It is not True that Transformers are Inductive Learners: Probing NLI Models with External Negation

Anonymous ACL submission

Abstract

NLI tasks necessitate a substantial degree of logical reasoning; as such, the remarkable performance of SoTA transformers on these tasks may lead us to believe that those models have learned to reason logically. The results presented in this paper demonstrate that (i) models fine-tuned on NLI datasets learn to treat external negation as a distractor, effectively ignoring its presence in hypothesis sentences; (ii) several near-SoTA encoder and encoder-decoder transformer models fail to inductively learn the law of the excluded middle for a single external negation prefix with respect to NLI tasks, despite extensive fine-tuning; (iii) those models which are are able to learn the law of the excluded middle for a single prefix are unable to generalize this pattern to similar prefixes. Given the critical role of negation in logical reasoning, we may conclude from these findings that transformers do *not* learn to reason logically when fine-tuned for NLI tasks. Furthermore, these results suggest that transformers *cannot* inductively learn the role of negation with respect to NLI tasks, calling into question their capacity to fully acquire logical reasoning abilities.

1 Introduction

Natural language inference (NLI) tasks require detecting inferential relations between pairs of sentences (Fyodorov et al., 2000). For NLI datasets such as MultiNLI (MNLI; Williams et al., 2017) and Stanford NLI (SNLI; Bowman et al., 2015), the task proceeds as follows: given a pair of sentences (P, H), an NLI model must determine whether the *premise* P *entails* the *hypothesis* H, H *contradicts* P, or P and H are *neutral* with respect to one another (i.e. P does not entail H and H does not contradict P).

NLI tasks require logical reasoning capabilities that extend beyond basic linguistic competence (Richardson et al., 2020). For example, understanding that *"Jane is travelling to Algeria"* entails *"Jane is travelling to Africa"* requires mereological world knowledge (Hovda, 2009); in particular, an agent must know that Algeria is contained within Africa. To understand that *"Jane is travelling to Algeria"* does *not* entail *"Jane is travelling to Algiers"*, the agent must understand that Algiers is contained within Algeria, but that Algeria is not solely comprised of the city of Algiers.

Due to the considerable amount of reasoning that is required to accomplish these tasks, it is prudent to scrutinize the degree to which current NLI models are *actually* learning to reason logically. McCoy et al.'s (2019) findings suggest, for example, that even state-of-the-art (SoTA) NLI models such as BERT (Devlin et al., 2019) adopt shallow, textual heuristics to achieve high-scoring results on the MNLI dataset, although the MNLI dataset itself is likely to be—at least partially—at fault (possibly because the provided training data is not sufficiently representative of the task; see the discussion in Section 2 below).

This paper investigates SoTA transformer (Vaswani et al., 2017) NLI models' ablility to inductively learn the law of the excluded middle with respect to *external negation* (negation that occurs externally to the proposition that is negated, e.g. *"it is not true that apples are red"*), in order to evaluate the degree to which they have learned to reason logically when performing NLI tasks. Using external negation, it is possible to automatically conduct adversarial attacks on examples from the MNLI and SNLI datasets that modify the examples' class labels in a predictable manner; given a premise, hypothesis, label triple (P, H, L), we may generate an adversarial example $(P, \neg H, L')$, where $L' = neutral$ if $L = neutral$, $L' = contradiction$ if $L = entailment$, and $L' = entailment$ if $L = contradiction$.

Experiments 1 and 2 evaluate the NLI models' inductive learning capacity along two respective axes:

Experiment 1 (Section 3) examines these models' ability to generalize double negation-cancellation to chains of repeated external negation prefixes longer that those seen during inoculation, with respect to a single prefix string. We observe that NLI models struggle to learn this pattern inductively, with many unable to learn it at all. Experiment 2 (Section 4) evaluates the ability of those NLI models which were successfully able to learn the law of the excluded middle for a single external negation prefix to generalize this pattern to prefix strings not seen during fine-tuning. We find that those inoculated models suffer drastic decreases in performance when presented with unseen prefixes; the results of Experiment 3 (Section 5) indicate that this is due to catastrophic forgetting of the similarity between the prefix that they were inoculated against and other, highly similar prefixes.

The experimental results contained in this paper[1] indicate that transformer models do not learn to reason logically when fine-tuned on NLI datasets, lending further support to McCoy et al.'s (2019) hypothesis that they are instead learning to leverage shallow heuristics. In Section 6, we find evidence (Theorem 1) that this failure of transformer models to inductively learn the law of the excluded middle arises from deficiencies in their training procedure and/or the structure (or lack thereof) of their input data, rather than flaws inherent to transformer architectures themselves.

2 Related Work

There is a large body of existing work on probing NLI models to gain insight into their reasoning abilities (Belinkov and Glass, 2019). As mentioned in Section 1, McCoy et al. (2019) find that language models fine-tuned on MNLI learn to leverage shallow heurisics to achieve exceptionally high accuracy on this dataset. Similarly, Chien and Kalita (2020) and Richardson et al. (2020) probe NLI models' performance with respect to specific syntactic and semantic phenomena (e.g. coordination, quantification, monotonicity, etc.). They find that SoTA models fine-tuned on MNLI and SNLI perform poorly on adversarial examples generated to evaluate the models with respect to these phenomena, but can be easily fine-tuned to master the adversarial data, while retaining their high performance on the original datasets.

In all three of these papers, their respective authors utilize the method of inoculation by fine-tuning. Liu et al. (2019a) introduces this paradigm as a technique for differentiating between deficiencies in a model's training data and deficiencies in the model itself. Inoculation by fine-tuning assumes that there is an *original* dataset (divided into train and test splits) and a smaller *challenge/adversarial* dataset (also divided into train and test splits), and that model's performance on the adversarial dataset is significantly lower than on the original dataset. The idea is to fine-tune the model on the adversarial dataset until validation performance on the *original* test set has not improved for five epochs, then measure the newly fine-tuned (*inoculated*) model on the adversarial test set. If the inoculated model maintains its performance on the original test set and performs (nearly) as well on the adversarial test set, this suggests that the model's poor performance on the adversarial data was due to flaws (e.g. a lack of diversity) in the original training data. Conversely, if the model's performance on the adversarial test set remains significantly worse than on the original data after inoculation, this suggests that its poor performance on the adversarial data is due to a deficiency in the model itself.

This paper probes various NLI models' logical reasoning abilities—in particular with respect to external negation—using adversarial attacks along with the inoculation by fine-tuning paradigm. Unlike most varieties of adversarial attack, which seek to perturb input examples without altering their class labels, the external negation prefixes used in Experiments 1-3 (Sections 3, 4, 5) do alter the examples' class labels, albeit in a predictable manner. This is similar to the adversarial attack that Niven and Kao (2019) conduct on BERT models with respect to the Argument Reasoning Comprehension Task (Habernal et al., 2018); these authors find that BERT *cannot* be inoculated against such adversarial attacks, and conclude that transformer models' inability to ground text to real-world concepts presents an insurmountable barrier to their logical-reasoning abilities.

In a similar vein, Naik et al. (2018) conduct "stress tests" on NLI models by concatenating logical distractor strings (e.g. *"and false is not true"*) to the input examples, and find that such distractors drastically reduce SoTA NLI models' performance on these tasks. While these authors investigate NLI models' performance with respect to logical

[1] All code available on GitHub: [link removed for anonymity]

reasoning, their experiments regarding negation are limited to negation items appearing in these distractor terms, rather than negating the original hypothesis sentence itself.

Yuan et al. (2023) examine pretrained language models' (PLMs) *deductive* reasoning abilities via cloze tests. These authors find that PLMs are unable to fully generalize rules of logical deduction to arbitrary contexts. Furthermore, they observe that these models struggle to differentiate between positive statements and their negated counterparts, in line with a wide body of recent literature suggesting that transformers have difficulty processing and comprehending negation (e.g. Laverghetta Jr. et al., 2021; Rogers et al., 2020; Ettinger, 2020). Of particular interest to this work, they find that while inoculating PLMs for deductive reasoning tasks improves performance, it results in catastrophic forgetting of previous knowledge. Similarly, in Sections 4 and 5 of this paper, we find that inoculating pretrained NLI models against adversarial external negation prefixes causes catastrophic forgetting of prior knowledge of their similarity to related prefixes.

In an experiment highly related to the present work, Laverghetta Jr. and Licato (2022) probe NLI models' performance with respect to negation, and find that the models struggle to contend with certain types of negation more so than others. In line with the results we observe in Section 3, they find that the models have difficulty inoculating against those problematic negation categories. Unlike the experiments in this paper, Laverghetta Jr. and Licato (2022) do not conduct adversarial attacks involving negation, but rather use examples drawn from NLI datasets that already contain negation.

Unique to this work is the evaluation of transformers' ability to learn the law of the excluded middle and our finding that, while many cannot learn this pattern, a few transformer NLI models are in fact able to inductively learn the law of the excluded middle for a single external negation prefix. Additionally, the results of Experiments 2 and 3 (Sections 4 and 5), extend Yuan et al.'s (2023) results (regarding catastrophic forgetting resulting from inoculation in the context of deductive reasoning tasks) to double negation-cancellation in the setting of NLI tasks. Finally, Theorem 1 (see Section 6) is the first known proof that there exists (at least, in principle) an encoder transformer capable of modeling the law of the excluded middle for arbitrary-length sequences of any combination of external negation prefixes with respect to any NLI dataset. This theorem sheds further light on evidence in the literature (Niven and Kao, 2019; Naik et al., 2018; Yuan et al., 2023; Laverghetta Jr. et al., 2021; Rogers et al., 2020; Ettinger, 2020; Laverghetta Jr. and Licato, 2022, etc.) indicating that transformers are unable to model negation, suggesting that this observed failure is not due to an inherent flaw in transformer architectures themselves, but instead may be due to deficiencies in their training procedure and/or the structure of their input data.

