On a Benefit of Masked Language Modeling: Robustness to Simplicity Bias

Ting-Rui Chiang
tingruic@usc.edu

What we know about pretrained models

- Require less data when fine-tuning
- Smoother loss surface [1]
- Lower intrinsic dimension [2]
- More robust to spurious (unreliable) features [3,4]

[1] Yaru Hao, Li Dong, Furu Wei, and Ke Xu. Visualizing and understanding the effectiveness of BERT
[2] Armen Aghajanyan, Luke Zettlemoyer, and Sona Gupta. Intrinsic dimensionality explains the effectiveness of language model fine-tuning.
[3] Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models.
[4] Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. Pretrained transformers improve out-of-distribution robustness.

Why is a model unrobust?

Conjecture: May be due to the **pitfall of simplicity bias** [1].

→ **Simplicity bias:** deep models tend to rely on simple features instead of utilizing all the features [2].

→ **Pitfall:** may not be robust.

[1] Harshay Shah, Kaustav Tamuly, Aditi Raghunathan, Prateek Jain, and Praneeth Netrapalli. The pitfalls of simplicity bias in neural networks.

[2] Dimitris Kalimeris, Gal Kaplun, Preetum Nakkiran, Benjamin Edelman, Tristan Yang, Boaz Barak, and Haofeng Zhang. 2019. Sgd on neural networks learns functions of increasing complexity.

Simplicity bias

Data Point X

Simple but spurious features

Complex but robust features

For example, in the toxic text detection task [1,2]:

The presence (or not) of some group identifiers, e.g. women, black, etc.
Single dimension

The *semantic* encoded by the tokens in the sentence.
Much higher dimension

Problem: Those spurious features are so tempting!

[1] Lucas Dix ... ting unintended b...
[2] Xuhui Zho... ated debiasing for toxic language detection.

How can the problem be alleviated?

Data Point $\boldsymbol{X}$

Spurious features: token $\boldsymbol{X_1}$

Robust features: context $\boldsymbol{X_2}$

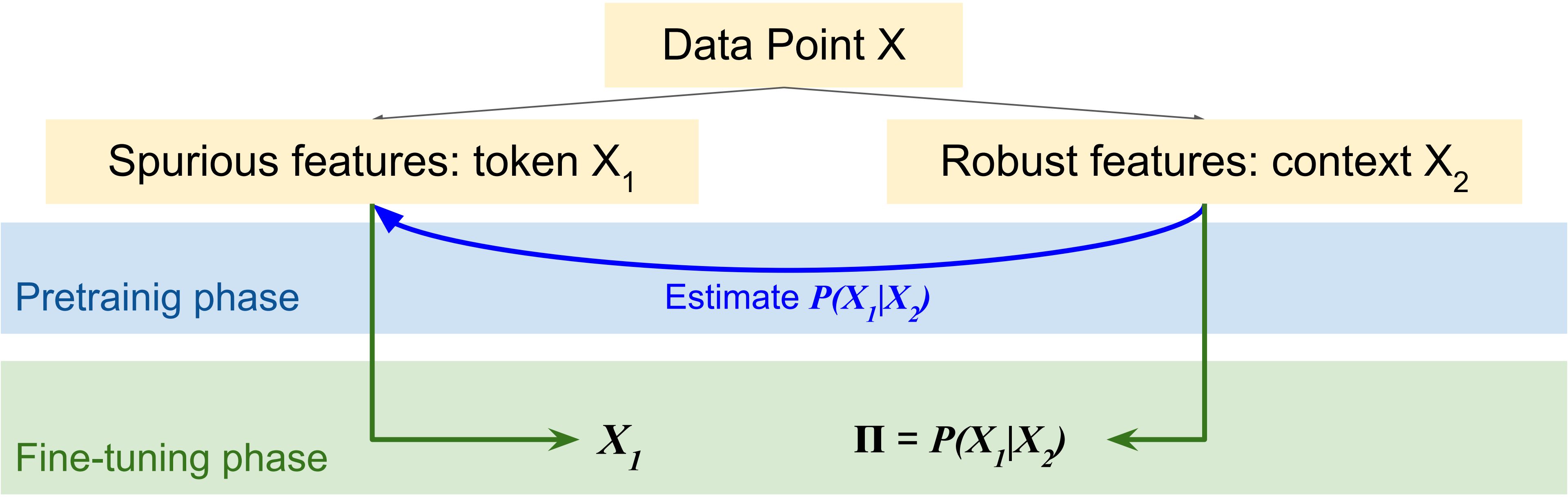
$\mathbf{\Pi}$

What if we extract a feature $\mathbf{\Pi} = \mathbf{f}(\boldsymbol{X_2})$ such that

