Sensitivity as a Complexity Measure for Sequence Classification Tasks

Michael Hahn, Dan Jurafsky, Richard Futrell

EMNLP 2021
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.

<table>
<thead>
<tr>
<th></th>
<th>sentiment analysis</th>
<th>POS tagging</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
Motivation

What makes some NLP tasks harder and others easier?

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

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

sentiment analysis
POS tagging
...
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.

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

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

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

Simple Task

![Simple Task Diagram]

Difficult Task

![Difficult Task Diagram]
Idea: Tasks are difficult when they have complex decision boundaries.

Most neighbors have the same label as the point.
Sensitivity

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

**Simple Task**

Most neighbors have the **same** label as the point.

**Difficult Task**

Neighbors have the **opposite** label as the point.
Sensitivity of Boolean Functions

For a function \( f : \{0,1\}^n \rightarrow \{0,1\} \):

\[
s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \}
\]

Input string

\( x \) in \( \{0,1\}^n \)

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

For a function $f : \{0,1\}^n \to \{0,1\}$:

$$s(f,x) = \# \{\text{Hamming neighbor of } x \text{ with opposite label}\}$$

Input string $x$ in $\{0,1\}^n$

Well-studied theory with links to many areas of computational complexity

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

\[
s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \}
\]

\[
\text{XOR}(0,0,0,0,0) \rightarrow 0
\]

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

\[ s(f,x) = \# \text{\{Hamming neighbor of } x \text{ with opposite label\}} \]

\[
\begin{align*}
\text{XOR}(0,0,0,0,0) & \rightarrow 0 \\
\text{XOR}(1,0,0,0,0) & \rightarrow 1
\end{align*}
\]

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[
\begin{align*}
\text{XOR}(0,0,0,0,0) & \rightarrow 0 \\
\text{XOR}(1,0,0,0,0) & \rightarrow 1 \\
\text{XOR}(0,1,0,0,0) & \rightarrow 1
\end{align*}
\]

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

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[
\begin{align*}
\text{XOR}(0,0,0,0,0) & \rightarrow 0 \\
\text{XOR}(1,0,0,0,0) & \rightarrow 1 \\
\text{XOR}(0,1,0,0,0) & \rightarrow 1 \\
\text{XOR}(0,0,1,0,0) & \rightarrow 1 \\
\end{align*}
\]

\[ \ldots \]

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

\begin{align*}
s(f, x) &= \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \\
\text{XOR}(0,0,0,0,0) &\rightarrow 0 \\
\text{XOR}(1,0,0,0,0) &\rightarrow 1 \\
\text{XOR}(0,1,0,0,0) &\rightarrow 1 \\
\text{XOR}(0,0,1,0,0) &\rightarrow 1 \\
&\ldots
\end{align*}
Sensitivity in NLP

Alphabet $\Sigma$ (e.g. words, BPE, characters)
Sensitivity in NLP

Alphabet $\Sigma$ (e.g. words, BPE, characters)

Distribution $\mathcal{D}$ over the set $\Sigma^*$ 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 in NLP

Alphabet $\Sigma$ (e.g. words, BPE, characters)
Distribution $\prod$ over the set $\Sigma^*$ of finite strings
Classification task = Function $f : \Sigma^* \rightarrow [-1,1]$
Sensitivity in NLP

Alphabet $\Sigma$ (e.g. words, BPE, characters)

Distribution $\Pi$ over the set $\Sigma^*$ 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 for Boolean Functions

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]
Sensitivity for Boolean Functions

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while staying within input distribution?”} \]
Sensitivity in NLP

\[ s(f,x) = \text{"In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]

\[ \textit{this was hilarious} \ +1 \]
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]

\[ \text{this was hilarious} \quad ^{+1} \]
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?“ \]

:\text{this was hilarious} +1
\text{it was hilarious}
\text{this was hilarious}
\text{she was hilarious}

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f,x) = \text{"In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]

Samples from generative distribution

In our implementation:

- XLNet (Yang et al 2019)
- u-PMLM (Liao et al 2020)

\[ \text{this was hilarious} \quad +1 \]
\[ \text{it was hilarious} \quad +1 \]
\[ \text{this was hilarious} \quad +1 \]
\[ \text{she was hilarious} \quad +1 \]

Annotated by trained model or human (see paper)
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\” \]

\[ \begin{align*}
\text{this was hilarious} &+1 \\
\text{it was hilarious} &+1 \\
\text{this was hilarious} &+1 \\
\text{she was hilarious} &+1 \\
\end{align*} \]

