UnNatural Language Inference

Koustuv Sinha, Prasanna Parthasarathi, Joelle Pineau, Adina Williams

ACL-IJCNLP 2021

AFNLP

McGill School of Computer Science

Mila

FACEBOOK AI

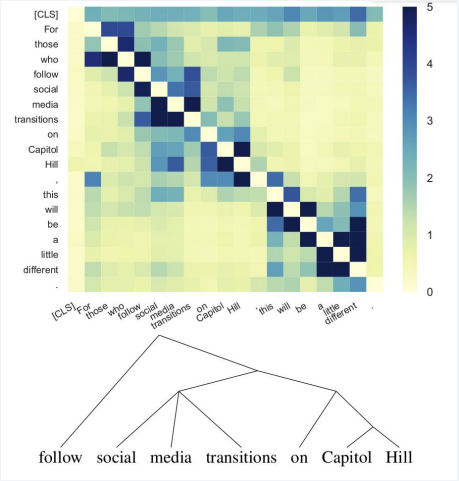
“Pretrained LMs know syntax”

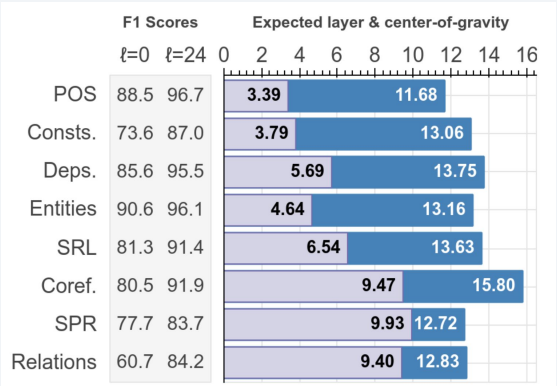
- Wu et al. (2020) recover syntactic trees from BERT considering attention patterns

- Tenney et al. (2019) conclude that BERT ‘recreates the classical NLP pipeline:’ POS tagging, parsing, NER, semantic roles, coreference…

- Many papers claim LMs “know syntax” on the basis of probes and diagnostic datasets

  (Goldberg, 2019; Hewitt and Manning, 2019; Jawahar et al., 2019; Wu et al., 2020; Warstadt et al 2019a,b; Warstadt and Bowman 2020; Linzen and Baroni 2021…)





Test of syntax: the order of words conveys important information.

The person bit the cat.

The cat bit the person.

mean very different things!

If models are genuinely learning syntax, they should know something about word order...

If models are genuinely learning syntax, they should know something about word order… **do they?**

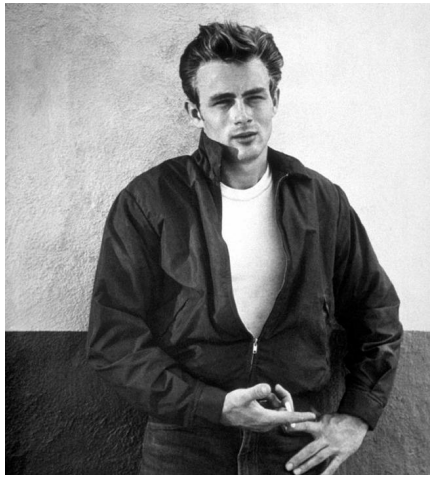
Natural Language Inference (NLI)

also known as recognizing textual entailment (RTE[1])

James Byron Dean refused to move without blue jeans

{entails, contradicts, neither}

James Dean didn't dance without pants



[1]Fyodorov et al., 2000; Condoravdi et al., 2003; Bos and Markert, 2005; Dagan et al., 2006; MacCartney and Manning, 2009

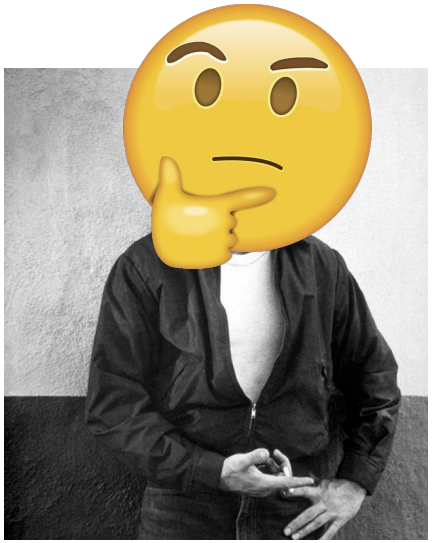
Example: MacCartney thesis '09

Wait a sec…how *should* a (humanlike) NLI model that's sensitive to word order behave?

