

UNIVERSITY OF CAMBRIDGE

An Empirical Study on Robustness to Spurious Correlations using Pre-trained Language Models

Tu, Lalwani, Gella & He (TACL 2020)

Simon Roth

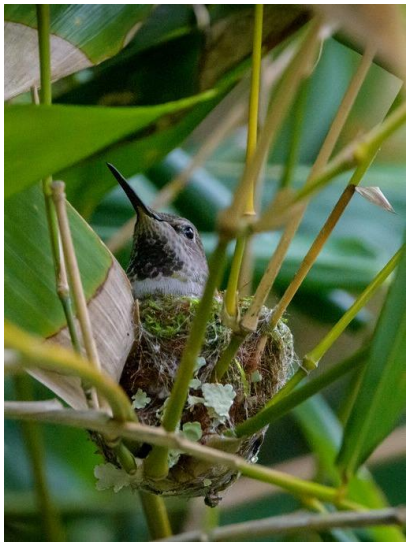
L101: Machine Learning for Language Processing
Michaelmas 2026

The paper in one sentence

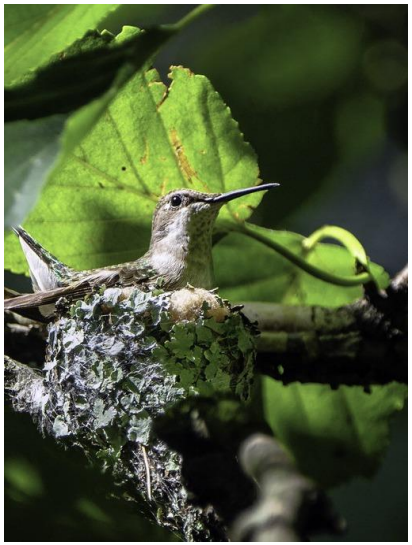
Pre-trained models are more robust to spurious correlations not because they extrapolate to unseen patterns, but because they learn better from the few counterexamples already in the training data.

- Pre-training and MTL help models learn more from the counterexamples in the data, but cannot make up for missing ones
- **Robustness depends on what is in the data**: a call for data diversity

What is a spurious correlation?



bird



bird



bird?

- **Spurious correlation:** a feature that predicts the label in the training data, but does not cause it
- A model that learned “nest ⇒ bird” is right on typical images, and wrong as soon as the pattern breaks

Bird photos: walnerz; Lorie Shaull (Flickr, CC BY 2.0)

The data: two tasks, two shortcuts

Natural Language Inference (NLI)

Does the premise entail the hypothesis?

Train MNLI → test HANS

① Shortcut: high word overlap ⇒ entailment

Paraphrase identification (PI)

Do two sentences mean the same?

Train QQP → test PAWS

② Shortcut: same bag of words ⇒ paraphrase

	Dataset	Size	Heuristic	Input	Label
			Natural language inference		
Train	MNLI	393k	high word overlap ⇒ entailment	P: The doctor mentioned the manager who ran. H: The doctor mentioned the manager.	entailment
1 Test	HANS	30k	high word overlap ⇏ entailment	P: The actors who advised the manager saw the tourists. H: The manager saw the tourists.	non-entailment

	Dataset	Size	Heuristic	Input	Label
			Paraphrase Identification		
Train	QQP	364k	same bag-of-words ⇒ paraphrase	S_1: Bangkok vs Shanghai? S_2: Shanghai vs Bangkok?	paraphrase
2 Test	$PAWS_{QQP}$	677	same bag-of-words ⇏ paraphrase	S_1: Are all dogs smart or can some be dumb? S_2: Are all dogs dumb or can some be smart?	non-paraphrase

Tu et al. (2020), Table 1

- The shortcut works on the training data, but the challenge sets are built so that it fails: a shortcut-only model scores at chance (≈50% on HANS)
- HANS uses three variants of the overlap heuristic: lexical overlap, subsequence, constituent (more on slide 11)