3 Experiment 1

Experiment 1 probes six different transformer NLI models' ability to inductively learn the law of the excluded middle with respect to external negation. The DeBERTa (He et al., 2020) model, denoted $DeBERTa_S$[2], is DeBERTa-large fine-tuned on SNLI. The first BART (Lewis et al., 2020) model, denoted $BART_M$[3], is BART-large fine-tuned on MNLI, while the second, $BART_{SMFA}$[4], is BART-large fine-tuned on MNLI, SNLI, FEVER (Thorne et al., 2018), and ANLI (Nie et al., 2020). The first RoBERTa (Liu et al., 2019b) model, $RoBERTa_M$[5], is RoBERTa-large fine-tuned on MNLI, and the second, $RoBERTa_S$[6], is RoBERTa-large fine-tuned on SNLI, while the third, $RoBERTa_{SMFA}$[7], is RoBERTa-large fine-tuned on SNLI, MNLI, FEVER, and ANLI.

3.1 Experimental Setup

For each $1 \leq n \leq 5$ and each NLI dataset $D \in \{\text{MNLI}, \text{SNLI}\}$, let $D^{\leq n}_{train}$ and $D^{\leq n}_{dev}$ denote the $\leq n$-fold adversarial training and development sets, respectively. $D^{\leq n}_{train}$ and $D^{\leq n}_{dev}$ are generated from examples randomly drawn from the original datasets' training splits: $\text{MNLI}^{\leq n}_{train}$ consists of 3271 entailment, neutral, and contradiction examples (9813 total), $\text{SNLI}^{\leq n}_{train}$ consists of 9999 examples (3333 of each class), $\text{MNLI}^{\leq n}_{dev}$ consists of 4905 examples (1635 of each class), and $\text{SNLI}^{\leq n}_{dev}$ consists of 4998 examples (1666 of each class).

Each of the two datasets contains many examples that are not complete sentences, but rather sentence

[2] https://huggingface.co/pepa/deberta-v3-large-snli
[3] https://huggingface.co/facebook/bart-large-mnli
[4] https://huggingface.co/ynie/bart-large-snli_mnli_fever_anli_R1_R2_R3-nli
[5] https://huggingface.co/roberta-large-mnli
[6] https://huggingface.co/pepa/roberta-large-snli
[7] https://huggingface.co/ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli

fragments, in which case the external negation prefix T_{NT} = *"it is not true that"* is grammatically nonsensical. To account for this, the pool of possible examples to be included into the adversarial datasets consists only of those in which the hypothesis H is a complete sentence. If the first word in H is (part of) a named entity (as determined by SpaCy's *EntityRecognizer*[8] named entitiy recognition pipeline), then the adversarial hypothesis $H_{adv} = (T_{NT})^n H$. If the first word in H does *not* belong to a named entity, then $H_{adv} = (T_{NT})^n H_0$, where H_0 is formed from H by lower-casing the first character. This is to control for potential confounding factors due to irregular capitalization.

For each $1 \leq n \leq 5$ and each $1 \leq k \leq n$, $1/n^{th}$ of the examples in each class in $D^{\leq n}_{train}$ and $D^{\leq n}_{dev}$ are k-fold negated by prefixing the adversarial trigger T_{NT} = *"it is not true that"* to the original hypothesis sentence. For example, in $D^{\leq 5}_{train}$, $1/5^{th}$ of the examples in each class are 5-fold negated (by converting (P, H) to $(P, T_{NT}T_{NT}T_{NT}T_{NT}T_{NT}H)$), $1/5^{th}$ are 4-fold negated, $1/5^{th}$ are 3-fold negated, etc.

Finally, for all $m > 1$ and each NLI dataset $D \in \{\text{MNLI}, \text{SNLI}\}$, let D^m_{test} denote the m-fold test set. The procedure for generating D^m_{test} is nearly identical to that of $D^{\leq m}_{train}$ and D^m_{dev} ($|D^m_{test}| = |D^m_{dev}|$), with the exception that D^m_{test} consists *only* of m-fold externally-negated examples.

For all $1 \leq n \leq 5$, each NLI model was inoculated against the adversarial sets $D^{\leq n}_{train}$ and $D^{\leq n}_{dev}$. Following the paradigm of inoculation by fine-tuning, the models were fine-tuned $D^{\leq n}_{train}$, and validated at each epoch on the *original* NLI dataset's development split, with early-stopping if validation performance does not improve after five epochs. Once inoculated on the $\leq n$-fold external negation data, the models were evaluated on D^m_{test} for multiple values of $m > n$. This is to evaluate the degree to which the models are able to generalize the law of the excluded middle beyond the number of external negation prefixes seen during inoculation: given an original example (P, H, L) which is converted to an adversarial example $(P, (T_{NT})^n H, L')$, then $L' = L$ if n is even, and *contradiction* flips to *entailment* (and vice-versa) if n is odd.

Each model was evaluated and inoculated on the adversarial datasets generated from the dataset(s) that the model was originally fine-tuned on: $BART_M$ and $RoBERTa_M$ were evaluated on $\text{MNLI}^n_{train/dev/test}$, $RoBERTa_S$ and $DeBERTa_S$ on $\text{SNLI}^n_{train/dev/test}$, and $BART_{SMFA}$ and $RoBERTa_{SMFA}$ on both $\text{MNLI}^n_{train/dev/test}$ and $\text{SNLI}^n_{train/dev/test}$. All models were fine-tuned with a batch size of 64 at a learning rate of 10^{-5}.

3.2 Results and Discussion

For the sake of brevity, model original/adversarial development set accuracies pre- and post-inoculation are located in Appendix C.1. Effectively all models were able to inoculate against the $\leq n$-fold external negation data for all $1 \leq n \leq 5$ (with the notable exception of $BART_M$, which struggled for $n \in \{1, 5\}$); they retain their high-performing accuracy on the original development sets, and perform as well (or nearly so) on the challenge development sets after inoculation.

However, the models struggled to generalize this knowledge to m-fold negation for values of $m > n$. Table 1 reports average model accuracy (individual model accuracies are located in Appendix C.2) on $m{>}n$-fold external negation after $\leq n$-fold inoculation for $1 \leq n \leq 3$, $2 \leq m \leq 6$. A clear pattern emerges in this table: before any inoculation, we observe high model accuracy (~80%) on the m-fold negtion data for even values of m, and near-random-chance accuracy (~34%) for odd values of m. This indicates that, before inoculation, the models were essentially entirely ignoring the external negation prefixes and treating them as distractors; m-fold negation does not alter the class label for even values of m, and so a model treating the prefix as a distractor will retain high accuracy on those examples, purely by chance. To reiterate: these models—ostensibly fine-tuned on a logical-reasoning task—have learned to *entirely ignore external negation* when predicting inferential relations.

Furthermore, when inoculated against 1-fold external negation, the pattern reverses: we note near-random-chance accuracy for *even* values of m, and high accuracy for *odd* values of m. After 1-fold inoculation, the models have learned to treat any m-fold external negation prefix as equivalent to a 1-fold (i.e. single) prefix.

Interestingly, after $\leq$2-fold inoculation, the models revert to the original pattern of high accuracy for even values of m, and poor performance for odd values. Despite being trained on both 1- and 2-fold external negation, the models merely mem-

[8] https://spacy.io/api/entityrecognizer

m-fold test	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.72	0.39	—	—
3	0.36	0.86	0.32	—
4	0.84	0.39	0.95	0.35
5	0.32	0.82	0.32	0.91
6	0.86	0.43	0.95	0.35

Table 1: Average accuracy across all models on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$). For the sake of brevity, individual results for each model are located in Appendix C.2. However, individual model accuracies largely do not deviate from the mean values in this table.

Model	m-fold test	No inoc.	≤4-fold inoc.
$BART_M$	5	0.33	0.34
$RoBERTa_M$	5	0.32	0.34
$DeBERTa_S$	5	0.30	0.33
$RoBERTa_S$	5	**0.34**	**0.89**
$BART_{SMFA}$	5	0.30	0.32
$RoBERTa_{SMFA}$	5	**0.32**	**0.79**
$BART_M$	6	0.86	0.93
$RoBERTa_M$	6	0.89	0.95
$DeBERTa_S$	6	0.88	0.95
$RoBERTa_S$	6	**0.83**	**0.93**
$BART_{SMFA}$	6	0.86	0.93
$RoBERTa_{SMFA}$	6	**0.84**	**0.95**

Table 2: Accuracy for all models on m>n-fold external negation after ≤4-fold inoculation ($m \in \{5, 6\}$).

Model	m-fold test	No inoc.	≤5-fold inoc.
$BART_M$	6	0.86	0.34
$RoBERTa_M$	6	**0.89**	**0.91**
$DeBERTa_S$	6	0.88	0.31
$RoBERTa_S$	6	**0.83**	**0.94**
$BART_{SMFA}$	6	0.86	0.30
$RoBERTa_{SMFA}$	6	**0.84**	**0.95**
$BART_M$	7	0.32	0.93
$RoBERTa_M$	7	**0.32**	**0.96**
$DeBERTa_S$	7	0.28	0.95
$RoBERTa_S$	7	**0.36**	**0.94**
$BART_{SMFA}$	7	0.29	0.92
$RoBERTa_{SMFA}$	7	**0.31**	**0.95**

Table 3: Accuracy for all models on m>n-fold external negation after ≤5-fold inoculation ($m \in \{6, 7\}$).

orize the effect of 1-fold negation on class labels, and do not generalize to odd values of $m > 1$. A similar pattern emerges after ≤3-fold inoculation; after fine-tuning on 1-, 2-, and 3-fold external negation, the models memorize the effect (or lack thereof) of 2-fold negation on class labels, and do not generalize to even values of $m > 2$.

However, Table 2 indicates that, after ≤4-fold inoculation, two of the RoBERTa models ($RoBERTa_S$ and $RoBERTa_{SMFA}$) do in fact inductively learn to repeatedly cancel double negation for values of $m > 4$. After ≤5-fold inoculation, $RoBERTa_M$ also learns the desired pattern (see Table 3); all three RoBERTa models have inductively learned the law of the excluded middle for arbitrary values of m.

Given all six models' difficulty with inoculation against m-fold external negation (for arbitrary values of m), it is reasonable to question the RoBERTa models' ability to generalize the negation-cancellation patterns that they have learned after ≤5-fold inoculation to external negation strings beyond the trigger T_{NT} = *"it is not true that"* that they saw during inoculation. The following experiment (Section 4) evaluates the three RoBERTa models' ability to repeatedly cancel double negation with the prefix T_F = *"it is false that"*, after they have been fine-tuned on $D^{\leq 5}_{train}$ (i.e. ≤5-fold *"it is not true that"* prefixes).