refused James jeans blue without Dean Byron move to

{entails?, contradicts?, neither?}

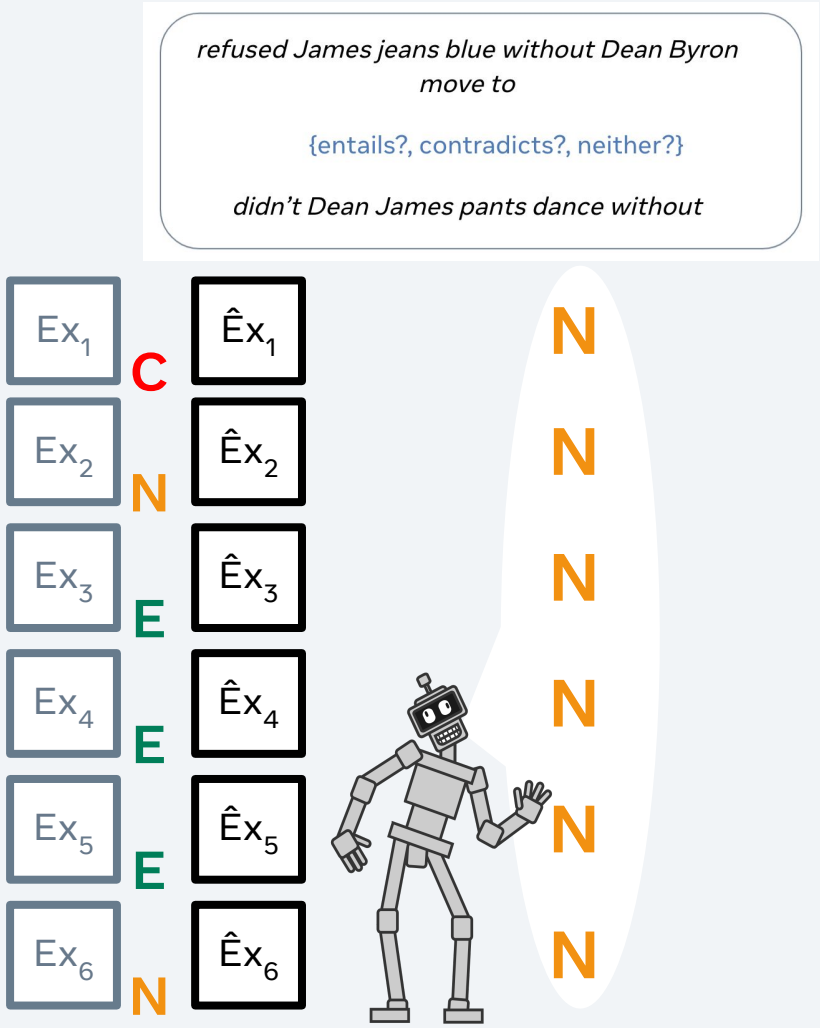
didn't Dean James pants dance without



(1) Maybe it just performs NLI...

For 3-way NLI, any pair that isn't clearly contradiction or entailment should be **neutral**.

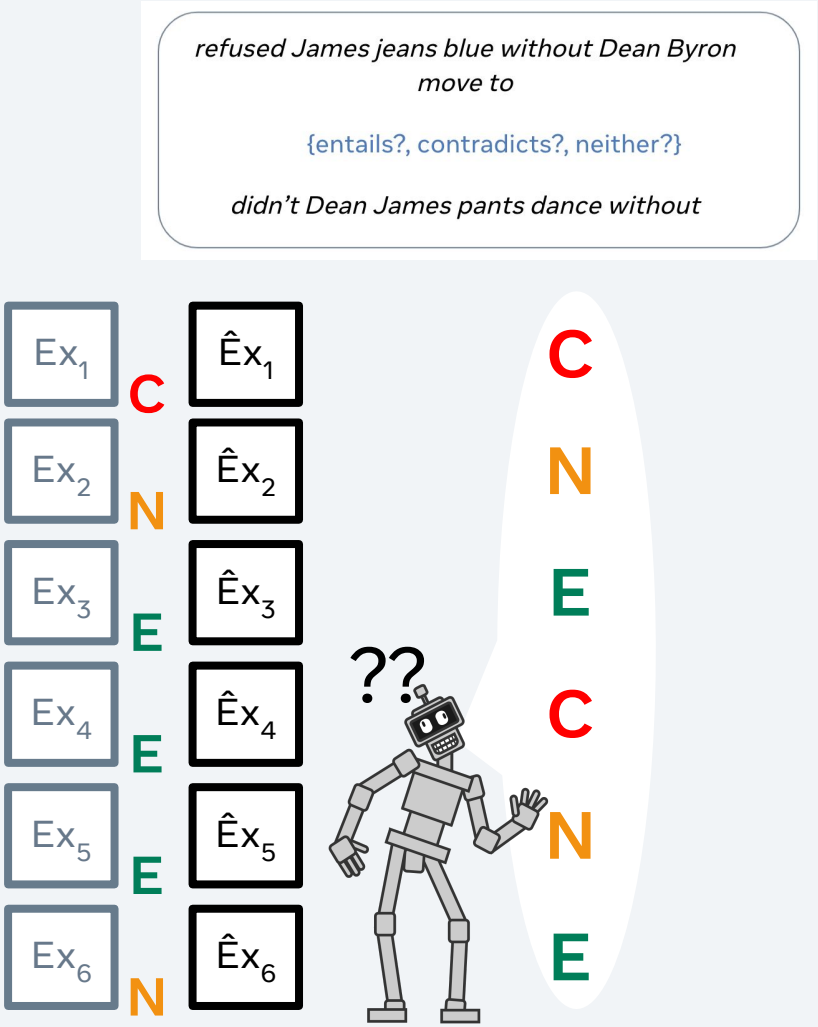
A model that learned this might just assign **neutral** always.



(2) Maybe it will just be very uncertain…

Perhaps it will just have no idea…then it should get roughly equal probability mass on all predictions.

This is approximately the **most frequent class** baseline.



Spoiler! **It's neither!**

State-of-the-art NLI **models are largely invariant to word order!**

Models often *accept* permuted examples (i.e. assign the original gold label to them).

Same for pre-Transformer era neural models, too!

P: Boats in daily use lie within feet of the fashionable bars and restaurants .
H: There are boats close to bars and restaurants .

Gold Label	Premise	Hypothesis
E	Boats in daily use lie within feet of the fashionable bars and restaurants.	There are boats close to bars and restaurants.
E	restaurants and use feet of fashionable lie the in Boats within bars daily .	bars restaurants are There and to close boats .
C	He and his associates weren't operating at the level of metaphor.	He and his associates were operating at the level of the metaphor.
C	his at and metaphor the of were He operating associates n't level .	his the and metaphor level the were He at associates operating of .