Motivation: pre-training helps, but inconsistently

Model	*Trained on MNLI* In-distribution **MNLI-m**	*Trained on MNLI* Challenging **HANS**	*Trained on QQP* In-distribution **QQP**	*Trained on QQP* Challenging **PAWS$_{QQP}$**
Non pre-trained baselines				
BERT$_{scratch}$	67.9 (0.5)	49.9 (0.2)	83.0 (0.7)	40.6 (1.9)
ESIM	78.1[a]	49.1[a]	85.3[b]	38.9[b]
pre-trained models				
BERT$_{BASE}$(prior)	84.0[c]	53.8[c]	90.5[d]	33.5[d]
BERT$_{BASE}$(ours)	84.5 (0.1)	62.5 (3.4)	90.8 (0.3)	36.1 (0.8)
BERT$_{LARGE}$	86.2 (0.2)	71.4 (0.6)	91.3 (0.3)	40.1 (1.8)
RoBERTa$_{BASE}$	87.4 (0.2)	74.1 (0.9)	91.5 (0.2)	42.6 (1.9)
RoBERTa$_{LARGE}$	89.1 (0.1)	77.1 (1.6)	89.0 (3.1)	39.5 (4.8)

Tu et al. (2020), Table 2

Robust accuracy = accuracy on the challenge set (HANS, PAWS); in-distribution = MNLI-m / QQP dev

Pre-training helps on HANS

- ① From chance without pre-training to up to 77 with it
- ② Larger models and more pre-training data help further

But not on PAWS

- ③ Little or no gain with pre-training

Fine-tuning length matters

- ④ 20 epochs instead of the standard: HANS 53.8 → 62.5

So the **task** and the **fine-tuning length** seem to make a difference.

→ **Where does the robustness come from, and why does it vary?**

Hypothesis: better learners, not extrapolators

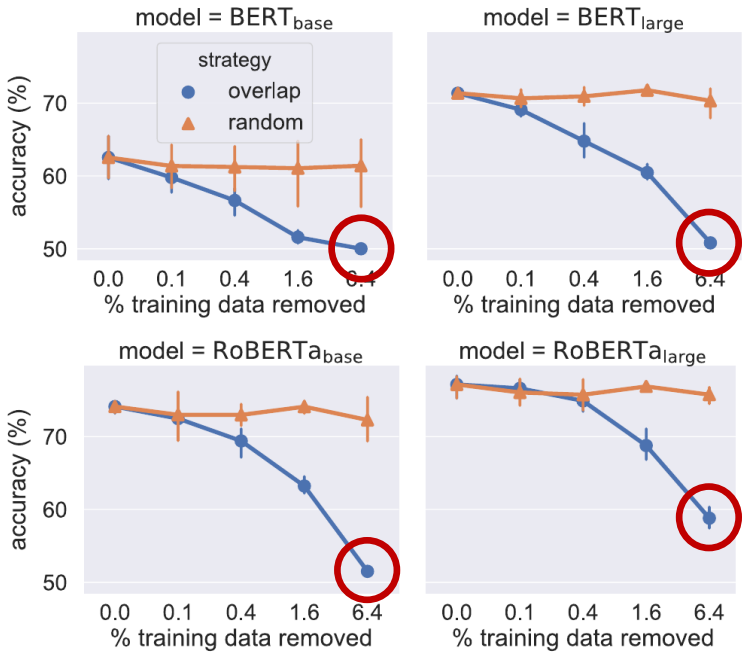
Minority examples = training examples that go against the shortcut (MNLI: 727 HANS-like, QQP: 247 PAWS-like)

What they did

- Remove counterexamples from MNLI, highest overlap first (blue)
- Control: remove random examples (orange)
- Same pre-trained models; only the fine-tuning data shrinks

What they found

- Without counterexamples, most models drop to chance on HANS
- Random removal: no effect
- RoBERTa-large (largest, most pre-training data) degrades more slowly



Tu et al. (2020), Fig. 2 (panels rearranged)

What it means

- **Pre-training alone does not make models robust**
- **Their robust performance depends on the minority examples**
- **Pre-training makes them learn better from few counterexamples ⇒ data diversity matters**

Does the underlying task make a difference?

Why does HANS improve, but PAWS does not?

What they did

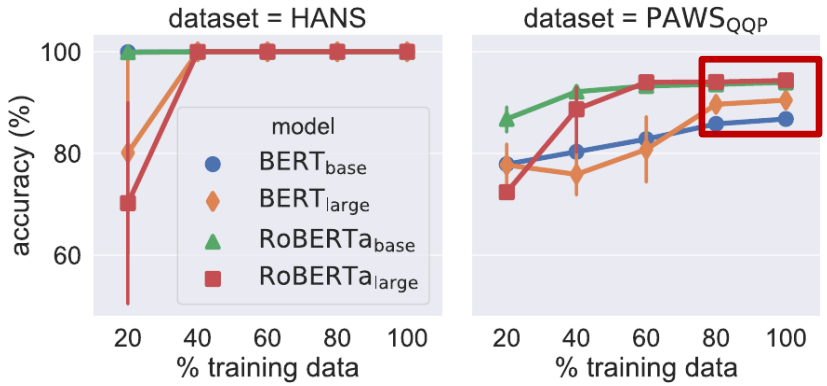
- Trained and fine-tuned directly on each challenge set, with growing amounts of data
- Compare sentence length and complexity