Variance = 0

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod \text{?”} \]

“Hard to flip the label by changing this word”

0

\[ \text{this was hilarious} \]

\[ \text{+1} \]

\[ \text{it was hilarious} \]

\[ \text{+1} \]

\[ \text{this was hilarious} \]

\[ \text{+1} \]

\[ \text{she was hilarious} \]

\[ \text{+1} \]

\[ \text{+1} \]

Variance = 0

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\” \]

0

\textit{this was hilarious} +1
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\" \]

\[ 0 \]

\[ \text{this was hilarious} \quad +1 \]

Samples from generative distribution

In our implementation:

XLNet (Yang et al 2019)

u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\" } \]

0

\textit{this was hilarious}  \hspace{1cm} +1

\textit{this was hilarious}  \hspace{1cm} +1

\textit{this is hilarious}  \hspace{1cm} +1

\textit{this wasn’t hilarious}  \hspace{1cm} -1

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]

“This was hilarious”

Can flip the label by changing this word

\[
\begin{array}{ccc}
0 & 0.88 & +1 \\
\end{array}
\]

\text{this was hilarious} +1

\text{this was hilarious} +1

\text{this is hilarious} +1

\text{this wasn’t hilarious} -1

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)

Variance = 0.88
Sensitivity in NLP

\[ s(f,x) = \text{"In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\" } \]

\[
\begin{array}{cc}
0 & 0.88 \\
\end{array}
\]

\textit{this was hilarious} +1
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?“ \]

\[
\begin{array}{ll}
0 & 0.88 \\
\end{array}
\]

\textcolor{blue}{this was hilarious}  +1

\textcolor{red}{this was awesome}

\textcolor{red}{this was boring}

\textcolor{red}{this was great}

\textcolor{red}{this was dumb}

Samples from generative distribution

In our implementation:

XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]

\[
\begin{array}{ll}
0 & 0.88 \\
\textit{this was} & \textit{hilarious} & +1 \\
\textit{this was} & \textit{awesome} & +1 \\
\textit{this was} & \textit{boring} & -1 \\
\textit{this was} & \textit{great} & +1 \\
\textit{this was} & \textit{dumb} & -1 \\
\end{array}
\]

Samples from generative distribution

In our implementation:
- XLNet (Yang et al 2019)
- u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f, x) = "\text{In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]

“Very easy to flip the label by changing this word.”

0 0.88 1

\( \text{this was hilarious} \) +1

\( \text{this was awesome} \) +1

\( \text{this was boring} \) -1

\( \text{this was great} \) +1

\( \text{this was dumb} \) -1

Samples from generative distribution

In our implementation:
XLNet (Yang et al 2019)
u-PMLM (Liao et al 2020)
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can} \ x \ \text{be changed to flip the label, while respecting} \ \prod?\text{”} \]
Sensitivity in NLP

\[ s(f, x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]
Sensitivity in NLP

\[ s(f,x) = \text{"In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can x be changed to flip the label, while respecting } \prod\text{”} \]

![Diagram with sensitivity scores for the text “this was hilarious”]

1.8  
1.88  
1.6  
1  
1.8  
...

This diagram illustrates the sensitivity scores for the text “this was hilarious.” The scores indicate how many places within the text need to be altered to flip the label, while respecting the constraints specified by \( \prod \).
Sensitivity in NLP

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]
Sensitivity in NLP

\[ s(f,x) = \text{"In how many places can } x \text{ be changed to flip the label, while respecting } \prod?" \]

- 1.88
- 1.8
- 1.6
- 1
- 1.8

...
Low-sensitivity functions are more learnable

**Inputs:**
Length 10 bitstrings

**Model:**
Transformer (4 layers, 4 heads, 32 units)

Similar results with larger transformers, and with LSTMs
Low-sensitivity functions are more learnable

Inputs:
Length 10 bitstrings

Model:
Transformer
(4 layers, 4 heads, 32 units)

Low sensitivity function learned in 100 iterations

Similar results with larger transformers, and with LSTMs

Number of SGD Iterations
100 1000 10000
Low-sensitivity functions are more learnable

**Inputs:**
Length 10 bitstrings

**Model:**
Transformer (4 layers, 4 heads, 32 units)

Mid sensitivity functions take more iterations

Similar results with larger transformers, and with LSTMs
Low-sensitivity functions are more learnable

Inputs:
Length 10 bitstrings

Model:
Transformer
(4 layers, 4 heads, 32 units)