1. $\mathbf{\Pi}$ is as useful as $\boldsymbol{X_1}$
2. Learning from $\mathbf{\Pi}$ is as easy as $\boldsymbol{X_1}$

Effect: Due to the simplicity bias, the model relies more on $\mathbf{\Pi}$, and so relies more on $\boldsymbol{X_2}$.

Theory in this work: MLM extracts Π



Theorem 1: Π is as informative as X_1 (at least)

Theorem 2: Π is as easy as X_1 (at least)

Effect: The model relies more on Π, and so relies more on X_2.

Experimental Settings

- To verify that modeling $P(X_1|X_2)$ makes models more robust.
- Pretrain two models with two masking policies:
  - Unmask spurious: Remove masks over the spurious features.
  - Unmask random: Remove some masks at random.
- Fine-tune the two models.
- Compare the performance on *out-of-distribution* data.
  - (the spurious features are not useful)
- Two tasks
  - NER: don’t just memorize the name entities.
  - Hate speech detection: don’t rely on the group identifiers.

Results

Mask Policy	NER Origin F1 ↑	NER Unseen F1 ↑	Hate Speech Detection All (12893) Accuracy ↑	Hate Speech Detection All (12893) F1 ↑	Hate Speech Detection NOI (602) Accuracy ↑	Hate Speech Detection NOI (602) FPR ↓
scratch	61.5 0.5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
vanilla	74.2 0.4	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
unmask random	72.7 0.6	56.5 0.8	83.3 1.1	78.9 1.1	75.8 0.9	25.7 2.3
unmask spurious	72.9 0.5	53.2 0.8	84.1 0.7	79.8 0.6	73.7 1.0	32.5 2.1
remove spurious	[illegible]	[illegible]	[illegible]	[illegible]	77.3 0.6	21.7 2.0

Modeling the spurious token performs better on OOD.

Similar performance on ID.

Modeling the spurious token indeed improves the robustness.