Concurrently, similar findings on GLUE and QA has been shown by Pham et al 2021, Gupta et al 2021

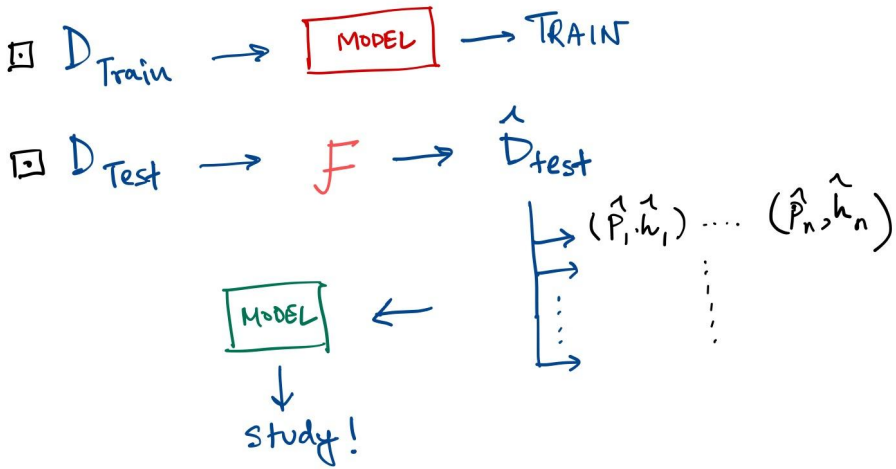
Constructing permutation function

No word should appear in its original position

A sentence of length *n* has *(n-1)!* possible permutations

We select only *unique* permutations from this operation

P: Boats in daily use lie within feet of the fashionable bars and restaurants .
H: There are boats close to bars and restaurants .



We are interested in **accuracy** on **permuted sentences**

Ex_1 = The cat sat on the mat → The cat was fat

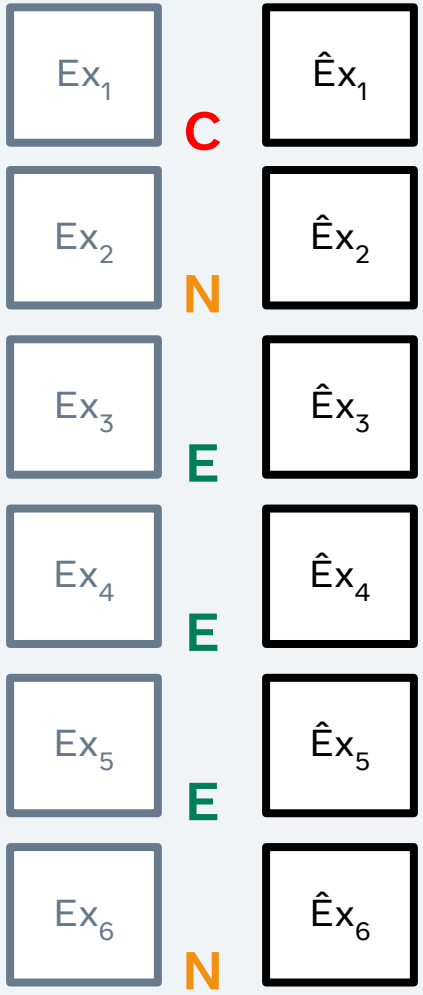
...

Ex_6 = All cats are mammals → Felix is a mammal

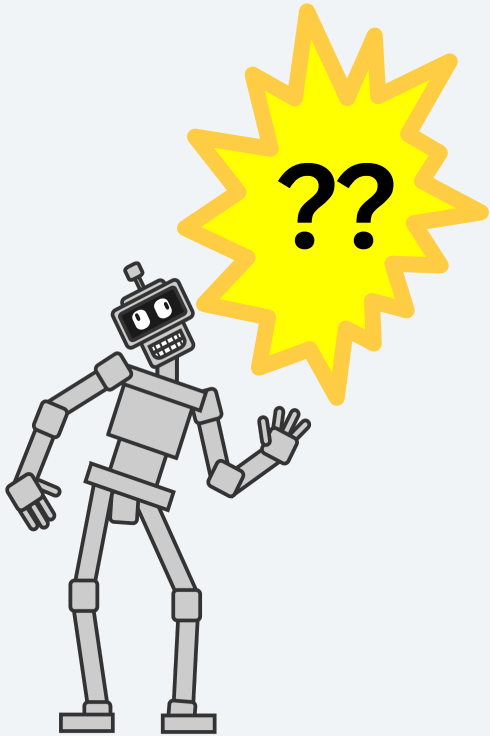
$\hat{E}x_1$ = on sat The the mat cat → cat was The fat

...

$\hat{E}x_6$ = are mammals cats All → Felix mammal a is



100 permutations per example



Experimental Setup:

Trained models (RoBERTa, BART, DistilBERT, InferSent, ConvNet, BiLSTM) on MNLI to SOTA levels.

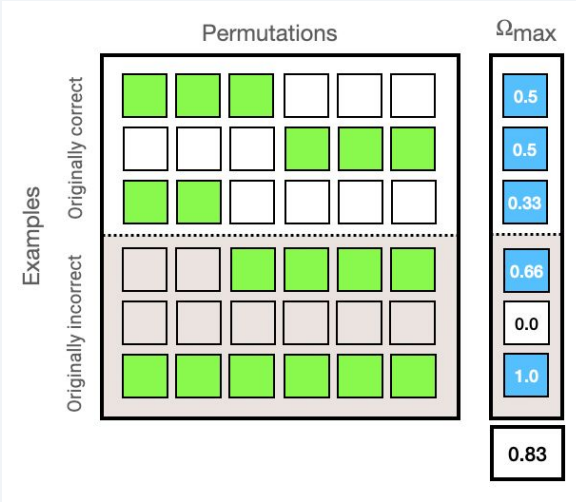
Fine-tuned on (normal) MNLI.