High sensitivity functions not learned at all even after 10,000 iterations

Similar results with larger transformers, and with LSTMs
Low-sensitivity functions are more learnable

Inputs:
Length 10 bitstrings

Model:
Transformer (4 layers, 4 heads, 32 units)

Similar results with larger transformers, and with LSTMs
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)
• GLUE challenge suite (Wang et al., 2019)
• $f$ estimated using finetuned RoBERTa
• GLUE challenge suite (Wang et al., 2019)
• $f$ estimated using finetuned RoBERTa

Stanford Sentiment Treebank (Socher et al. 2013)
• GLUE challenge suite (Wang et al., 2019)
• $f$ estimated using finetuned RoBERTa
• Stanford Sentiment Treebank (Socher et al 2013)
• Recognizing Textual Entailment (Dagan et al 2009)
• GLUE challenge suite (Wang et al., 2019)
• $f$ estimated using finetuned RoBERTa

- Stanford Sentiment Treebank (Socher et al 2013)
- Recognizing Textual Entailment (Dagan et al 2009)
- Winograd Schema Challenge (Levesque et al 2012)
a gorgeous, witty, seductive movie.
a gorgeous, witty, seductive movie.
a farce of ideas squanders this movie.

From Stanford Sentiment Treebank (SST-2)
a gorgeous, witty, seductive movie.
a farce of ideas squanders this movie.

Estimated sensitivity: $s(f,x) = 0.93$

From Stanford Sentiment Treebank (SST-2)
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.
- Chris Cook was attacked by Sculley and other Apple executives [...] and resigned from the company a few weeks later.
- 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.
- Jobs later worked for Apple
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) Estimated sensitivity: $s(f,x) = 4.8$
Error Reduction beyond majority class baseline (in %)

![Graph showing error reduction for BoE, BiLSTM, and RoBERTa across different average block sensitivities.](image)
Error Reduction beyond majority class baseline (in %)

- BoE: $R = -0.71, p = 0.001$
- BiLSTM: $R = -0.82, p = 0.0002$
- RoBERTa: $R = -0.05, p = 0.87$
Error Reduction beyond majority class baseline (in %)

- BoE: R = -0.71, p = 0.001
- BiLSTM: R = -0.82, p = 0.0002
- RoBERTa: R = -0.05, p = 0.87
Error Reduction beyond majority class baseline (in %)

- For BoE: $R = -0.71, p = 0.001$
- For BiLSTM: $R = -0.82, p = 0.0002$
- For RoBERTa: $R = -0.05, p = 0.87$
Error Reduction beyond majority class baseline (in %)

Provably empty due to theoretical bounds (see paper)

R = -0.71, p = 0.001
R = -0.82, p = 0.0002
R = -0.05, p = 0.87
Error Reduction beyond majority class baseline (in %)

BoE

R = −0.71, p = 0.001

BiLSTM

R = −0.82, p = 0.0002

RoBERTa

Error
Reduction
beyond
majority
class
baseline
(in %)

a
GLUE
a
Parsing
a
Syntax
a
Text Clas.
Error Reduction beyond majority class baseline (in %)

High-sensitivity tasks are hard to learn
Error Reduction beyond majority class baseline (in %)

BoE: $R = -0.71, p = 0.001$

BiLSTM: $R = -0.82, p = 0.0002$

RoBERTa: $R = -0.05, p = 0.87$
Sensitivity Identifies Difficult Inputs

Task: SST-2 (Binary Sentiment Classification)

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

Task: SST-2 (Binary Sentiment Classification)
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**

\[
\begin{align*}
\text{Simple Task} & \\
\text{Difficult Task} & \\
XOR(0,0,0,0,0) & \rightarrow 0 \\
XOR(1,0,0,0,0) & \rightarrow 1 \\
XOR(0,1,0,0,0) & \rightarrow 1 \\
XOR(0,0,1,0,0) & \rightarrow 1 \\
\end{align*}
\]

\[s(f,x) = N\]
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

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!
Sensitivity using Human Oracle Labels
Sensitivity using Human Oracle Labels
Why Subsets?
Why Subsets instead of Individual Words?

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

(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?

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

(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?

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

(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?

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

(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.

