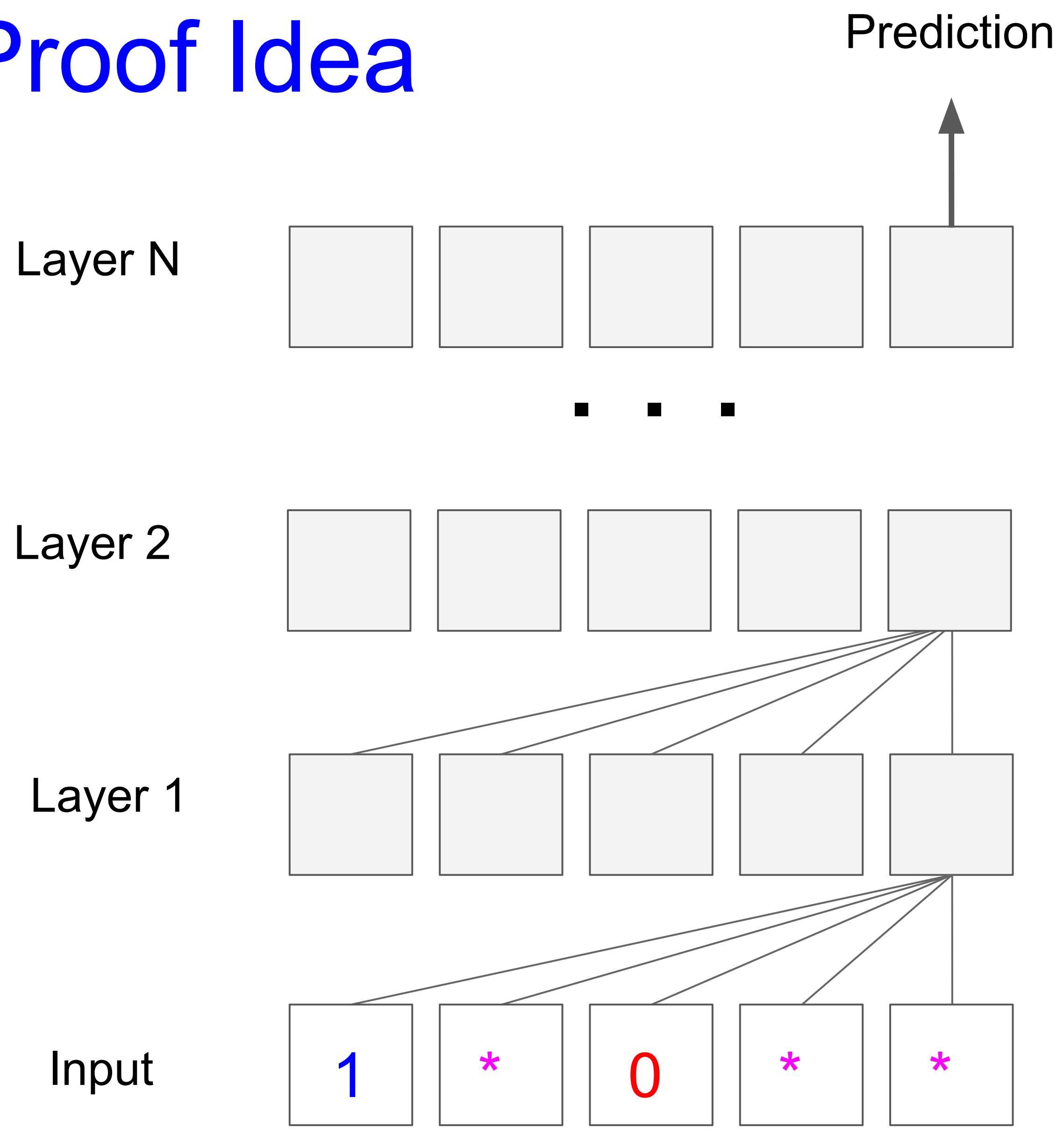


What is Probability that

1) each head depends on only k inputs, and
2) only < 10% of bits are fixed?

Enough to show that this is > 0!