Evaluated on permuted MNLI, SNLI (in domain), ANLI (out of domain).

How many examples have at least one permutation predicting the gold label?

Model	Eval Dataset	$\mathcal{A}$	$\Omega_{\max}$	$\mathcal{P}^c$	$\mathcal{P}^f$	Ω_{rand}
RoBERTa (large)	MNLI_m_dev	0.906	0.987			
	MNLI_mm_dev	0.901	0.987			
	SNLI_dev	0.879	0.988			
	SNLI_test	0.883	0.988			
	A1_dev	0.456	0.897			
	A2_dev	0.271	0.889			
	A3_dev	0.268	0.902			
	Mean	0.652	0.948			
	Harmonic Mean	0.497	0.946			
BART (large)	MNLI_m_dev	0.902	0.989			
	MNLI_mm_dev	0.900	0.986			
	SNLI_dev	0.886	0.991			
	SNLI_test	0.888	0.990			
	A1_dev	0.455	0.894			
	A2_dev	0.316	0.887			
	A3_dev	0.327	0.931			
	Mean	**0.668**	**0.953**			
	Harmonic Mean	**0.543**	**0.951**			
DistilBERT	MNLI_m_dev	0.800	0.968			
	MNLI_mm_dev	0.811	0.968			
	SNLI_dev	0.732	0.956			
	SNLI_test	0.738	0.950			
	A1_dev	0.251	0.750			
	A2_dev	0.300	0.760			
	A3_dev	0.312	0.830			
	Mean	0.564	0.883			
	Harmonic Mean	0.445	0.873			

98.9%



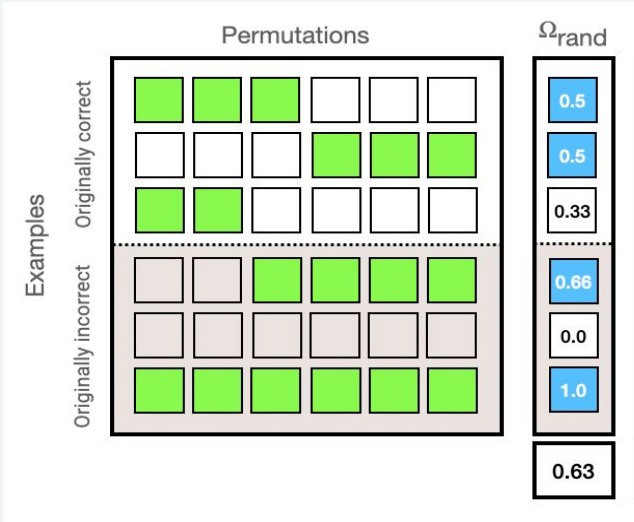
E_1: 3 gold label assignments (50%)
E_2: 3 gold label assignments (50%)
E_3: 2 gold label assignments (33%)
E_4: 4 gold label assignments (66%)
E_5: 0 gold label assignments ~~(00%)~~
E_6: 6 gold label assignments (100%)

Ω_{max} = ⅚ examples = 83%

How many examples have at least 1/3rd permutations predicting the gold label?

Model	Eval Dataset	$\mathcal{A}$	Ω_{max}	$\mathcal{P}^c$	$\mathcal{P}^f$	Ω_{rand}
RoBERTa (large)	MNLI_m_dev	0.906	0.987			0.794
	MNLI_mm_dev	0.901	0.987			0.790
	SNLI_dev	0.879	0.988			0.826
	SNLI_test	0.883	0.988			0.828
	A1_dev	0.456	0.897			0.364
	A2_dev	0.271	0.889			0.359
	A3_dev	0.268	0.902			0.397
	Mean	0.652	0.948			0.623
	Harmonic Mean	0.497	0.946			0.539
BART (large)	MNLI_m_dev	0.902	0.989			0.784
	MNLI_mm_dev	0.900	0.986			0.788
	SNLI_dev	0.886	0.991			0.834
	SNLI_test	0.888	0.990			0.836
	A1_dev	0.455	0.894			0.374
	A2_dev	0.316	0.887			0.397
	A3_dev	0.327	0.931			0.424
	Mean	0.668	0.953			**0.634**
	Harmonic Mean	0.543	0.951			**0.561**
DistilBERT	MNLI_m_dev	0.800	0.968			0.779
	MNLI_mm_dev	0.811	0.968			0.786
	SNLI_dev	0.732	0.956			0.731
	SNLI_test	0.738	0.950			0.725
	A1_dev	0.251	0.750			0.300
	A2_dev	0.300	0.760			0.343
	A3_dev	0.312	0.830			0.363
	Mean	0.564	0.883			0.575
	Harmonic Mean	0.445	0.873			0.490

83.6%



E_1: 3 gold label assignments (50%)
E_2: 3 gold label assignments (50%)
E_3: 2 gold label assignments ~~(33%)~~
E_4: 4 gold label assignments (66%)
E_5: 0 gold label assignments ~~(00%)~~
E_6: 6 gold label assignments (100%)

Ω_{rand} = ⅔ examples = 63%

How many examples have ALL permutations predicting the gold label?