Conclusion

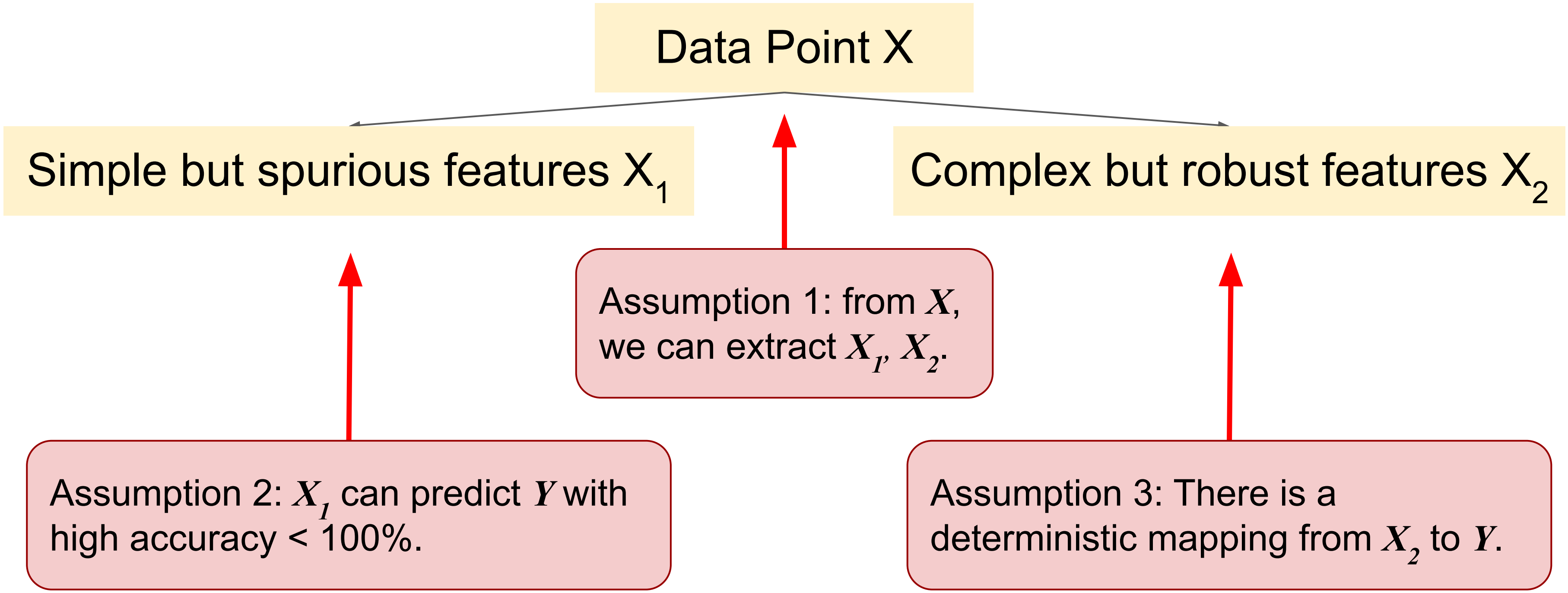
- Propose the hypothesis why MLM is useful
    - Theoretically: prove that MLM can extract simple features from the robust feature.
    - Empirically: show that modeling the spurious features make models more robust.

Q&A

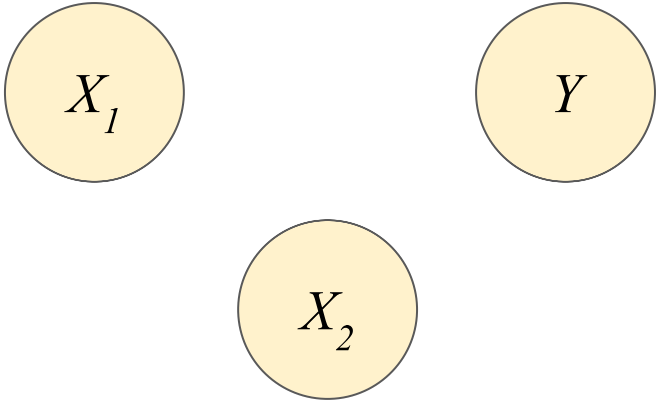
Ting-Rui Chiang
https://ctinray.github.io/

Alert: Math Ahead!

My theory: MLM makes models more robust to lexical bias



Graphical Model

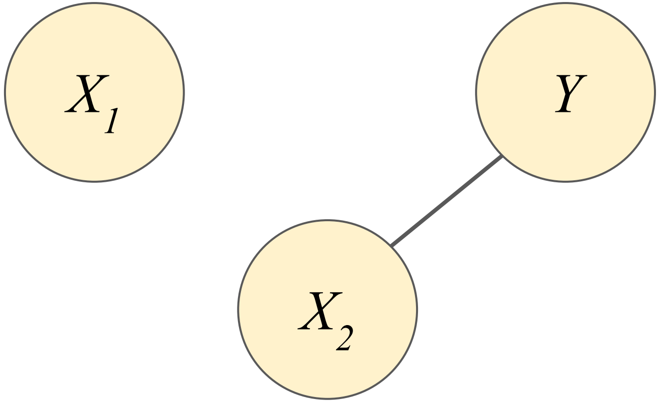


Assumption 1: from $\boldsymbol{X}$, we can extract $\boldsymbol{X_1}$, $\boldsymbol{X_2}$.

Assumption 2: $\boldsymbol{X_1}$ can predict $\boldsymbol{Y}$ with high accuracy < 100%.

Assumption 3: There is a deterministic mapping from $\boldsymbol{X_2}$ to $\boldsymbol{Y}$.