What they found

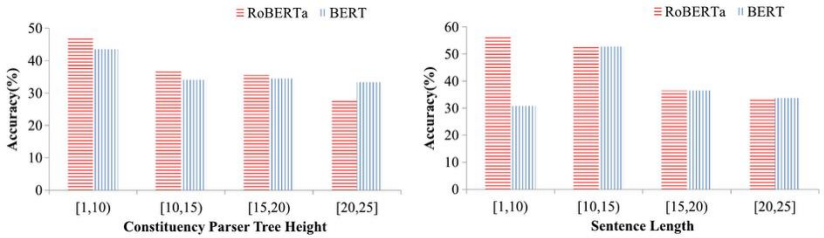
- HANS is learned from little data; PAWS stays hard even with all ~12k examples (Fig. 3)
- PAWS sentences are longer and more complex; accuracy drops with both (Fig. 4)

What it means

- **QQP's few counterexamples are not enough for PAWS**
- **How much is enough augmented data?**
- **Is that a problem for data augmentation? Can you know how many counterexamples a pattern needs?**



Tu et al. (2020), Fig. 3



Tu et al. (2020), Fig. 4

Does longer fine-tuning make a difference?

Why does training for 20 instead of a few epochs help on HANS?

What they did

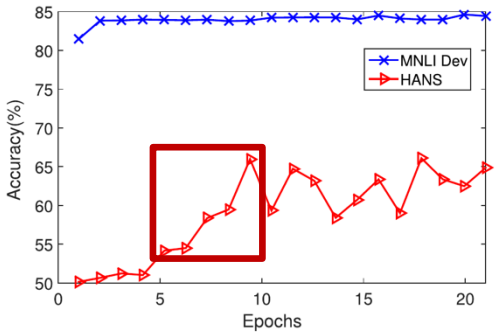
- Track accuracy and loss over 20 epochs (BERT-base, MNLI), for all and for minority examples

What they found

- Dev accuracy plateaus after ~3 epochs; HANS keeps improving until ~10 (Fig. 1)
- Minority examples are learned much later (Fig. 5, red dashed)
- Their dev accuracy rises too: generalisation, not memorisation

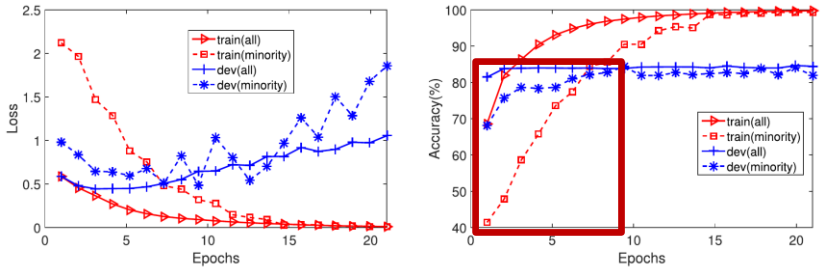
What it means

- **Longer fine-tuning gives the model time to learn the counterexamples**
- **Early stopping on the normal dev set misses robustness**
- **How do you select a good stopping point for robust accuracy?**



(a) NLI/HANS

Tu et al. (2020), Fig. 1a (MNLI dev vs. HANS)



(a) Loss (b) Accuracy

Tu et al. (2020), Fig. 5 (BERT-base, MNLI)

What helps when counterexamples are scarce?

Two options

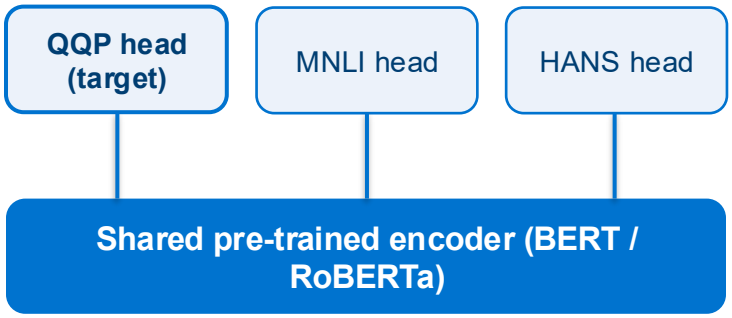
- **Data augmentation:** add counterexamples. Works (McCoy et al. 2019; Zhang et al. 2019), but needs knowledge of the shortcut, and how many are enough is unknown (e.g., Paws vs Hans)
- **Multi-task learning (MTL):** add generic data from related tasks; no knowledge of the shortcut needed → this paper