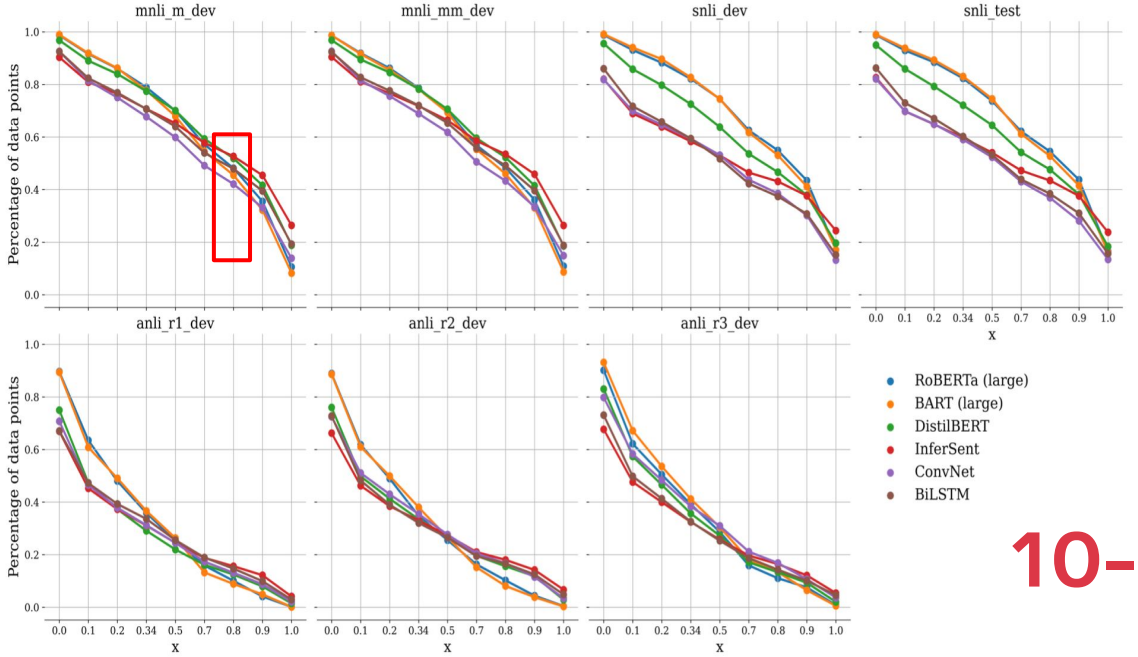
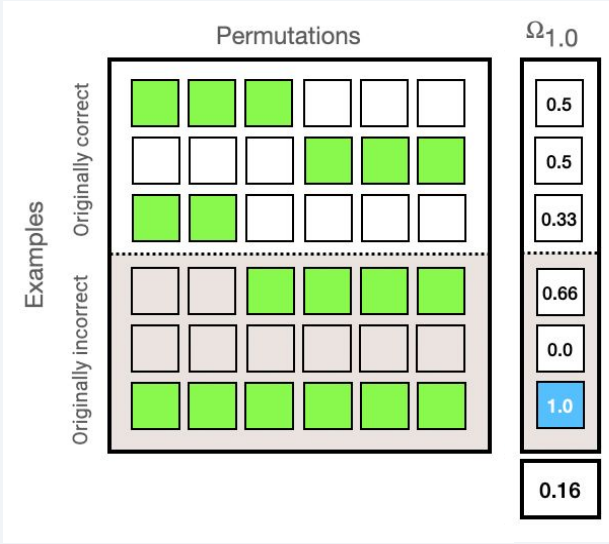


Figure 7: Ω_x threshold for all datasets with varying x and computing the percentage of examples that fall within the threshold. The top row consists of in-distribution datasets (MNLI, SNLI) and the bottom row contains out-of-distribution datasets (ANLI)



10-20%

E_1: 3 gold label assignments (50%)
E_2: 3 gold label assignments (50%)
E_3: 2 gold label assignments ~~(33%)~~
E_4: 4 gold label assignments (66%)
E_5: 0 gold label assignments ~~(00%)~~
E_6: 6 gold label assignments (100%)

$\Omega_{1.0}$ = ⅙ examples = 16%

We observed that **for some examples the models initially got wrong**, there exists (a) permutation(s) that receive(s) the **gold label**!

FLIPS: *What percentage of permutations predict gold label, whose original pairs were INCORRECTLY predicted?*

Model	Eval Dataset	$\mathcal{A}$	Ω_{max}	$\mathcal{P}^c$	$\mathcal{P}^f$	Ω_{rand}
RoBERTa (large)	MNLI_m_dev	0.906	0.987	0.707	0.383	0.794
	MNLI_mm_dev	0.901	0.987	0.707	0.387	0.790
	SNLI_dev	0.879	0.988	0.768	0.393	0.826
	SNLI_test	0.883	0.988	0.760	0.407	0.828
	A1_dev	0.456	0.897	0.392	0.286	0.364
	A2_dev	0.271	0.889	0.465	0.292	0.359
	A3_dev	0.268	0.902	0.480	0.308	0.397
	Mean	0.652	0.948	0.611	0.351	0.623
	Harmonic Mean	0.497	0.946	0.572	0.344	0.539
BART (large)	MNLI_m_dev	0.902	0.989	0.689	0.393	0.784
	MNLI_mm_dev	0.900	0.986	0.695	0.399	0.788
	SNLI_dev	0.886	0.991	0.762	0.363	0.834
	SNLI_test	0.888	0.990	0.762	0.370	0.836
	A1_dev	0.455	0.894	0.379	0.295	0.374
	A2_dev	0.316	0.887	0.428	0.303	0.397
	A3_dev	0.327	0.931	0.428	0.333	0.424
	Mean	0.668	0.953	0.592	**0.351**	0.634
	Harmonic Mean	0.543	0.951	0.546	**0.347**	0.561
DistilBERT	MNLI_m_dev	0.800	0.968	0.775	0.343	0.779
	MNLI_mm_dev	0.811	0.968	0.775	0.346	0.786
	SNLI_dev	0.732	0.956	0.767	0.307	0.731
	SNLI_test	0.738	0.950	0.770	0.312	0.725
	A1_dev	0.251	0.750	0.511	0.267	0.300
	A2_dev	0.300	0.760	0.619	0.265	0.343
	A3_dev	0.312	0.830	0.559	0.259	0.363
	Mean	0.564	0.883	**0.682**	0.300	0.575
	Harmonic Mean	0.445	0.873	**0.664**	0.296	0.490