Graphical Model

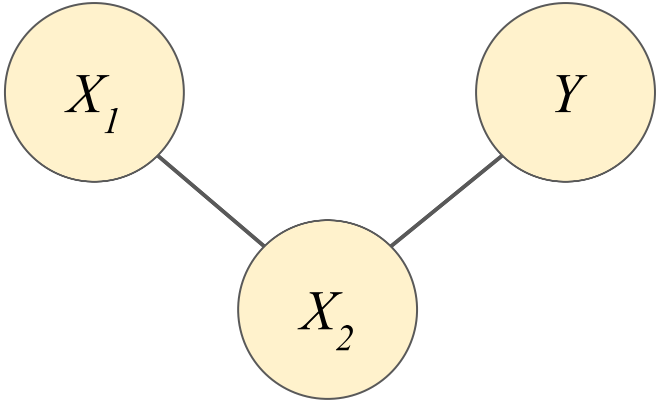


Assumption 1: from X, we can extract X_1, X_2.

Assumption 2: X_1 can predict Y with high accuracy < 100%.

Assumption 3: There is a deterministic mapping from X_2 to Y.

Graphical Model

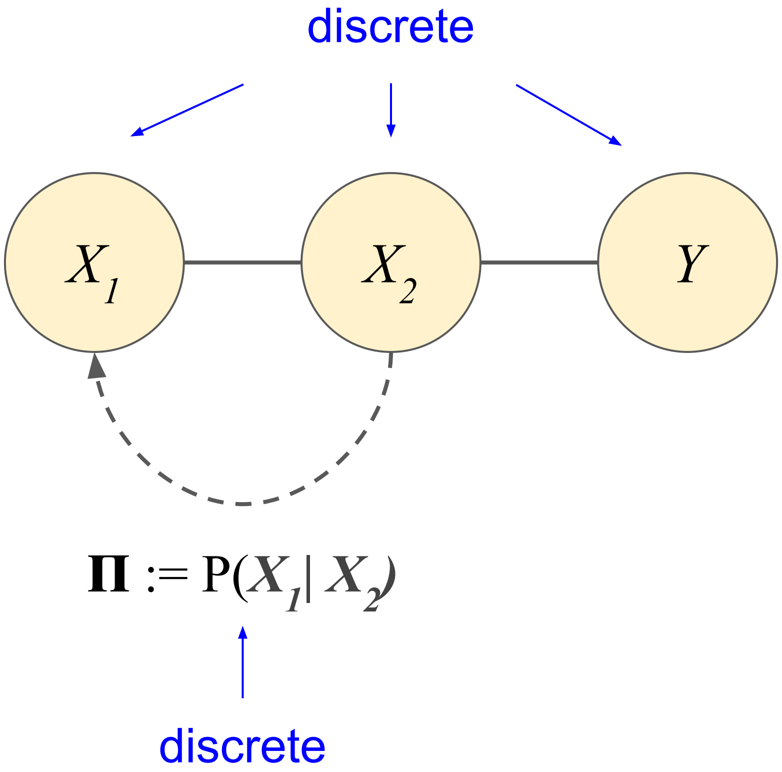


Assumption 1: from X, we can extract X_1, X_2.

Assumption 2: X_1 can predict Y with high accuracy < 100%.

Assumption 3: There is a deterministic mapping from X_2 to Y.

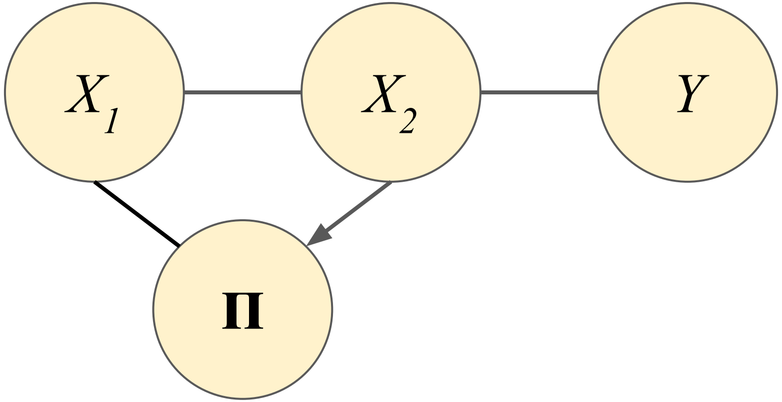
Lemma



$$\mathbf{I}(X_1; X_2) = \mathbf{I}(\mathbf{\Pi}; X_1)$$

$$\boxed{\mathrm{P}(\ \ | X_2) \in \arg\max_f \mathrm{I}(\ f(X_1); X_1)}$$

Theorem 1



$\mathbf{\Pi} := \mathrm{P}(X_1 | X_2)$

Lemma 1:

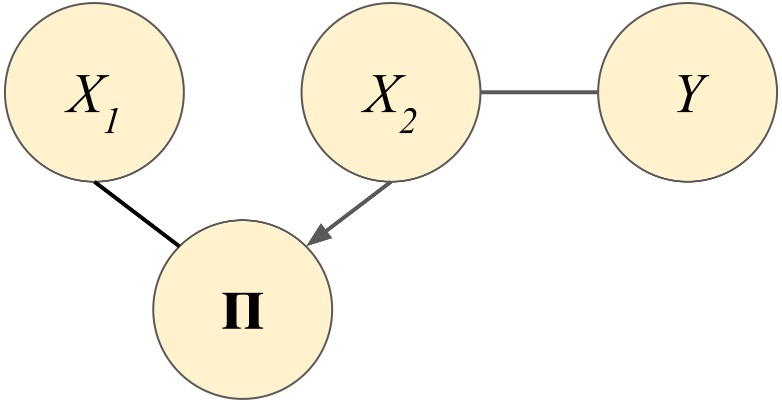
$$\mathbf{I}(X_1; X_2) = \mathbf{I}(\mathbf{\Pi}; X_1)$$

Theorem 1:

$$\mathbf{I}(\mathbf{\Pi}; Y) \geqq \mathbf{I}(X_1; Y)$$

$\mathbf{\Pi}$ is informative

Theorem 2



$\mathbf{\Pi} := \mathrm{P}(X_1 | X_2)$

Theorem 2:

Learning from $\mathbf{\Pi}$

- Converges as fast as from X_1
- Converges to a solution as good as the optimal solution with X_1
- The model is linear

Theorem 2: Formal Results

- Both $\tilde{h}_{X_1}^{(n)}$ and $\tilde{h}_{\Pi}^{(n)}$ converge in $\mathrm{O}\left(\frac{1}{n}\right)$
- When $n \to \infty$, the loss of $\tilde{h}_{\Pi}^{(n)}$ is less than $\tilde{h}_{X_1}^{(n)}$.

Learning from $\mathbf{\Pi}$ is easy

Theorem 2: Outline of the Proof

- Given $(x_1^{(1)}, y^{(1)}), (x_1^{(2)}, y^{(2)}), \cdots, (x_1^{(n)}, y^{(n)})$

$$\tilde{h}_{X_1}^{(n)}(Y = 1 | X_1 = 1) = \frac{\sum_i^n \mathbb{1}[x_1^{(i)} = 1] \mathbb{1}[y^{(i)} = 1]}{\sum_i^n \mathbb{1}[x_1^{(i)} = 1]}$$

contains (sort of) underlying dist. of x_1

- Given $(\pi^{(1)}, y^{(1)}), (\pi^{(2)}, y^{(2)}), \cdots, (\boxed{\pi^{(n)}}, y^{(n)})$

$$\tilde{h}_{\Pi}^{(n)}(Y = 1 | X_1 = 1) = \frac{\sum_i^n \pi^{(i)}(X_1 = 1) \mathbb{1}[y^{(i)} = 1]}{\sum_i^n \pi^{(i)}(X_1 = 1)}$$

$$\tilde{h}_{\Pi}^{(n)}(Y = 1 | \Pi) = \tilde{h}_{\Pi}^{(n)}(Y = 1 | X_1 = 0)\Pi(X_1 = 0) + \tilde{h}_{\Pi}^{(n)}(Y = 1 | X_1 = 1)\Pi(X_1 = 1)$$

Alert Lifted