Proof Idea

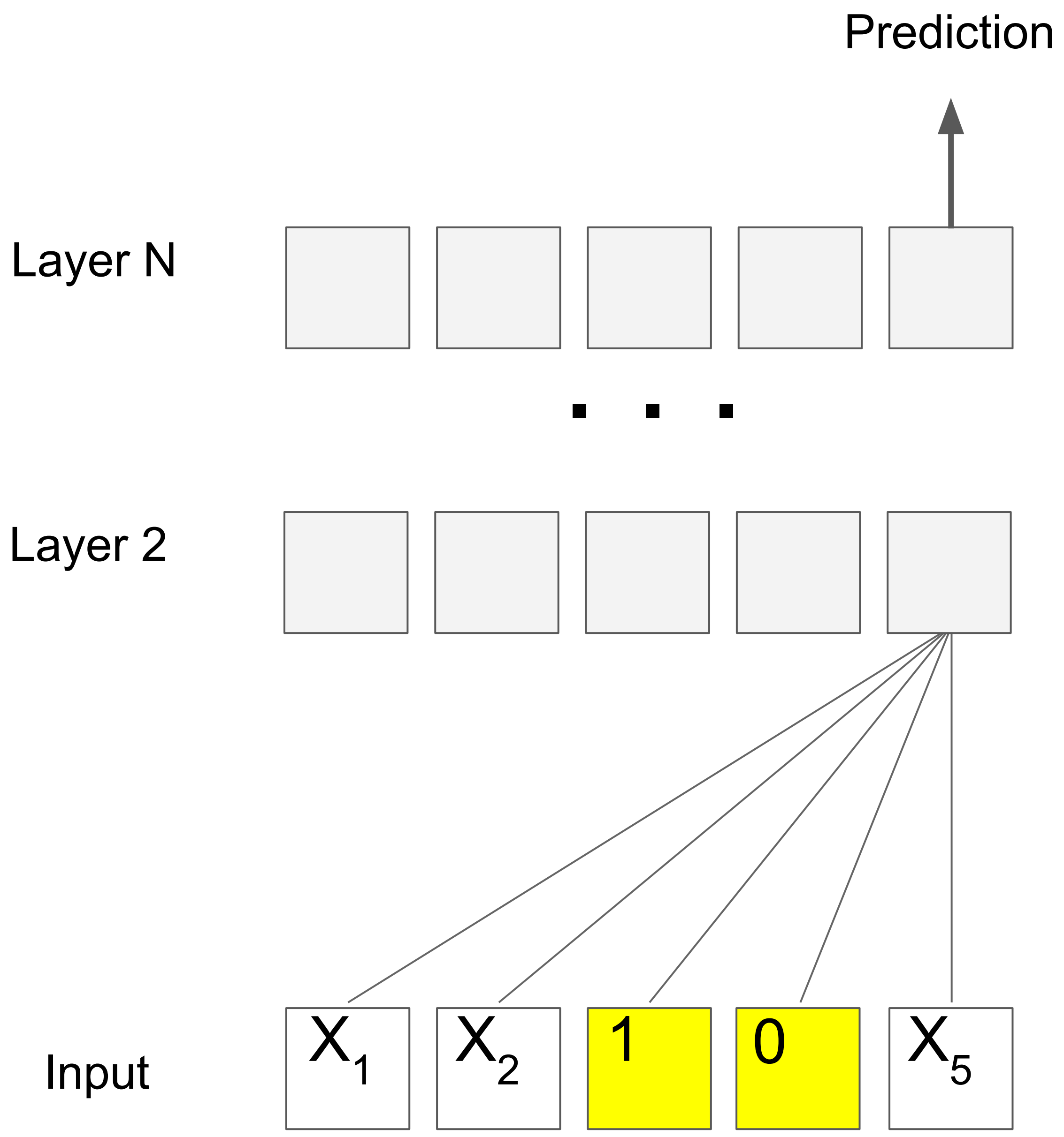


What is Probability that

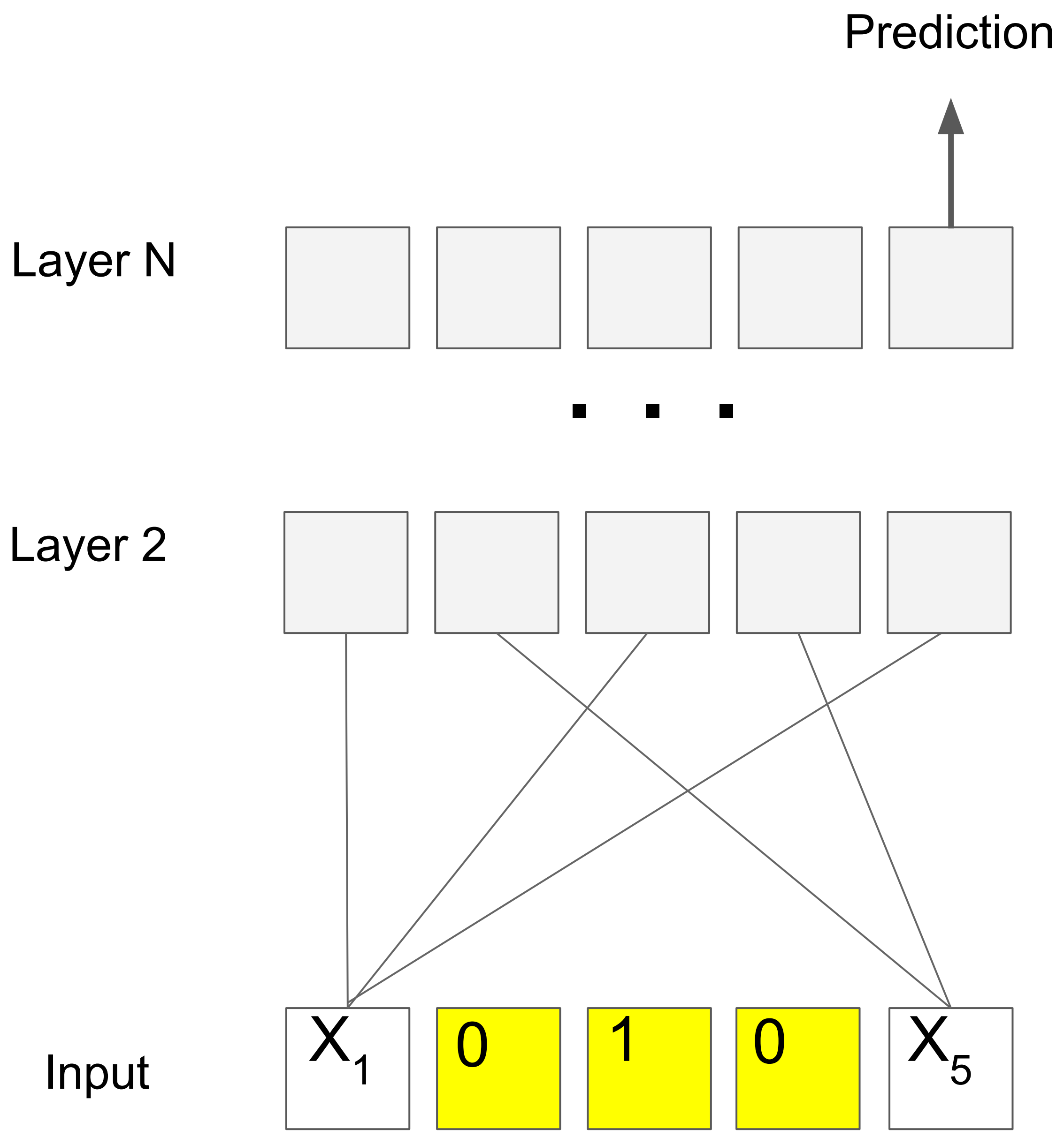
1) each head depends on only k inputs, and
2) only < 10% of bits are fixed?

Enough to show that this is > 0!

Show this by calculating for each head and combining via Lovasz Local Lemma.