4 Experiment 2

This experiment restricts its analysis to the three RoBERTa models, as they were the only models of the six evaluated in Experiment 1 (Section 3) that were able to fully generalize m-fold negation-cancellation to arbitrary values of $m > 5$.

4.1 Experimental Setup

For all $m \geq 1$ and each NLI dataset $D \in \{\text{MNLI}, \text{SNLI}\}$, let D^m_F denote the m-fold adversarial test set. Each D^m_F was created in an identical manner to the m-fold adversarial test sets D^m_{test} defined in Section 3.1 above: D^m_F consists only of examples (drawn from the dataset's original development split) modified to have m-fold externally-negated hypothesis sentences with an equal number of examples per class label ($|D^m_F| = |D^m_{test}|$).

However, in place of the adversarial trigger T_{NT} = *"it is not true that"* used in D^m_{test}, in this experiment D^m_F was generated using the trigger T_F = *"it is false that"*. These two triggers are effectively semantically equivalent; the phrase *"not true"* has simply been replaced by the synonymous *"false"*. Assuming that the models have truly learned the law of the excluded middle, we should expect to see similar performance on D^m_F to that of D^m_{test}.

After inoculation on the ≤5-fold T_{NT} external negation data, each of the three RoBERTa models ($RoBERTa_S$, $RoBERTa_M$, $RoBERTa_{SMFA}$) was evaluated on D^m_F for all $1 \leq m \leq 8$. As in the procedure for Experiment 1 (see Section 3.1), each model was evaluated on the adversarial datasets generated from the dataset(s) that the model was originally fine-tuned on.

4.2 Results and Discussion

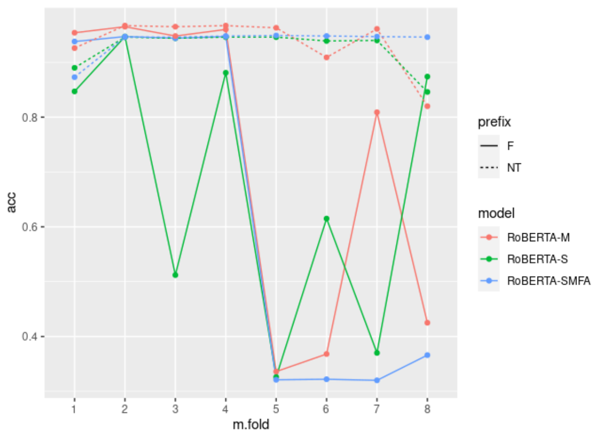

Figure 1: Accuracy for the ≤5-fold T_{NT}-inoculated RoBERTa models on m-fold externally-negated examples with T_{NT} (dashed) and T_F (solid).

Figure 1 shows the results of this experiment: $RoBERTa_S$ failed to generalize the law of the excluded middle from T_{NT} to T_F for values of $m >$ 2, while $RoBERTa_M$ and $RoBERTa_{SMFA}$ experience precipitous decreases in accuracy at $m =$ 5 (and erratic accuracy thereafter). Clearly, while these models can generalize external negation-cancellation to arbitrary-length repeated T_{NT} prefixes, they cannot extend this pattern to near-synonymous prefixes.

We may object that the models have failed to learn the pattern for T_F because they did not see it during inoculation. This objection may be valid, but belies the critical point: *these models have failed to generalize the law of the excluded middle* from T_{NT} to T_F. While the models very well may learn to cancel external negation prefixes after fine-tuning on all possible sequences of this type (see the discussion in Section 6), at that point they are not learning—but rather memorizing—the pattern.

The results of this experiment beg the question as to *why* the RoBERTa models cannot fully generalize the law of the excluded middle from T_{NT} to T_F. The following experiment (Section 5) examines the embeddings generated by the RoBERTa models pre- and post-inoculation, shedding light on the root of their failure to generalize the law of the excluded middle to arbitrary prefixes.

5 Experiment 3

As in Experiment 2 (Section 4), this experiment restricts its analysis to the three RoBERTa models.

5.1 Experimental Setup

As mentioned above, this experiment probes the embeddings that these models generate before and after ≤5-fold T_{NT} inoculation. The experiment proceeds as follows: for each dataset $D \in \{\text{MNLI}, \text{SNLI}\}$, take a subset D' of the original development set (D' contains ~50-100 examples of each class, depending on the size of the dataset). For each $1 \leq m \leq 8$, generate $(D')^m_{NT}$ and $(D')^m_F$ by prefixing $(T_{NT})^m$ and $(T_F)^m$ to each hypothesis sentence (respectively), and compute the cosine similarity between the (mean-pooled) embeddings of $(T_{NT})^m H_i$ and $(T_F)^m H_i$ for each $(P_i, H_i) \in D'$.

For even values of m, compute the (respective) cosine similarities between $(T_{NT})^m H_i$ and $(T_F)^2 H_i$ (as all three models retain high accuracy on T_F prefixes for $m = 2$); $(T_{NT})^2 H_i$ and $(T_F)^m H_i$; $(T_F)^m H_i$ and H_i (for even m, $(T_F)^m H_i$ should be synonymous with H_i); and $(T_{NT})^m H_i$ and H_i, for each premise, hypothesis pair $(P_i, H_i) \in D'$.

For odd values of m, compute the (respective) cosine similarities between $(T_{NT})^m H_i$ and $(T_{NT})^1 H_i$ (for odd m, $(T_{NT})^m H_i$ should be synonymous with $(T_{NT})^1 H_i$); $(T_F)^m H_i$ and $(T_F)^1 H_i$; $(T_{NT})^m H_i$ and $(T_F)^1 H_i$; and $(T_F)^m H_i$ and $(T_{NT})^1 H_i$.

As in Experiments 1 and 2 (Sections 3 and 4, respectively), each model was evaluated using the adversarial datasets generated from the dataset(s) that the model was originally fine-tuned on.

5.2 Results and Discussion

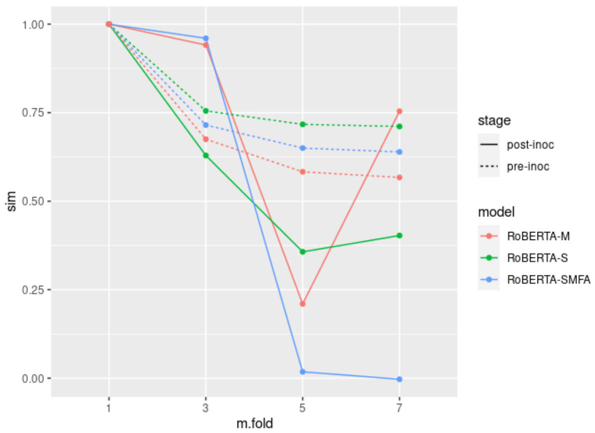

Figure 2: Mean cosine similarity between $(T_F)^n H_i$ and $(T_F)^1 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

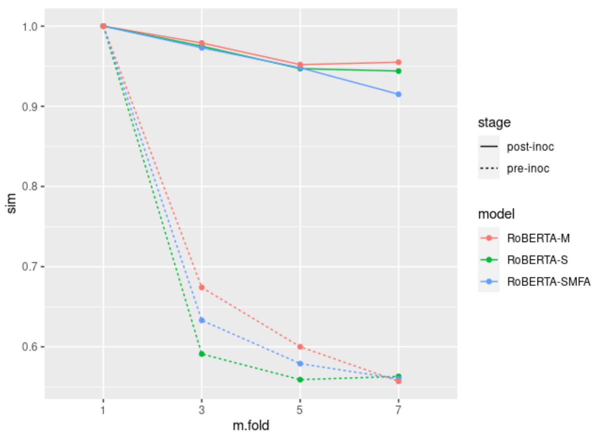

Figure 3: Mean cosine similarity between $(T_{NT})^n H_i$ and $(T_{NT})^1 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

For the sake of brevity, Appendix B reports the majority of the results of this experiment.

We observe that ≤5-fold inoculation drastically increases the similarity between $(T_{NT})^m H_i$ and $(T_{NT})^1 H_i$ for all three models for odd values of m (Figure 3), but *decreases* the similarity between $(T_F)^m H_i$ and $(T_F)^1 H_i$ for $m \geq 5$ (Figure 2). The results are analogous for even values of m (see Figures 4-7 in the appendix).

Additionally, as m increases, we observe decreases in mean cosine similarity for the inoculated models between $(T_F)^m H_i$ and $(T_{NT})^2 H_i$, and $(T_F)^m H_i$ and $(T_{NT})^1 H_i$ (see Figures 8 and 10 in the appendix, respectively). We also observe decreases in cosine similarity between $(T_F)^m H_i$ and $(T_{NT})^m H_i$ for the inoculated models for even and odd $m > 4$ (see Figure 12 in the appendix).

Model	Before	After
$RoBERTa_M$	0.996	0.268
$RoBERTa_S$	0.996	0.712
$RoBERTa_{SMFA}$	0.996	0.646

Table 4: Cosine similarity between the RoBERTa models' (mean-pooled) embeddings of the strings *"false"* and *"not true"* before and after ≤5-fold inoculation.

These results indicate that learning to cancel repeated double negation with respect to the external negation prefix T_{NT} has lead to catastrophic forgetting. In particular, it seems that learning to cancel double negation for T_{NT} has drastically altered the models' encodings of the string *"not true"* to adapt to the task at hand, pulling its representation in the embedding space away from those of other negation phrases such as *"false"*. This conjecture is supported by the results in Table 4: we observe that—before inoculation—the models' representations of the strings *"not true"* and *"false"* are nearly identical. However, after ≤5-fold T_{NT} inoculation, the models' representations of the two strings are significantly further apart in the embedding space.

Furthermore, the results of this experiment indicate that the models have not learned the linguistic function of negation during pre-training or original fine-tuning on the MNLI and SNLI datasets, analogous to the findings of Yuan et al. (2023) with respect to deductive reasoning tasks. Aside from the results in Table 1 indicating that these NLI models simply treat external negation prefixes as distractors (before inoculation), we note that if the models already understood the logical function of prefixes such as *"it is not true that"*, then further refining the models' knowledge of the function of that prefix (i.e. fine-tuning on the ≤5-fold T_{NT} data) should not significantly alter its representation in the embedding space relative to highly similar prefixes such as *"it is false that"*, contrary to what we observe in Table 4.