35-40%

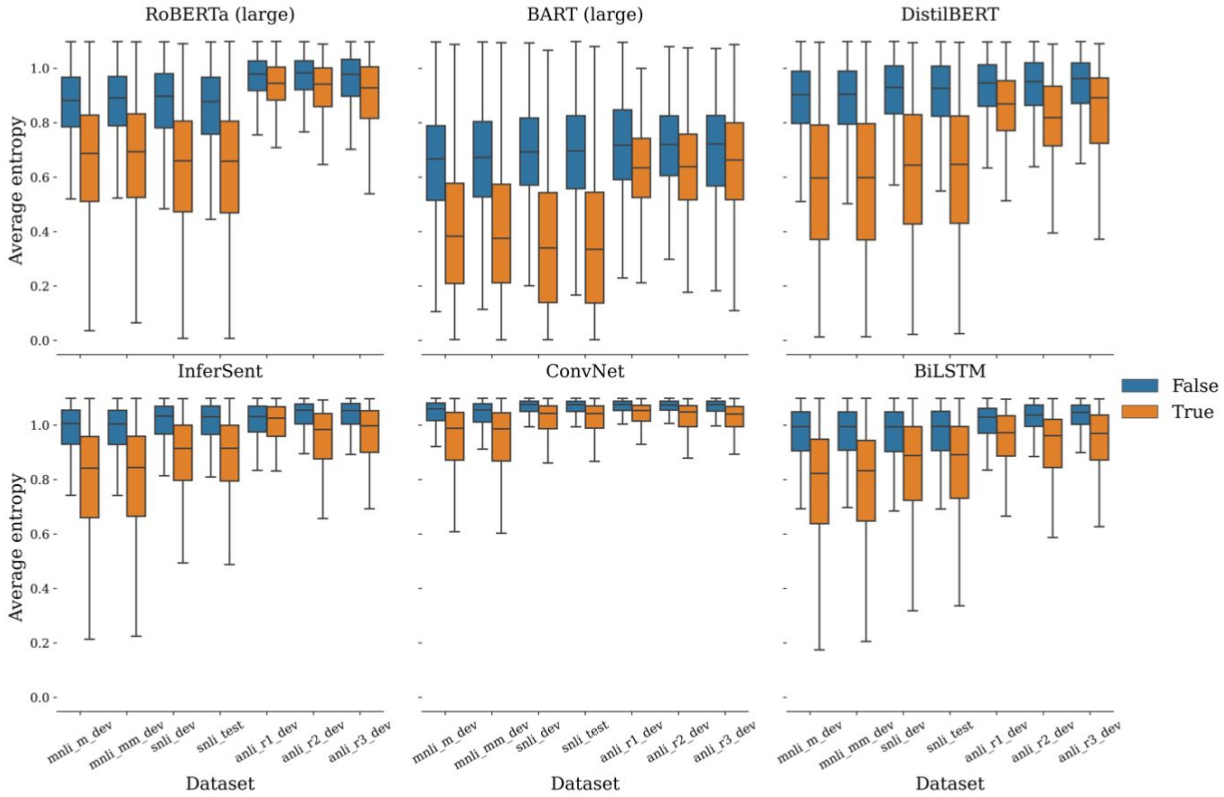
Note: for a classic Bag-of-Words, P^c would be 100% and P^f would be 0%!

Is it just for Transformers? No!

- *Weaker models, weaker effect.*
- P^f for non-Transformers is approximately the same as for transformers.
- Both architectures are similarly bag-of-words-y (though no investigated model is a strict BOW).

Model	Eval Dataset	$\mathcal{A}$	Ω_{max}	$\mathcal{P}^c$	$\mathcal{P}^f$	Ω_{rand}
InferSent	MNLI_m_dev	0.658	0.904	0.842	0.359	0.712
	MNLI_mm_dev	0.669	0.905	0.844	0.368	0.723
	SNLI_dev	0.556	0.820	0.821	0.323	0.587
	SNLI_test	0.560	0.826	0.824	0.321	0.600
	A1_dev	0.316	0.669	0.425	0.395	0.313
	A2_dev	0.310	0.662	0.689	0.249	0.330
	A3_dev	0.300	0.677	0.675	0.236	0.332
	Mean	**0.481**	0.780	0.731	**0.322**	0.514
	Harmonic Mean	0.429	0.767	0.694	**0.311**	0.455
ConvNet	MNLI_m_dev	0.631	0.926	0.773	0.340	0.684
	MNLI_mm_dev	0.640	0.926	0.782	0.343	0.694
	SNLI_dev	0.506	0.819	0.813	0.339	0.597
	SNLI_test	0.501	0.821	0.809	0.341	0.596
	A1_dev	0.271	0.708	0.648	0.218	0.316
	A2_dev	0.307	0.725	0.703	0.224	0.356
	A3_dev	0.306	0.798	0.688	0.234	0.388
	Mean	0.452	**0.817**	**0.745**	0.291	0.519
	Harmonic Mean	0.404	**0.810**	**0.740**	0.279	**0.473**
BiLSTM	MNLI_m_dev	0.662	0.925	0.800	0.351	0.711
	MNLI_mm_dev	0.681	0.924	0.809	0.344	0.724
	SNLI_dev	0.547	0.860	0.762	0.351	0.598
	SNLI_test	0.552	0.862	0.771	0.363	0.607
	A1_dev	0.262	0.671	0.648	0.271	0.340
	A2_dev	0.297	0.728	0.672	0.209	0.328
	A3_dev	0.304	0.731	0.656	0.219	0.331
	Mean	0.472	0.814	0.731	0.301	**0.520**
	Harmonic Mean	0.410	0.803	0.725	0.287	0.463

Wait a minute! The labels must be chosen by chance!