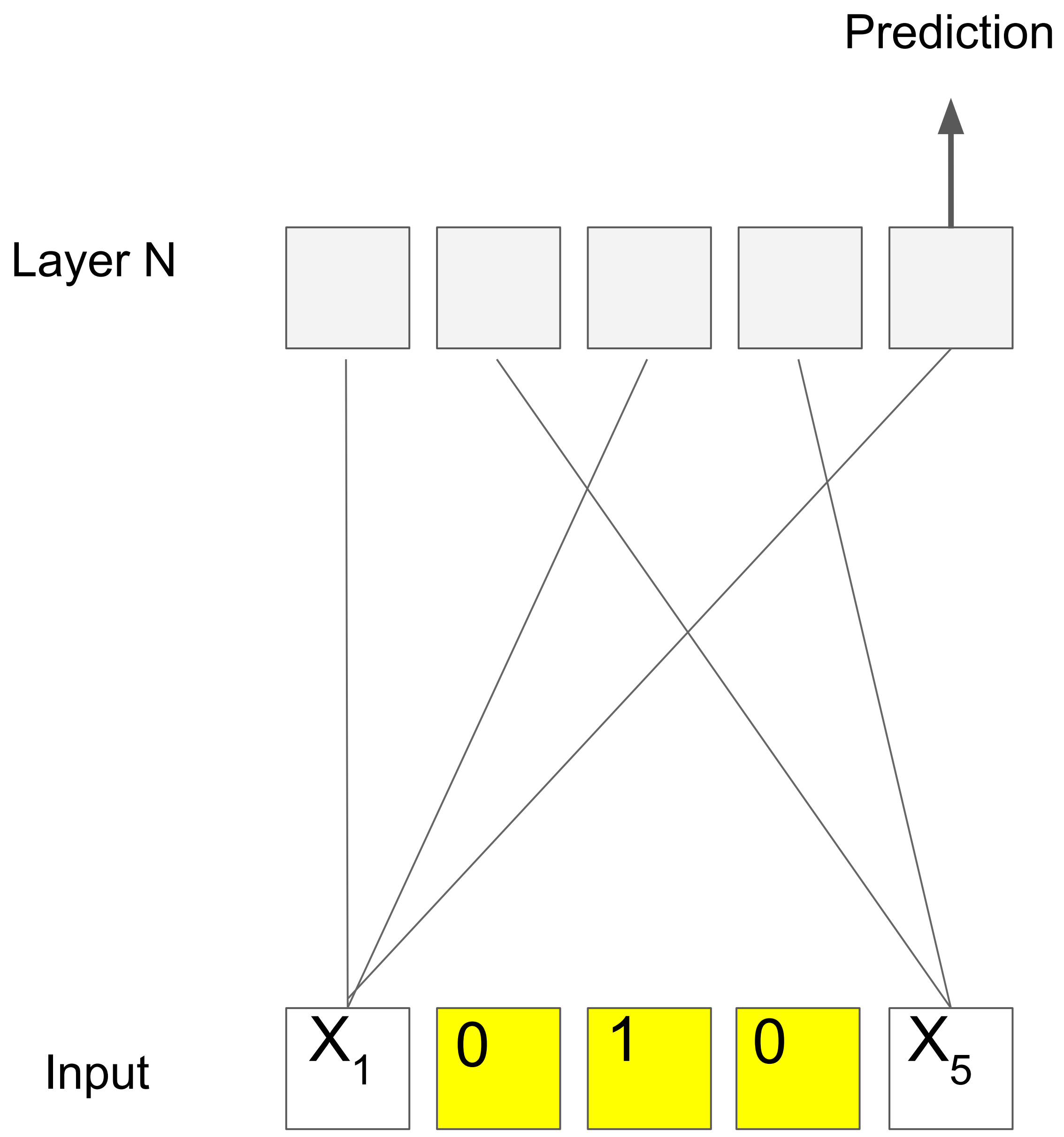


We can now fold Layer 1 into Layer 2….



We can now fold Layer 1 into Layer 2….