6 Discussion

The question arises as to *why* these models are unable to inductively learn the law of the excluded middle: is this failure due to transformer architectures themselves, or are inadequacies in their training regimens and/or the structure (or lack thereof) of their input data at fault?

Theorem 1 proves that (encoder) transformer architectures are in fact capable of modeling the law of the excluded middle (at least, with respect to NLI tasks) for arbitrary-length sequences of any combination of external negation prefixes—note that the NLI datasets, transformer models (with the exception of BART), and set of external negation prefixes used in Experiments 1-3 satisfy the assumptions of Theorem 1. This suggests that these transformer NLI models' failure to inductively learn the law of the excluded middle is *not* due to a deficiency in transformer architectures *per se*.

Theorem 1. *Let $D = \{(P_i, H_i, L_i)\}_{i \in I}$ be a finite-cardinality NLI dataset, and for any NLI model M, let $Acc(M, D)$ denote the classification accuracy of M on D. Let Σ' be a finite alphabet such that $D \subset (\Sigma')^* \times (\Sigma')^* \times \Lambda$ (where $\Lambda = \{\mathcal{E}, \mathcal{N}, \mathcal{C}\}$ denotes the set of labels). Let $N \subset (\Sigma')^*$ be any finite-cardinality set of external-negation prefixes such that no prefix is a substring of one or more other prefixes*[9].

Then there exists an alphabet $\Sigma \supset \Sigma'$ and an injective $f\colon (\Sigma')^* \to \Sigma^*$ such that for any fixed (finite) $w > max_{i \in I}|P_i H_i|$ and any fixed-precision transformer encoder (with an NLI classification head) T, there exists a fixed-precision transformer encoder T' such that T' matches the accuracy of T on D and on any dataset D' formed by prefixing any $\eta \in N^*$ to each hypothesis sentence in D[10].

Proof. Appendix A. □

Critically, the proof of Theorem 1 relies on a function f that re-structures the input data, suggesting that the structure (or lack thereof) of purely textual data may be insufficient for transformers to inductively learn to model the law of the excluded middle. Furthermore, Theorem 1 merely states that there exists an encoder transformer capable of modeling the law of the excluded middle for external negation with respect to NLI tasks; it makes no claim regarding its architectural configuration (i.e. layer size, floating-point precision, etc.). It may be the case that the transformer models of Experiments 1-3 do not have the specific architecture required to accomplish this task.

The proof of Theorem 1 also does not make any claims regarding the (inductive) *learnability* of these tasks. It may be the case that the specific parameter values required to model the role of (external) negation in the context of NLI tasks cannot be reached by training on any NLI dataset using gradient descent or any other currently known training procedures. It may also be the case that the function of (external) negation is in fact learnable, but only via the brute-force approach of training these models on multiple-fold external negation for every such prefix—in other words, (encoder) transformers may not be capable of *inductively* learning the law of the excluded middle.

7 Conclusion

The results of Experiments 1-3 (Sections 3, 4, 5) demonstrate that near-SoTA transformer NLI models struggle to inductively learn the law of the excluded middle. Furthermore, the results of Experiment 1 (Section 3) strongly suggest that all six NLI models studied in this work learned to treat the external negation prefix *"it is not true that"* as a distractor when initially fine-tuned on the NLI dataset(s) (see Table 1). Experiment 1 also suggests that DeBERTa and BART models are incapable of learning to inductively generalize the law of the excluded middle, despite extensive fine-tuning.

These findings lend further support to a large body of existing evidence (e.g. Niven and Kao, 2019; Naik et al., 2018; Yuan et al., 2023; Laverghetta Jr. et al., 2021; Rogers et al., 2020; Ettinger, 2020; Laverghetta Jr. and Licato, 2022) indicating that transformers are unable to model the meaning of negation. Unique to this work is our finding that certain encoder transformers (in particular, RoBERTa) can learn the law of the excluded middle for a single external negation prefix.

While the three RoBERTa models did manage to grasp the function of the prefix *"it is not true that"*, the process of learning this behavior resulted in catastrophic forgetting, entirely inhibiting the generalization of this pattern to the highly similar prefix *"it is false that"* (see Sections 4 and 5).

However, Theorem 1 proves that encoder transformers are—in principle—capable of modeling the law of the excluded middle for arbitrary-length sequences of any combination of external negation prefixes with respect to any NLI dataset. This suggests that these models' inability to inductively learn the law of the excluded middle may not be a consequence of their transformer architectures, but rather may result from the structure of the input data and/or the procedure used to train them.

[9]Formally: for all $\eta \in N$, $\eta', \eta'' \in (N - \{\eta\})^*$, there does not exist i, j such that $\eta = \eta'_{i:} \mid\mid \eta''_{:j}$

[10]Formally: $Acc(T', f(D)) = Acc(T, D)$, and for any $\eta \in N^*$ such that $max_{i \in I}|P_i \eta H_i| \leq w$: $Acc(T', \{f(P_i \eta H_i)\}_{i \in I}) = Acc(T, D)$

8 Limitations

While Experiments 1-3 (Sections 3, 4, 5) probe a variety of encoder and encoder-decoder transformers, they do not consider decoder-only models such as LLaMa-2 (Touvron et al., 2023) or GPT-3 (Brown et al., 2020); evaluation of decoder transformers is left to future work. Additionally, these experiments only utilize MNLI and SNLI for adversarial data generation and evaluation, although both datasets have been shown to consist of non-representative data and contain annotation artifacts that permit models to achieve high performance by leveraging shallow heuristics (McCoy et al., 2019; Richardson et al., 2020). However, the use of more challenging NLI datasets such as ANLI was precluded by all six models' (including those fine-tuned on ANLI) already-poor performance on the ANLI test set prior to any adversarial attacks.

The main limitation regarding the adversarial attacks themselves is the fact that they consist of only two external negation prefixes: *"it is true that"* and *"it is false that"*. While this suffices to demonstrate the models' inability to inductively learn the law of the excluded middle and/or generalize this knowledge to similar prefixes, future work should involve similar experiments conducted using a wider variety of adversarial triggers.

Note that Theorem 1 applies only to *encoder* transformers, as the proof is formulated using a variant of first-order logic (FOC[+;MOD]; Immerman, 2012) that has only been shown to be an upper-/lower-bound for fixed-precision encoder transformers (Chiang et al., 2023). Additionally, the proof of Theorem 1 requires a fixed input length w. While the input sequence length of all "real-world" transformers is practically bounded by the quadratic growth rate of their self-attention mechanism (Beltagy et al., 2020), this assumption of a fixed input size still represents a limitation in the expressive power of the theorem.

References

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

David Chiang, Peter Cholak, and Anand Pillay. 2023. Tighter bounds on the expressivity of transformer encoders. In *Proc. ICML*.

Tiffany Chien and Jugal Kalita. 2020. Adversarial analysis of natural language inference systems. In *2020 IEEE 14th International Conference on Semantic Computing (ICSC)*, pages 1–8. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Yaroslav Fyodorov, Yoad Winter, and Nissim Francez. 2000. A natural logic inference system. In *Proceedings of the 2nd Workshop on Inference in Computational Semantics (ICoS-2)*.

Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. 2018. SemEval-2018 task 12: The argument reasoning comprehension task. In *Proceedings of the 12th International Workshop on Semantic Evaluation*, pages 763–772, New Orleans, Louisiana. Association for Computational Linguistics.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2020. Deberta: Decoding-enhanced bert with disentangled attention. *arXiv preprint arXiv:2006.03654*.

Paul Hovda. 2009. What is classical mereology? *Journal of Philosophical Logic*, 38:55–82.

Neil Immerman. 2012. *Descriptive complexity*. Springer Science & Business Media.

Antonio Laverghetta Jr. and John Licato. 2022. Developmental negation processing in transformer language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 545–551, Dublin, Ireland. Association for Computational Linguistics.

Antonio Laverghetta Jr., Animesh Nighojkar, Jamshidbek Mirzakhalov, and John Licato. 2021. Can transformer language models predict psychometric properties? In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 12–25, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2171–2179, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Aakanksha Naik, Abhilasha Ravichander, Norman Sadeh, Carolyn Rose, and Graham Neubig. 2018. Stress test evaluation for natural language inference. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2340–2353, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial NLI: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901, Online. Association for Computational Linguistics.

Timothy Niven and Hung-Yu Kao. 2019. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy. Association for Computational Linguistics.

Kyle Richardson, Hai Hu, Lawrence Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8713–8721.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in BERTology: What we know about how BERT works. *Transactions of the Association for Computational Linguistics*, 8:842–866.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Zhangdie Yuan, Songbo Hu, Ivan Vulić, Anna Korhonen, and Zaiqiao Meng. 2023. Can pretrained language models (yet) reason deductively? In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1447–1462, Dubrovnik, Croatia. Association for Computational Linguistics.

A Proof of Theorem 1

A.1 FOC[+;MOD]

Chiang et al. (2023) prove that FOC[+;MOD] (a variant of first-order logic defined over strings over a finite alphabet Σ; see Immerman, 2012) is both an upper bound for fixed-precision transformer encoders and a lower bound for arbitrary-precision encoder transformer encoders, in the sense that every language that is recognizable by a fixed-precision encoder transformer classifier is definable by a sentence of FOC[+;MOD] (Chiang et al., 2023, Theorem 2), and every language defined by a sentence of FOC[+;MOD] is recognizable by an (arbitrary-precision) encoder transformer classifier (Chiang et al., 2023, Theorem 5). Given an FOC[+;MOD] formula ϕ, the language defined by ϕ is the set of all strings $\sigma \in \Sigma^*$ such that ϕ holds with respect to σ.

The syntax of FOC[+;MOD] consists of two sorts:

- *Positions*: (positive) integer variables p that range over positions in strings σ.
- *Counts*: variables x ranging over the rational numbers $\mathbb{Q}$, and terms $c_0 + c_1x_1 + \cdots + c_nx_n$, where each c_i is a (constant) rational number and each x_i is a count variable.