MTL in the paper

- Train jointly on the target plus auxiliary datasets; evaluate on the target
- NLI and PI as related tasks: both compare the meaning of two sentences
- Shared encoder, one classifier per dataset; the other task's challenge set is part of the auxiliary data

The aim

- **Related data helps the model learn from the target's few counterexamples**



Example: target QQP, auxiliary MNLI, SNLI and HANS (one head each; SNLI omitted here)

Does MTL help, and why?

What they found

- Robust accuracy improves; in-distribution barely changes (Table 3)
- Largest gains on PAWS, where counterexamples are scarcest
- On HANS, mainly the minority type that is more common in the training data

Why?

- Not the auxiliary challenge sets: leaving them out hurts no more than leaving out generic data
- Removing the target's own counterexamples erases most of the gain (Table 8)

What it means

- **MTL helps the model learn from the target's counterexamples (lower sample complexity), but cannot replace them**

		Task = MNLI		*Task = QQP*		
Model	Algo.	In-distribution **MNLI-m**	Challenging **HANS**	In-distribution **QQP**	Challenging **$PAWS_{QQP}$**	Challenging **$PAWS_{Wiki}$**
$BERT_{BASE}$	STL	84.5 (0.1)	62.5 (0.2)	90.8 (0.3)	36.1 (0.8)	46.9 (0.3)
	MTL	83.7 (0.3)	**68.2** (1.8)	91.3 (.07)	**45.9** (2.1)	**52.0** (1.9)
$RoBERTa_{BASE}$	STL	87.4 (0.2)	74.1 (0.9)	91.5 (0.2)	42.6 (1.9)	49.6 (1.9)
	MTL	86.4 (0.2)	72.8 (2.4)	91.7 (.04)	**51.7** (1.2)	**57.7** (1.5)

Tu et al. (2020), Table 3

Removed	**QQP**	**$PAWS_{QQP}$**	Δ
None	91.3 (.07)	45.9 (2.1)	–
random examples			
QQP	91.3 (.03)	44.3 (.31)	−1.6
MNLI	91.4 (.02)	45.0 (1.5)	−0.9
minority examples			
QQP	91.3 (.09)	**38.2** (.73)	−7.7
MNLI	91.3 (.08)	44.3 (2.0)	−1.6

Tu et al. (2020), Table 8 (BERT-base, target QQP)

Evaluation and critique: were the stated aims achieved?

Claim	Verdict	But…
Robustness comes from minority examples	**Yes, for NLI**	• Removal experiment only on MNLI/HANS, not repeated on QQP/PAWS
Longer fine-tuning helps minority groups	**Yes**	• How long is enough? They warn against i.i.d. early stopping but give no criterion
MTL helps by learning more from the target's counterexamples	**Yes**	• On HANS it mainly helps the type that is common in the training data • That the other types are "too rare" is assumed, not tested: maybe they are just harder

Overall: a qualitative, descriptive study. It explains well *why* robustness appears, but gives no rules for how many counterexamples are enough, what makes a pattern too hard, or how long to fine-tune.

Technical reproducibility

Yes: code, exact checkpoints, all hyperparameters, 5 seeds with t-tests. **But:** 5 seeds is thin, as HANS accuracy varies widely across seeds (McCoy et al. 2020)

How does it advance the state of the art?

- **Moves the problem from algorithms to data:**
  - call for diverse data and counterexamples in benchmarks
  - augmentation alone is no fix, since the amount needed depends on the pattern
- **Calls for rethinking standard fine-tuning practice:**
  - how long to train
  - how to select models (the i.i.d. dev set misses robustness)
  - how much data to augment
- **Opened questions taken up by later work:**
  - Lovering et al. 2021: the number of counterexamples needed is predictable from how extractable a feature is
  - Joshi & He 2022: augmented data that varies too few features does not help
  - Kirichenko et al. 2023: features may already be good; retraining only the last layer can suffice (next talk)

Open questions for discussion

- LLM era: do trillion-token corpora contain enough counterexamples for most shortcuts? Are PAWS-like patterns what remains?
- Should we select models on an OOD dev set, and is it still OOD once we do?
- Can “data diversity” be achieved, or will every large dataset hide shortcuts nobody has found yet?