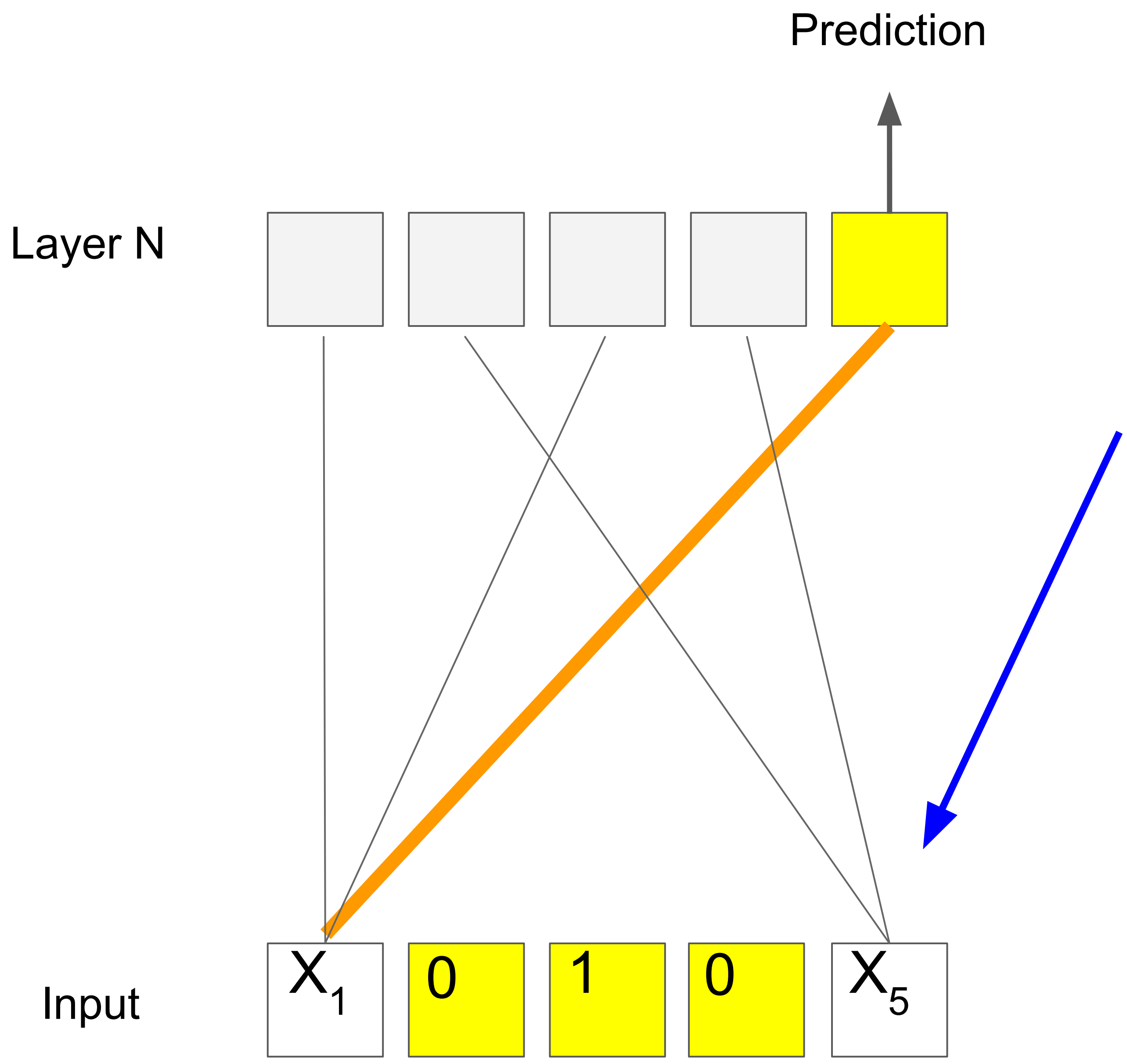
...and repeat the construction...



We can now fold Layer 1 into Layer 2….

...and repeat the construction…

...until only the final layer remains!



The prediction ignores bit X_5!

Thus, the transformer could never have modeled Parity (or $Dyck_2$).