Formulas of FOC[+;MOD] are defined as one of:

- $\top$ (true) or $\bot$ (false).
- $Q_a(p)$, where $a \in \Sigma$, and $Q_a(p) \leftrightarrow \sigma_p = a$
- $MOD^a_b(p)$, where $a \geq 0$, $b > 0$, and p is a position variable; $MOD^a_b(p) \leftrightarrow p \equiv_b a$
- $\phi \wedge \psi$, $\phi \vee \psi$, or $\neg\psi$, where ϕ and ψ are formulas.[11]
- $x_1 = x_2$ or $x_1 < x_2$, where x_1, x_2 are in the sort of counts.[12]
- $\exists x.\phi$ or $\forall x.\phi$, where x is a count variable and ϕ is a formula.
- $\exists^{=x} p.\phi$, where x is a count variable, p is a position variable ($\exists^{=x} p.\phi$ binds p but leaves x free), and ϕ is a formula; $\exists^{=x} p.\phi$ holds if and only if ϕ is true for exactly x values of p.

In particular, note that FOC[+;MOD] does *not* permit arithmetic operations (addition or multiplication) or comparisons ($=, <$) of position variables, only of count variables. This is the primary reason for much of the machinery introduced in the proof of Theorem 1 (Appendix A.3).

[11]We can derive $\phi \rightarrow \psi$ and $\phi \leftrightarrow \psi$ as $\psi \vee \neg\phi$ and $\phi \rightarrow \psi \wedge \psi \rightarrow \phi$, respectively.

[12]We can derive $x_1 \leq x_2$ as $x_1 = x_2 \vee x_1 < x_2$, $x_1 > x_2$ as $x_2 < x_1$, $x_1 \geq x_2$ as $x_2 \leq x_1$, and $x_1 \neq x_2$ as $\neg(x_1 = x_2)$.

A.2 Notation

We now introduce additional notation employed in the proof of Theorem 1 (Section A.3):

- $\sigma \mathbin{||} \sigma'$: denotes the concatenation of the strings σ and σ'. Note that when convenient (and unambiguous), we omit the operator and write $\sigma\sigma'$ to denote $\sigma \mathbin{||} \sigma'$.
- $\overset{n}{\underset{i=k}{||}} (\ldots)$: denotes iterated string concatenation.
- $|\sigma|$: unless otherwise specified, denotes the length of the string σ.
- σ_i: denotes the i^{th} character of the string σ.
- $\Sigma^* = \bigcup_{i=1}^{\infty} \Sigma^i$: denotes the set of all *non-empty* strings over the alphabet Σ. Note that unless otherwise specified, we slightly abuse notation and let A^* (for any $A \subseteq \Sigma^*$) denote the set of "flattened" strings of A—i.e. $A^* = \bigcup_{i=1}^{\infty} \bigcup_{a \in A^i} \{\overset{|a|}{\underset{k=1}{||}} a_k\}$ so that for all $a' \in A^*$, $a' \in \Sigma^*$.
- ϵ: denotes the empty string.
- $\sigma_{i:j} = \overset{j}{\underset{k=i}{||}} \sigma_k$: denotes the substring spanning the i^{th} to j^{th} characters (inclusive) of σ; if $i = j$, then $\sigma_{i:j} = \sigma_i$. For all $1 \leq i \leq |\sigma|$, $j > |\sigma|$: $\sigma_{i:j} = \sigma_{i:|\sigma|}$. For all $i < 1$, $j \geq 1$: $\sigma_{i:j} = \sigma_{1:j}$. If $j < 1$, $i > |\sigma|$, and/or $i > j$, then $\sigma_{i:j} = \epsilon$.
- $\sigma_{i:}$, $\sigma_{:j}$: denote $\sigma_{i:|\sigma|}$ and $\sigma_{1:j}$, respectively.
- $\sigma^n = \overset{n}{\underset{i=1}{||}} \sigma$: denotes the string σ repeated n times ($\sigma^0 = \epsilon$).
- $\phi[x \Rightarrow y] = \lambda x.[\phi](y)$: denotes the formula obtained from ϕ by replacing all instances of the free variable x with the variable (or constant) y.
- $[\phi](\sigma) = \sigma \models \phi$: indicates that the formula ϕ holds for the string σ (i.e. σ belongs to the language defined by ϕ).

A.3 Proof

Let $\Lambda = \{\mathcal{E}, \mathcal{N}, \mathcal{C}\}$ denote the set of NLI labels and let Σ' denote the input alphabet of (i.e. set of tokens for) the transformer T—we assume without loss of generality that Λ and Σ' are disjoint (i.e. $\Lambda \cap \Sigma' = \emptyset$); Theorem 1 applies only to *encoder* transformers, so we need not consider the labeling approach taken by encoder-decoder or decoder-only transformers.

By Chiang et al. (2023) Theorem 2, T corresponds to the FOC[+;MOD] formula S_T defined in Equation 1. To be explicit: Chiang et al. (2023) Theorem 2 guarantees that there exists some FOC[+;MOD] formula S_T that defines the language recognized by T. For each $(P_k, H_k, L_k) \in D$, the input to S_T is the string $P_k H_k L_k$: for all $x \in \Lambda$, $[S_T](P_k H_k L_k)$ holds if and only if the transformer T assigns the label L_k to (P_k, H_k).

$$S_T = \bigwedge_{x \in \Lambda} \phi_x \leftrightarrow \exists^{=1} p.Q_x(p) \tag{1}$$

Note that we may assume the existence of $\phi_\mathcal{E}$, $\phi_\mathcal{N}$, and $\phi_\mathcal{C}$ as in Equation 1 without loss of generality. Regardless of the approach that the particular transformer T takes to predicting labels, the output of T with respect to an input $\sigma \in (\Sigma')^*$ ($\mathcal{O}_T(\sigma)$) must be an element of Λ. As such, for each $x \in \Lambda$ and $\sigma \in (\Sigma')^*$, $[\phi_x](\sigma) =_{def} \mathcal{O}_T(\sigma) = x$.

Let $\Sigma = \Sigma' \cup \{\Omega\}$ (where Ω is a special padding character introduced for formal reasons and distinct from the actual padding character used by the transformer T), and for any $\sigma \in (\Sigma')^*$, define $f(\sigma) \in \Sigma^*$ as follows in Equation 2 (where w is the fixed input length specified in Theorem 1).

$$f(\sigma) = \mathop{||}_{i=1}^{|\sigma|+1} \mathop{||}_{k=i}^{w} (\Omega^{k-1} \mid\mid \sigma_{k:} \mid\mid \Omega^{w-|\sigma|}) \tag{2}$$

For all (integer) count terms $1 \leq b \leq w$, define $MODC_b(a, x)$ (where a, x are count variables) as follows (Equation 3)

$$MODC_b(a, x) = \bigvee_{y=0}^{w} yb + a = x \tag{3}$$

Note that by Chiang et al. (2023) Theorem 1, we may assume without loss of generality that each ϕ_x in Equation 1 is in normal form (for some integer $k \geq 0$), as in Equation 4.

$$\phi_x = \exists z_1 \ldots \exists z_k [\bigwedge_{i=1}^{k} \exists^{=z_i} p.(\phi_x)_i \wedge \chi] \tag{4}$$

Where each $(\phi_x)_i$ is quantifier-free and has no free count variables, and χ is quantifier-free.

Now, for each $x \in \Lambda$, construct $\alpha((\phi_x)_i)$ as follows: for each $a \in \Sigma'$, replace $Q_a(p)$ with $Q'_a(p)$ as defined in Equation 5 (where p is a position variable in the former, and a count variable in the latter), and replace each instance of a modular predicate $MOD^x_y(p)$ with $MODC_y(x, p)$ (where again p is a position variable in the former, and a count variable in the latter).

$$Q'_a(p) = \exists^{=p} p'[Q_a(p') \wedge \bigvee_{i=1}^{w} (MOD^i_w(p') \wedge p = i)] \tag{5}$$

Lemma 1. *For any $\sigma \in (\Sigma')^*$ such that $|\sigma| \leq w$, all $a \in \Sigma'$, and all $1 \leq p \leq w$: $[Q'_a(p)](f(\sigma)) \leftrightarrow [Q_a(p)](\sigma)$*

Proof. First, assume $[Q_a(p)](\sigma)$ holds. By assumption, $\sigma_p = a$, so by construction (Equation 2), $f(\sigma)_{yp} = a$ for all $1 \leq y \leq p$ and $f(\sigma)_{y'p} = \Omega$ for all $y' > p$. Therefore $[Q_a(p)](\sigma) \rightarrow [Q'_a(p)](f(\sigma))$.

Now, assume $[Q'_a(p)](f(\sigma))$ holds. By assumption and construction (Equation 2), $f(\sigma)_{yp} = a$ for all $1 \leq y \leq p$, so in particular $f(\sigma)_p = a$. By construction, $f(\sigma)_{:|\sigma|} = \sigma$. This implies that $\sigma_p = a$; therefore $[Q'_a(p)](f(\sigma)) \rightarrow [Q_a(p)](\sigma)$. □

Now, for any count variables p, z and any FOC[+;MOD] formula ϕ, define $E(p, z, \phi)$ as follows (Equation 6).

$$E^i_1(p, \phi) = \bigwedge_{j=1}^{i} \phi[p \Rightarrow m_j] \wedge m_j \leq w \tag{6a}$$

$$E^i_2 = \bigwedge_{a=1}^{i-1} \bigwedge_{b=a+1}^{i} m_a \neq m_b \tag{6b}$$

$$E^{i+2}_3(p, \phi) = \exists m_1 \ldots m_i [E^{i+2}_1(p, \phi) \wedge E^{i+2}_2] \tag{6c}$$

$$E^1_3(p, \phi) = \exists m_1 . E^1_1(p, \phi) \tag{6d}$$

$$E^0_3(p, \phi) = \top \tag{6e}$$

$$E(p, z, \phi) = \bigvee_{i=0}^{w} (E^i_3(p, \phi) \wedge z = i) \tag{6f}$$

Where $E^i_1(-,-)$, E^i_2, and $E^i_3(-,-)$ are defined for all integers $1 \leq i \leq w$, $2 \leq i \leq w$, and $0 \leq i \leq w$, respectively.