Backup: MTL results by HANS type

What they did

- Compare STL vs. MTL (BERT-base, RoBERTa-base)

What they found

- Robust accuracy improves; in-distribution barely changes (Table 3)
- Largest gains on PAWS, where counterexamples are scarcest
- HANS: helps BERT, not RoBERTa (already learns enough from MNLI)
- HANS gains mainly on the type most MNLI counterexamples belong to; rarer types barely improve (Table 5)

What it means

- **MTL helps where counterexamples are scarce, but cannot create missing ones**
- **Its value is out of distribution, not in-distribution**

		Task = MNLI		*Task = QQP*		
Model	Algo.	In-distribution **MNLI-m**	Challenging **HANS**	In-distribution **QQP**	Challenging **$PAWS_{QQP}$**	Challenging **$PAWS_{Wiki}$**
$BERT_{BASE}$	STL	84.5 (0.1)	62.5 (0.2)	90.8 (0.3)	36.1 (0.8)	46.9 (0.3)
	MTL	83.7 (0.3)	**68.2** (1.8)	91.3 (.07)	**45.9** (2.1)	**52.0** (1.9)
$RoBERTa_{BASE}$	STL	87.4 (0.2)	74.1 (0.9)	91.5 (0.2)	42.6 (1.9)	49.6 (1.9)
	MTL	86.4 (0.2)	72.8 (2.4)	91.7 (.04)	**51.7** (1.2)	**57.7** (1.5)

Tu et al. (2020), Table 3

Model	Algo.	HANS-O	HANS-C	HANS-S
$BERT_{BASE}$	STL	75.8 (4.9)	59.1 (4.8)	52.7 (1.2)
$BERT_{BASE}$	MTL	89.5 (1.9)	61.9 (2.3)	53.1 (1.1)
$RoBERTa_{BASE}$	STL	88.5 (2.0)	70.0 (2.3)	63.9 (1.4)
$RoBERTa_{BASE}$	MTL	90.3 (1.2)	64.8 (3.1)	63.5 (4.9)

Tu et al. (2020), Table 5

O lexical overlap, **C** constituent, **S** subsequence

Backup: Why does MTL help?

(a) Do the auxiliary challenge sets teach the model directly?

- **Test:** leave out one auxiliary dataset at a time
- In-distribution barely changes (blue); only challenge accuracy varies (Δ)
- Leaving out HANS hurts less than leaving out the generic MNLI (red)

⇒ **Not the main reason**

Removed	QQP	PAWS_{QQP}	Δ
None	91.3 (.07)	45.9 (2.1)	–
HANS	91.5 (.06)	45.3 (1.8)	−0.6
MNLI	91.2 (.11)	42.3 (1.8)	−3.6
SNLI	91.3 (.09)	44.2 (1.3)	−1.7

Table 7: BERT-base, target QQP (Table 6: similar for MNLI)

(b) Does MTL help the model learn from the target's own counterexamples?

- **Test:** remove the counterexamples from the target (QQP) or the auxiliary data (MNLI)
- Target's removed: PAWS falls back to near single-task level (solid)
- Auxiliary's removed: same as removing random examples (dashed)

⇒ **Confirmed: MTL lowers the sample complexity for the target's counterexamples**

Removed	QQP	PAWS_{QQP}	Δ
None	91.3 (.07)	45.9 (2.1)	–
random examples			
QQP	91.3 (.03)	44.3 (.31)	−1.6
MNLI	91.4 (.02)	45.0 (1.5)	−0.9
minority examples			
QQP	91.3 (.09)	**38.2** (.73)	−7.7
MNLI	91.3 (.08)	44.3 (2.0)	−1.6

Table 8: BERT-base, target QQP

Backup: MTL setup

The idea

- **Single-task (STL):** fine-tune on the target dataset only
- **Multi-task (MTL):** train jointly on the target plus auxiliary datasets from the same or related tasks

Setup in the paper

- NLI and PI as related tasks: both compare the meaning of two sentences
- Shared encoder, one classifier per dataset; all datasets weighted equally
- Auxiliary: benchmarks plus the other task's challenge set (Table 4)

The aims

- **Knowledge from related tasks helps the model learn from the target's few counterexamples**

		Target	
Auxiliary Datasets	Size	NLI	PI
MNLI	393k		✓
SNLI	549k	✓	✓
QQP	364k	✓	
$\text{PAWS}_{\text{QQP+Wiki}}$	60k	✓	
HANS	30k		✓

Tu et al. (2020), Table 4