- Unfortunately, no. The average entropy for Transformers is pretty low, suggesting overconfidence*!
- BART has the lowest entropy/highest confidence!
- Pre-Transformer models are somewhat better, but probably due to their lower capacity.

Recall: highest entropy for 3-labels is ~1.58

*although miscalibration might also come into play.

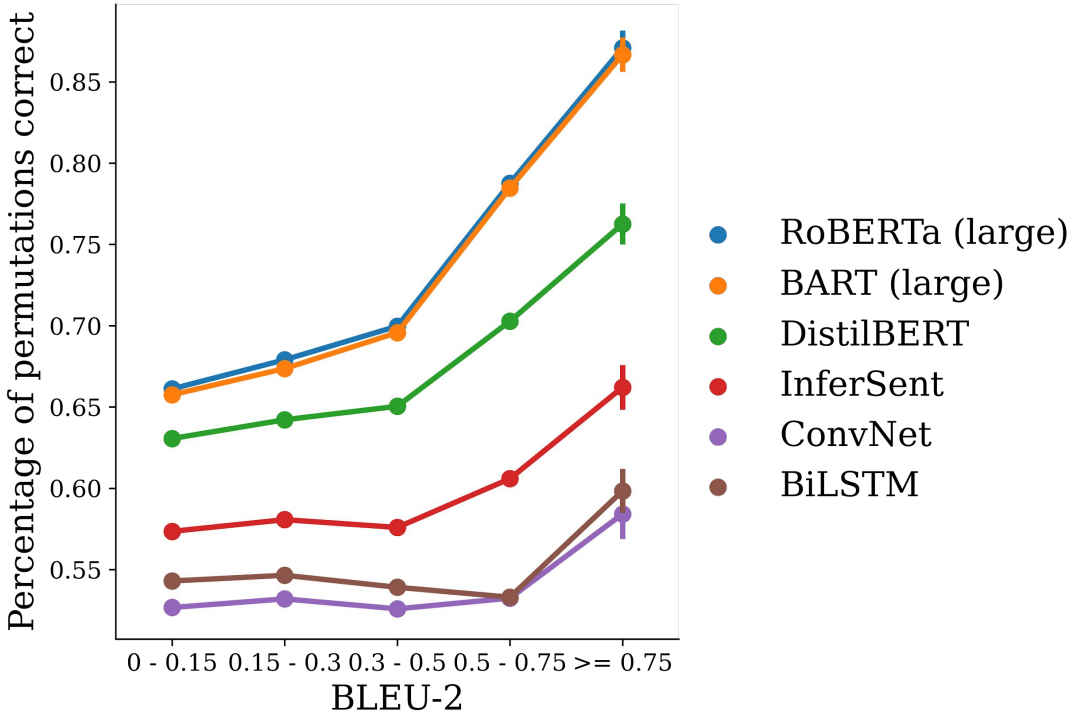
Which permutations do our models accept?

FACEBOOK AI

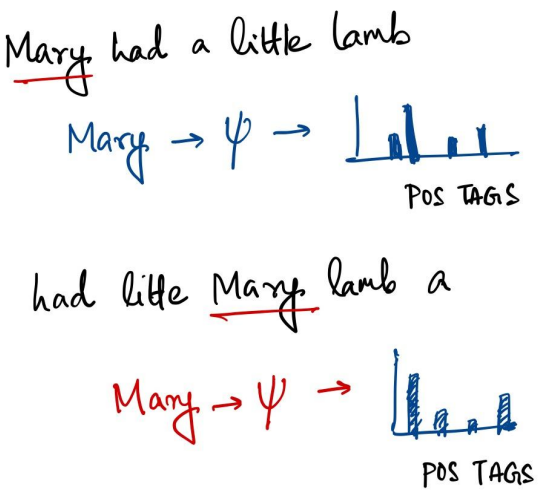
Preserving local word order leads to accepted permutations

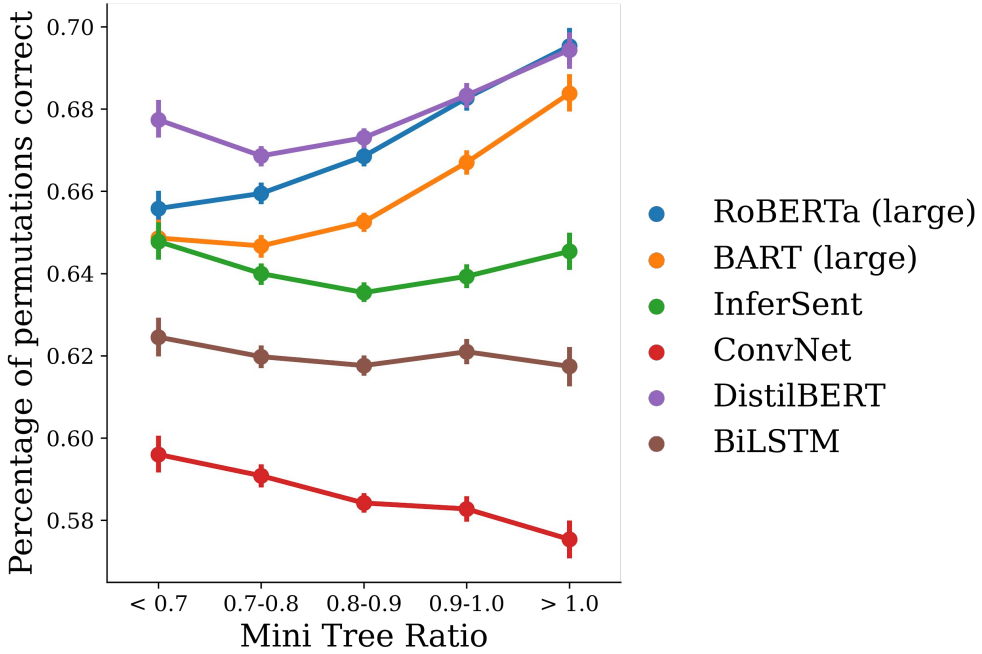
Percentage of permutations correct increases with more bi-gram overlap!