Now, for each $(\phi_x)_i$ in Equation 4, define $A((\phi_x)_i)$ as in Equation 7 (where z_i and p are free count variables).

$$A_1((\phi_x)_i) = E(p, z_i, \alpha((\phi_x)_i)) \tag{7a}$$
$$A_2((\phi_x)_i) = \neg\exists y[y > z_i \wedge E(p, y, \alpha((\phi_x)_i))] \tag{7b}$$
$$A((\phi_x)_i) = A_1((\phi_x)_i) \wedge A_2((\phi_x)_i) \tag{7c}$$

Lemma 2. *For any $\sigma \in (\Sigma')^*$ such that $|\sigma| \leq w$, all $x \in \Lambda$, and all $(\phi_x)_i$ as in Equation 4: $[\exists z_i \exists^{=z_i} p.(\phi_x)_i](\sigma) \leftrightarrow [\exists z_i.A((\phi_x)_i)](f(\sigma))$*

Proof. First, note that $(\phi_x)_i$ is quantifier-free and has no free count variables (Chiang et al., 2023, Theorem 1); therefore $(\phi_x)_i$ consists only of positional $(Q_a(p))$ and modular $(MOD^x_y(p))$ predicates (where the only bound variable is p) and logical operators acting on them. $A((\phi_x)_i)$ is constructed from $(\phi_x)_i$ by replacing each instance of $Q_a(p)$ and $MOD^x_y(p)$ with $Q'_a(p)$ and $MODC_y(x,p)$ (where p is a position variable in the first pair of terms, and a count variable in the second), respectively.

By Lemma 1, $[Q_a(p)](\sigma) \leftrightarrow [Q'_a(p)](f(\sigma))$ for all $1 \leq p \leq w$, where p is a position variable in the left-hand side of the equation and a count variable in the right-hand side. Similarly, for all p, x and all $1 \leq y \leq w$, $MOD^x_y(p) \leftrightarrow MODC_y(x,p)$ by construction (Equation 3), where again p is a position variable in the left-hand side of the equation and a count variable in the right-hand side.

Therefore, for all $1 \leq p \leq w$, $(\phi_x)_i$ holds with respect to σ if and only if $\alpha((\phi_x)_i)$ holds with respect to $f(\sigma)$.

By construction (Equation 6), $E(p, z, \phi)$ holds for any predicate ϕ with the count variable p free if and only if there are $\geq z$ unique values of p such that ϕ holds. By definition (Equation 7), $A((\phi_x)_i)$ holds if and only if there are exactly z_i values of p such that $\alpha((\phi_x)_i)$ holds. □

Now, for each ϕ_x in Equation 1, we define $A(\phi_x)$ as in Equation 8.

$$A(\phi_x) = \exists z_1 \ldots \exists z_k [\bigwedge_{i=1}^{k} A((\phi_x)_i) \wedge \chi] \tag{8}$$

Lemma 3. *For all $x \in \Lambda$ and all $\sigma \in (\Sigma')^*$ such that $|\sigma| \leq w$: $[\phi_x](\sigma) \leftrightarrow [A(\phi_x)](f(\sigma))$*

Proof. By Lemma 2, each $A((\phi_x)_i)$ of Equation 8 holds for $f(\sigma)$ if and only if each $(\phi_x)_i$ holds for σ. As such, for each bound count variable z_i, the set (of cardinality z_i) of values that make $A((\phi_x)_i)$ true with respect to $f(\sigma)$ is identical to the set of values that make $(\phi_x)_i$ true with respect to σ. The predicate χ contains no position variables (Chiang et al., 2023, Theorem 1), and is defined identically in Equation 8 as in Equation 4; therefore, χ (within $A(\phi_x)$) holds for $f(\sigma)$ if and only if χ (within ϕ_x) holds for σ. □

Now, for each external negation prefix $\eta \in N$, define $\psi_\eta(i)$ and $\psi'_\eta(i,j)$ (where i and j are count variables) as in Equation 9, where $Q'_{(-)}(-)$ is defined as in Equation 5.

$$\psi_\eta(i) = \bigwedge_{k=0}^{|\eta|-1} Q'_{\eta_k}(i+k) \tag{9a}$$
$$\psi'_\eta(i,j) = \psi_\eta(i) \wedge i + |\eta| - 1 = j \tag{9b}$$

Then define $\psi(i)$ and $\psi'(i,j)$ (where i and j are count variables) as in Equation 10.

$$\psi(i) = \bigvee_{\eta \in N} \psi_\eta(i) \tag{10a}$$
$$\psi'(i,j) = \bigvee_{\eta \in N} \psi'_\eta(i,j) \tag{10b}$$

Now define $\rho(i,j)$ (where i and j are count variables) as in Equation 11.

$$\rho_1(k,a,b,i,j) = i \leq a \leq k \wedge k \leq b \leq j \wedge \psi'(a,b) \tag{11a}$$
$$\rho(i,j) = \forall k[i \leq k \leq j \rightarrow \exists a, b.\rho_1(k,a,b,i,j)] \tag{11b}$$

Lemma 4. *For any $\sigma \in (\Sigma')^*$ such that $|\sigma| \leq w$, and all $1 \leq i < j \leq w$: $[\rho(i,j)](f(\sigma)) \leftrightarrow \sigma_{i:j} \in N^*$ (i.e. if and only if the span $i \rightarrow j$ in σ is a sequence of one or more external negation prefixes).*

Proof. We first prove the right-to-left direction: $\sigma_{i:j} \in N^* \rightarrow [\rho(i,j)](f(\sigma))$. The proof proceeds by induction. First, assume that σ is a single external negation prefix (i.e. $\sigma_{i:j} \in N$). Then by assumption and definition (Equation 9), $\psi'_{\sigma_{i:j}}(i,j)$ holds; by definition (Equation 10), this implies $\psi'(i,j)$. For all $i \leq k \leq j$, let $a = i$, $b = j$:

by definition (Equation 11), $\rho_1(k, a, b, i, j)$ holds. This implies $\rho(i, j)$. This proves the base case.

Now suppose $\sigma_{i:j} = \eta \,||\, \eta'$, with $\eta \in N^*$ and $\eta' \in N$. By the inductive hypothesis, $\rho(i, i + |\eta| - 1)$ holds. By the base case above, $\rho(i + |\eta|, j)$ holds. It now remains to prove that $\rho(i, i + |\eta| - 1) \wedge \rho(i + |\eta|, j) \rightarrow \rho(i, j)$. For all $1 \leq k \leq j$, if $k < i + |\eta|$, then there exist $a, b < i + |\eta|$ such that $\rho_1(k, a, b, i, j)$ (by the validity of $\rho(i, i + |\eta| - 1)$), and if $k \geq i + |\eta|$, there exist $a, b \geq i + |\eta|$ such that $\rho_1(k, a, b, i, j)$ (by the validity of $\rho(i + |\eta|, j)$); therefore, $\rho(i, j)$. This proves the induction step.

We now prove the right-to-left direction by contradiction: assume $\rho(i, j)$ and $\sigma_{i:j} \notin N^*$. By assumption, there exists $\eta \in N^* \cup \{\epsilon\}$ such that η is a substring of $\sigma_{i:j}$. For all $i \leq k \leq j$ such that σ_k is not contained within η: $\neg\exists a, b.\rho_1(k, a, b, i, j)$, by the assumption that external negation prefixes do not overlap (see Theorem 1). Therefore, $\rho(i, j)$ does not hold—this is a contradiction. □

Now define $\rho'(i, j)$ as in Equation 12.

$$\rho'_1(a, b, i, j) = (a \leq i \wedge b > j) \vee (a < i \wedge b \geq j) \tag{12a}$$

$$\rho'_2(a, b, i, j) = a > 1 \wedge \rho'_1(a, b, i, j) \tag{12b}$$

$$\rho'(i, j) = \rho(i, j) \wedge \neg\exists a, b[\rho'_2(a, b, i, j) \wedge \rho(a, b)] \tag{12c}$$

For all $x \in \Lambda$, define $F_1(x)$ as in Equation 13.

$$F_1(x) = \neg\exists i, j[j > i > 1 \wedge \rho'(i, j)] \wedge A(\phi_x) \tag{13}$$

$F_1(x)$ is intended to coincide with ϕ_x on any $(P_k, H_k, L_k) \in D$ (i.e. where the hypothesis is *not* externally negated). The term $j > i > 1$ in Equation 13 allows for the possibility that the premise P_k may be externally negated in the original dataset D.

Lemma 5. *For all $x \in \Lambda$ and all $\sigma \in (\Sigma')^*$ such that $|\sigma| \leq w$ and there does not exist $\eta \in N^*$ such that η is a subsequence of $\sigma_{2:}$: $[\phi_x](\sigma) \leftrightarrow [F_1(x)](f(\sigma))$*

Proof. By Lemma 3, $[\phi_x](\sigma) \leftrightarrow [A(\phi_x)](f(\sigma))$. By assumption, $\neg\exists i, j[j > i > 1 \wedge \rho'(i, j)]$ holds for all such $f(\sigma)$. □

We then define $A'(\phi_x)$ by replacing each predicate $Q'_a(z)$ in $A(\phi_x)$ (Equation 8) with $\beta(Q'_a(z))$, as defined in Equation 14 (where i and j are free count variables in $A'(\phi_x)$).

$$\beta_1(Q'_a(z)) = z < i \wedge Q'_a(z) \tag{14a}$$

$$\beta_2(Q'_a(z)) = z \geq i \wedge Q'_a(z + (j - i) + 1) \tag{14b}$$

$$\beta(Q'_a(z)) = \beta_1(Q'_a(z)) \vee \beta_2(Q'_a(z)) \tag{14c}$$

Lemma 6. *For all $(P_k, H_k, L_k) \in D$, all $x \in \Lambda$, and all $\eta \in N^*$ such that $|P_k \eta H_k| \leq w$: $[\phi_x](P_k H_k) \leftrightarrow [A'(\phi_x)](f(P_k \eta H_k))$ when the free variables $i = |P_k| + 1$, $j = |P_k \eta|$ in Equation 14.*

Proof. We first prove that $[A(\phi_x)](f(P_k H_k)) \leftrightarrow [A'(\phi_x)](f(P_k \eta H_k))$. Note that $A'(\phi_x)$ is constructed from $A(\phi_x)$ by replacing each instance of $Q'_a(p)$ with $\beta(Q'_a(p))$. It therefore suffices to prove that for all $a \in \Sigma'$ and all $1 \leq z \leq w$: $[Q'_a(z)](f(P_k H_k)) \leftrightarrow [\beta(Q'_a(z))](f(P_k \eta H_k))$.