![Graph showing model fit predicting accuracy on SST-2](image)
Relation to Adversarial Examples
**Relation to Adversarial Examples**

<table>
<thead>
<tr>
<th>Adversarial Brittleness</th>
<th>High Sensitivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Szegedy et al., 2013; Jia and Liang, 2017)</td>
<td></td>
</tr>
</tbody>
</table>

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 |

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 |
Role of Task Model
Role of Task Model

![Graph showing the sensitivity of BERT and RoBERTa models for different tasks. The x-axis represents RoBERTa Sensitivity, and the y-axis represents BERT Sensitivity. The graph includes lines for QQP, RTE, and SST2 tasks.]
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 $\iff$ Fourier spectrum concentrated on low frequencies!
Other Complexity Metrics
The review sounds POSITIVE. Can you change the text so it sounds NEGATIVE?
Approximating Sensitivity without Models

The review sounds POSITIVE. Can you change the text so it sounds NEGATIVE?
Approximating Sensitivity without Models

The review sounds POSITIVE.
Can you change the text so it sounds NEGATIVE?
The review sounds POSITIVE. Can you change the text so it sounds NEGATIVE?
Approximating Sensitivity without Models

The review sounds POSITIVE. Can you change the text so it sounds NEGATIVE?
Approximating Sensitivity without Models

The review sounds POSITIVE.
Can you change the text so it sounds NEGATIVE?
Poisson regression: $\beta = 0.061$, $p = 0.0023$
controlling for task, sentence length, and random variation between sentences and annotators.
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)
Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

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 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

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

**Sensitivity and Difficulty of NLP Tasks**
Sensitivity as a Complexity Measure for Sequence Classification Tasks

Sensitivity for Sequence Classification

\[ bs(f, x) := \max_{k,P_1 \cup \cdots \cup P_k} \sum_{i=1}^{k} s(f, x, P_i) \]

Sensitivity and Difficulty of NLP Tasks
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
Sensitivity of Boolean Functions

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

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

\[ s(f, x) = \# \{\text{Hamming neighbor of} \ x \ \text{with opposite label}\} \]

\[ f(x, y) = x \ XOR \ y \]

\[ s(f, x) = 2 \]

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

\[ s(f,x) = \# \{\text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \oplus y \]

\[ s(f,x) = 2 \]

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

\[ s(f,x) = \# \{\text{Hamming neighbor of } x \text{ with opposite label}\} \]

\[ f(x,y) = x \text{ XOR } y \]

\[ s(f,x) = 2 \]

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

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

\[ s(f, x) = 2 \]
Sensitivity of Boolean Functions

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

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

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

\[ s(f,x) = 2 \]

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

\[ s(f,x) = 2 \]

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

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

\[ s(f, x) = 2 \]

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

\[ s(f,x) = 2 \]
Sensitivity of Boolean Functions

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

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

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

Partitions of \( \{1, \ldots, |x|\} \) into disjoint subsets

Variance of \( f \) when resampling words in \( P_i \)

"In how many places can \( x \) be changed to flip the label, while respecting input distribution?"
Sensitivity of Boolean Functions

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \quad 0 \text{ XOR } 0 \rightarrow 0 \]

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

\[ s(f, x) = \# \{\text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

- \( 0 \text{ XOR } 0 \rightarrow 0 \)
- \( 1 \text{ XOR } 0 \rightarrow 1 \)
- \( 0 \text{ XOR } 1 \rightarrow 1 \)

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

0 XOR 0 → 0

1 XOR 0 → 1 \quad \text{s}(f,x) = 2

0 XOR 1 → 1

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

\[ s(f,x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x,y) = x \text{ XOR } y \]

0 XOR 0 → 0
1 XOR 0 → 1
0 XOR 1 → 1

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

\[ s(f, x) = \# \{ \text{Hamming neighbor of } x \text{ with opposite label} \} \]

\[ f(x, y) = x \text{ XOR } y \]

0 XOR 0 → 0

1 XOR 0 → 1 \hspace{1cm} s(f, x) = 2

0 XOR 1 → 1

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

\[ s(f,x) = \text{“In how many places can } x \text{ be changed to flip the label, while respecting } \prod?\text{”} \]

\[
\begin{array}{ccc}
0 & 0.88 & 1 \\
\end{array}
\]

Sensitivity 1.88

this was hilarious

Various theoretical motivations for this definition (see paper)
Estimating Sensitivity on NLP Tasks

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

“Plug-in estimation”:

Estimate sensitivity of a classifier known to do the task accurately

Can verify accuracy with human labels
a painfully funny ode to bad behavior.
a painfully funny ode to bad behavior.

1. Not a funny story, just bad behavior. $s(f,x,P) = 0.96$
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.  $s(f,x,P) = 0.74$
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