(BLEU-3 and BLEU-4 were too low to compare)



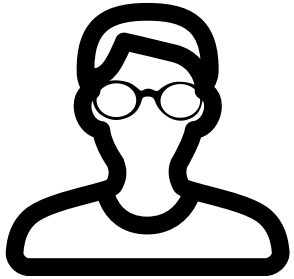
Transformer LMs aren't *entirely* BOW, they can handle *some* more abstract syntactic information





Human Analysis

Evaluator	Accuracy	Macro F1	Acc on D^c	Acc on D^f
X	0.581 ±0.068	0.454	0.649 ±0.102	0.515 ±0.089
Y	0.378 ±0.064	0.378	0.411 ±0.098	0.349 ±0.087



- 200 permuted sentences of varying length
- Annotators are “experts” in NLI

Human Analysis

Evaluator	Accuracy	Macro F1	Acc on D^c	Acc on D^f
X	0.581 ±0.068	0.454	0.649 ±0.102	0.515 ±0.089
Y	0.378 ±0.064	0.378	0.411 ±0.098	0.349 ±0.087

- 200 permuted sentences of varying length - **RoBERTa gets all of them "correct"!**
- Annotators are "experts" in NLI

Note: concurrent work on various perturbations of the GLUE Benchmark finds "turkers can only 'predict' the correct label for invalid examples in 35%" of cases (Gupta et al 2021; AAAI)

Once again, this time, in Chinese!

Just to verify this, we looked into another language…

Similar issue in Chinese OCNLI corpus!

This isn't a tokenization complication, or some quirk of English.

Model	$\mathcal{A}$	$\Omega_{\max}$	$\mathcal{P}^c$	$\mathcal{P}^f$	Ω_{rand}
RoBERTa (large)	**0.784**	**0.988**	0.726	**0.339**	**0.773**
InferSent	0.573	0.931	0.771	0.265	0.615
ConvNet	0.407	0.752	**0.808**	0.199	0.426
BiLSTM	0.566	0.963	0.701	0.271	0.611

Hu, Richardson, Xu, Li, Kuebler, Moss 2020 (EMNLP)
OCNLI: Original Chinese Natural Language Inference

What can we do about it?

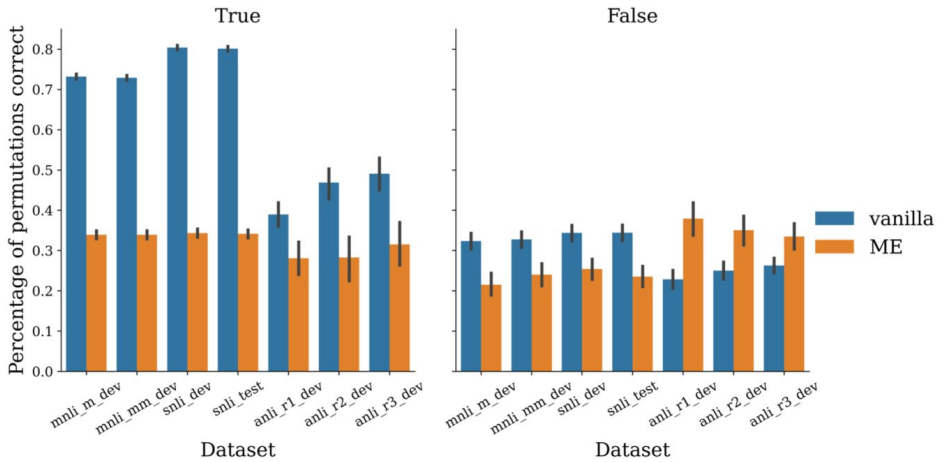
Preliminary attempt : Entropy maximization

Initial Attempt: Max Entropy Training

A simple technique, but it **works**!

- Accuracy is constant while the percentage of accepted permutations reduced considerably!
- However, there's still room to improve!

$$\mathcal{L} = \underset{\theta}{\operatorname{argmin}} \sum_{((p,h),y)} y \log(p(y|(p,h);\theta)) + \sum_{i=1}^{n} \mathbf{H}\left(y|(\hat{p}_i,\hat{h}_i);\theta\right)$$



Similar approach concurrently by Gupta et al 2021

Takeaways:

1. All tested models are largely insensitive to permutations of word order, though humans are not.

2. Reordering words can cause models to flip classification labels

3. Models have learned some distributional information (*POS neighborhood*) that enable them to perform reasonably well under the permuted set up

Thank You

https://arxiv.org/abs/2101.00010
https://github.com/facebookresearch/unlu

It is not enough that models should succeed where humans succeed, they should also fail where humans fail.

AFNLP

McGill | School of Computer Science

Mila

FACEBOOK AI