If $z \leq |P_k|$, then $[Q'_a(z)](f(P_k H_k)) \leftrightarrow [\beta(Q'_a(z))](f(P_k \eta H_k))$ by definition (Equation 14). Otherwise, $[Q'_a(z)](f(P_k H_k)) \leftrightarrow [\beta(Q'_a(z))](f(P_k \eta H_k))$ if and only if $(P_k H_k)_z = (P_k \eta H_k)_{z+(j-i)+1}$. By assumption, $z + (j - i) + 1 = z + (|P_k \eta| - (|P_k| + 1)) + 1 = z + |\eta|$ and $(P_k H_k)_z = (P_k \eta H_k)_{z+|\eta|}$.

By Lemma 3 and the above result, we have: $[\phi_x](P_i H_i) \leftrightarrow [A(\phi_x)](f(P_i H_i)) \leftrightarrow [A'(\phi_x)](f(P_i \eta H_i))$. □

Now, define $F_2(x)$ as in Equation 15, where $G(\mathcal{E}) = \mathcal{C}$, $G(\mathcal{C}) = \mathcal{E}$, and $G(\mathcal{N}) = \mathcal{N}$.

$$\gamma_x^1(n) = MODC_2(1, n) \wedge A'(\phi_{G(x)}) \tag{15a}$$

$$\gamma_x^2(n) = MODC_2(0, n) \wedge A'(\phi_x) \tag{15b}$$

$$\gamma_x^3(k) = i \leq k \leq j \wedge \psi(k) \tag{15c}$$

$$\gamma_x^4(n) = E(k, n, \gamma_x^3(k)) \tag{15d}$$

$$\gamma_x^5(n) = \neg\exists y[y > n \wedge E(k', y, \gamma_x^3(k'))] \tag{15e}$$

$$\gamma_x = \exists n[\gamma_x^4(n) \wedge \gamma_x^5(n) \wedge (\gamma_x^1(n) \vee \gamma_x^2(n))] \tag{15f}$$

$$F_2(x) = \exists i, j[j > i > 1 \wedge \rho'(i, j) \wedge \gamma_x] \tag{15g}$$

Lemma 7. *Define $N_0, N_1 \subset N^*$ as the sets of even- and odd-length (in terms of number of prefixes, rather than characters) sequences of external negation prefixes, respectively. Then for all $x \in \Lambda$ and all $(P_k, H_k, L_k) \in D$:*

i. for all $\eta \in N_0$: $[\phi_x](P_k H_k) \leftrightarrow [F_2(x)]$

ii. for all $\eta' \in N_1$*:* $[\phi_{G(x)}](P_k H_k) \leftrightarrow [F_2(x)](f(P_k \eta' H_k))$

Proof. We first prove (i). By Lemma 4 and the definition of $\rho'(i,j)$ (Equation 12), the respective values of i, j that make the term $j > i > 1 \wedge \rho'(i,j)$ hold in Equation 15 are $i = |P_k| + 1$ and $j = |P_k \eta|$. By the definitions of $E(k,n,-)$, $\psi(-)$, and γ_x (Equations 6, 10, and 15, respectively)—and the assumption that $\eta \in N_0$—the value of n that makes $[\gamma_x](f(P_k \eta H_k))$ hold is even. Therefore, the term $MODC_2(0,n)$ in $\gamma_x^2(n)$ holds, and so $[A'(\phi_x)](f(P_k \eta H_k)) \leftrightarrow [F_2(x)](f(P_k \eta H_k))$.

By Lemma 6 and the above result: $[\phi_x](P_k H_k) \leftrightarrow [A'(\phi_x)](f(P_k \eta H_k)) \leftrightarrow [F_2(x)](f(P_k \eta H_k))$.

We now prove (ii); the proof proceeds in a similar fashion as that of (i) above. But now n is odd, and so the term $MODC_2(1,n)$ in $\gamma_x^1(n)$ holds. Therefore, $[A'(\phi_{G(x)})](f(P_k \eta' H_k)) \leftrightarrow [F_2(x)](f(P_k \eta' H_k))$.

Again by Lemma 6 and the above result: $[\phi_{G(x)}](P_k H_k) \leftrightarrow [A'(\phi_{G(x)})](f(P_k \eta H_k)) \leftrightarrow [F_2(x)](f(P_k \eta H_k))$. □

For all $x \in \Lambda$, we define $F(x)$ as follows (Equation 16).

$$F(x) = F_1(x) \vee F_2(x) \tag{16}$$

We may now define the formula $S_{T'}$ in Equation 17 below.

$$S_{T'} = \bigwedge_{x \in \Lambda} F(x) \leftrightarrow \exists^{=1} p.Q_x(p) \tag{17}$$

Lemma 8. *For all* $(P_k, H_k, L_k) \in D$*, all* $\eta \in N_0$ *such that* $|P_k \eta H_k| \leq w$*, and all* $\eta' \in N_1$ *such that* $|P_k \eta' H_k| \leq w$*:*

i. $[S_{T'}](f(P_k H_k) L_k) \leftrightarrow [S_T](P_k H_k L_k)$

ii. $[S_{T'}](f(P_k \eta H_k) L_k) \leftrightarrow [S_T](P_k H_k L_k)$

iii. $[S_{T'}](f(P_k \eta' H_k) G(L_k)) \leftrightarrow [S_T](P_k H_k L_k)$

Proof. By Lemma 5, $[F_1(L_k)](f(P_k H_k))$ holds if and only if $[\phi_{L_k}](P_k H_k)$ does as well, for all $(P_k, H_k, L_k) \in D$. $F_2(x)$ does not hold for any $x \in \Lambda$ by definition, and $[F_1(x)](f(P_k H_k)) \leftrightarrow [\phi_x](P_k H_k)$ for any $x \in \Lambda - \{L_k\}$ by Lemma 5. This proves (i).

For all $\eta \in N_0$ such that $|P_k \eta H_k| \leq w$, $[F_1(x)](f(P_k H_k))$ does not hold for any $x \in \Lambda$ by definition, and $[F_2(x)](f(P_k H_k)) \leftrightarrow [\phi_x](P_k H_k)$ for all $x \in \Lambda$ by Lemma 7(i). This proves (ii).

For all $\eta' \in N_1$ such that $|P_k \eta' H_k| \leq w$, $[F_1(x)](f(P_k H_k))$ does not hold for any $x \in \Lambda$ by definition, and $[F_2(x)](f(P_k H_k)) \leftrightarrow [\phi_{G(x)}](P_k H_k)$ for all $x \in \Lambda$ by Lemma 7(ii). This proves (iii). □

By Chiang et al. (2023) Theorem 5, there exists a transformer encoder T'' that recognizes the language defined by $S_{T'}$. By Lemma 8(i), $Acc(T'', f(D)) = Acc(T, D)$, and $Acc(T'', \{f(P_i \eta H_i)\}_{i \in I}) = Acc(T, D)$ for any $\eta \in N^*$ such that $max_{i \in I}|P_i \eta H_i| \leq w$ by Lemma 8(ii-iii).

But T'' is an arbitrary-precision transformer. It remains to show that we can derive a *fixed-precision* transformer T' from T''. Note that by definition (Equation 2), for any $\sigma \in (\Sigma')^*$ such that $|\sigma| < w$: $|f(\sigma)| = w(|\sigma| + 1)$. By assumption (Theorem 1), no input example (adversarial or otherwise) exceeds the fixed (finite) $w > max_{i \in I}|P_i H_i|$ in length. It follows that the upper bound on the length of possible inputs to T'' (within the assumptions of Theorem 1) is $w^2 + w$.

By definition, the floating-point precision of an arbitrary-precision transformer varies as a function of input length. Let $\pi \colon \mathbb{N} \to \mathbb{N}$ be the function mapping input length to floating-point precision (in bits) of T''. Presumably, π is monotone-increasing, but it need not be: let $\ell_{max} = max_{1 \leq n \leq w^2 + w} \pi(n)$. Define T' as T'' with floating-point precision fixed at ℓ_{max}.

This completes the proof of Theorem 1.

B Experiment 3

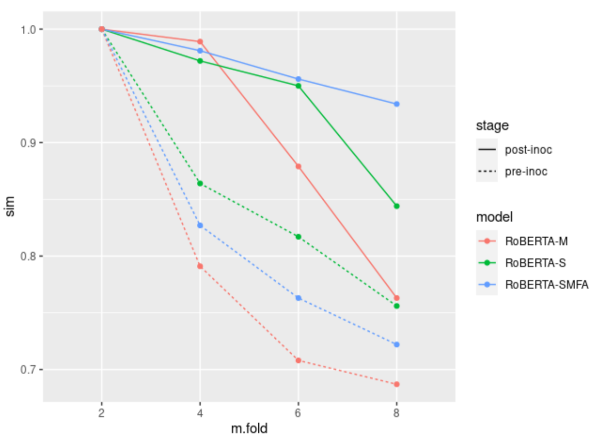

Figure 4: Mean cosine similarity between $(T_{NT})^n H_i$ and $(T_{NT})^2 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

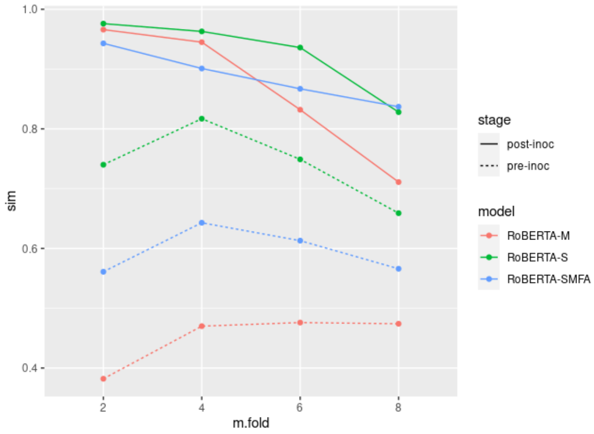

Figure 5: Mean cosine similarity between $(T_{NT})^n H_i$ and H_i for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

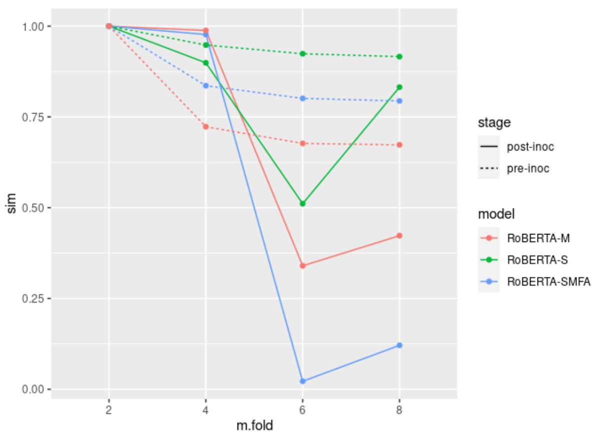

Figure 6: Mean cosine similarity between $(T_F)^n H_i$ and $(T_F)^2 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

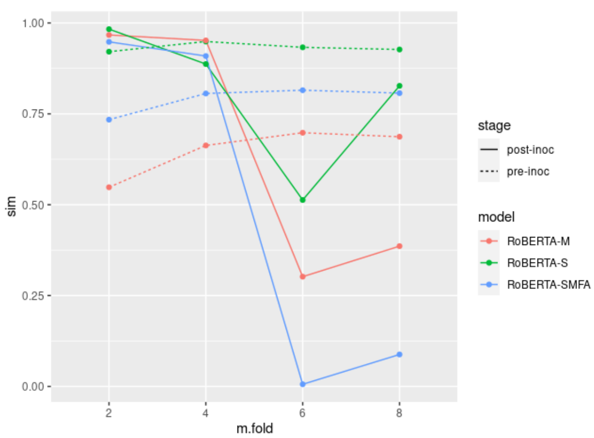

Figure 7: Mean cosine similarity between $(T_F)^n H_i$ and H_i for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

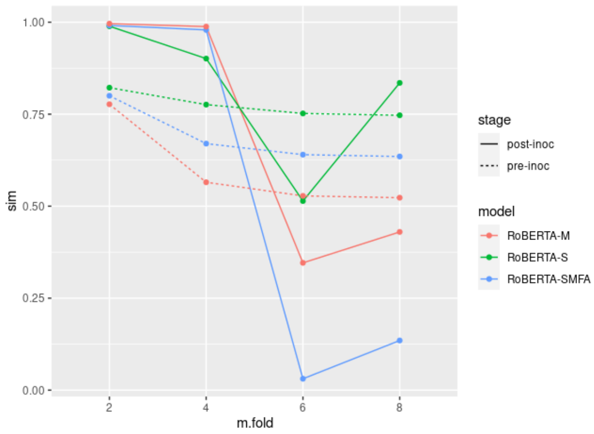

Figure 8: Mean cosine similarity between $(T_F)^n H_i$ and $(T_{NT})^2 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

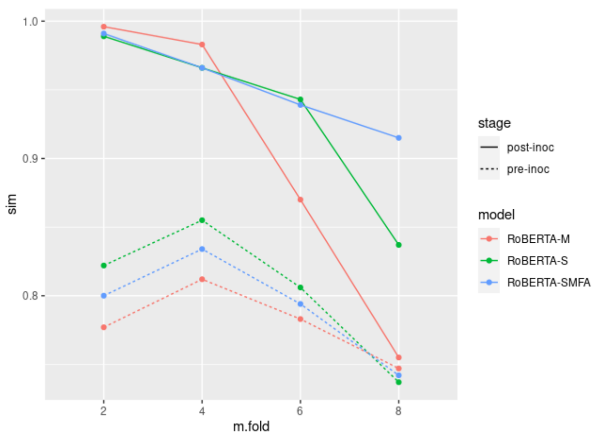

Figure 9: Mean cosine similarity between $(T_{NT})^n H_i$ and $(T_F)^2 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

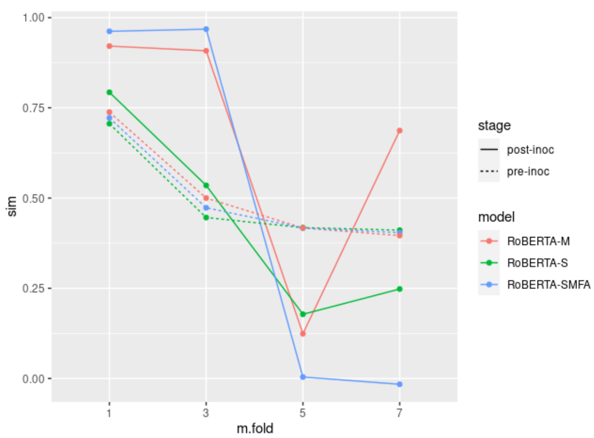

Figure 10: Mean cosine similarity between $(T_F)^n H_i$ and $(T_{NT})^1 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

1265

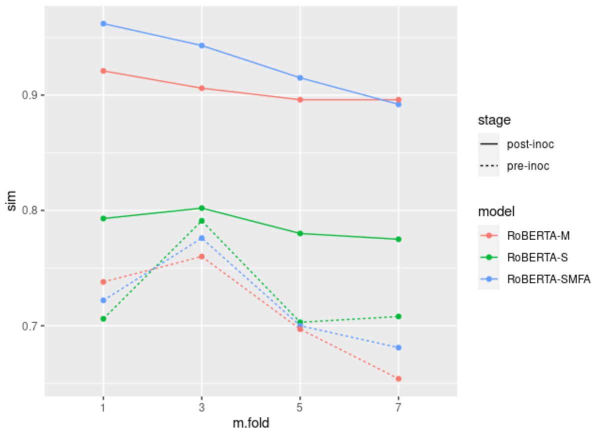

Figure 11: Mean cosine similarity between $(T_{NT})^n H_i$ and $(T_F)^1 H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

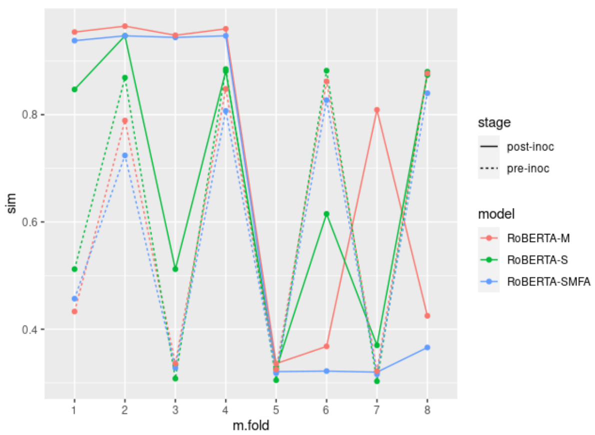

Figure 12: Mean cosine similarity between $(T_{NT})^n H_i$ and $(T_F)^n H_i$ for the three RoBERTa models before (dashed) and after (solid) ≤5-fold T_{NT} inoculation.

C Experiment 1

C.1 Inoculation Development Set Accuracies

Model	Initial Acc. (Original)	Initial Acc. (Adversarial)	Inoculated Acc. (Original)	Inoculated Acc. (Adversarial)
$BART_M$	0.89	0.52	0.77	0.94
$RoBERTa_M$	0.89	0.51	0.87	0.93
$DeBERTa_S$	0.9	0.39	0.9	0.91
$RoBERTa_S$	0.88	0.57	0.88	0.89
$BART_{SMFA}$	0.89	0.69	0.87	0.92
$RoBERTa_{SMFA}$	0.87	0.51	0.86	0.91

Table 5: Model accuracy on the original and adversarial development sets before and after 1-fold inoculation.

Model	Initial Acc. (Original)	Initial Acc. (Adversarial)	Inoculated Acc. (Original)	Inoculated Acc. (Adversarial)
$BART_M$	0.89	0.61	0.86	0.94
$RoBERTa_M$	0.89	0.63	0.87	0.97
$DeBERTa_S$	0.9	0.48	0.9	0.96
$RoBERTa_S$	0.88	0.66	0.88	0.94
$BART_{SMFA}$	0.89	0.72	0.88	0.95
$RoBERTa_{SMFA}$	0.87	0.65	0.88	0.95

Table 6: Model accuracy on the original and adversarial development sets before and after $\leq$2-fold inoculation.

Model	Initial Acc. (Original)	Initial Acc. (Adversarial)	Inoculated Acc. (Original)	Inoculated Acc. (Adversarial)
$BART_M$	0.89	0.53	0.87	0.95
$RoBERTa_M$	0.89	0.54	0.87	0.96
$DeBERTa_S$	0.9	0.45	0.9	0.96
$RoBERTa_S$	0.88	0.57	0.88	0.93
$BART_{SMFA}$	0.89	0.6	0.76	0.93
$RoBERTa_{SMFA}$	0.87	0.54	0.88	0.94

Table 7: Model accuracy on the original and adversarial development sets before and after $\leq$3-fold inoculation.

Model	Initial Acc. (Original)	Initial Acc. (Adversarial)	Inoculated Acc. (Original)	Inoculated Acc. (Adversarial)
$BART_M$	0.89	0.61	0.62	0.75
$RoBERTa_M$	0.89	0.62	0.74	0.88
$DeBERTa_S$	0.9	0.54	0.89	0.76
$RoBERTa_S$	0.88	0.64	0.89	0.89
$BART_{SMFA}$	0.89	0.66	0.62	0.86
$RoBERTa_{SMFA}$	0.87	0.61	0.88	0.89

Table 8: Model accuracy on the original and adversarial development sets before and after $\leq$4-fold inoculation.

Model	Initial Acc. (Original)	Initial Acc. (Adversarial)	Inoculated Acc. (Original)	Inoculated Acc. (Adversarial)
$BART_M$	0.89	0.55	0.32	0.74
$RoBERTa_M$	0.89	0.56	0.88	0.93
$DeBERTa_S$	0.9	0.5	0.9	0.91
$RoBERTa_S$	0.88	0.58	0.88	0.89
$BART_{SMFA}$	0.89	0.59	0.87	0.88
$RoBERTa_{SMFA}$	0.87	0.54	0.86	0.87

Table 9: Model accuracy on the original and adversarial development sets before and after $\leq$5-fold inoculation.

C.2 Post-Inoculation Test Accuracy

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.71	0.32	—	—
3	0.36	0.93	0.31	—
4	0.82	0.36	0.94	0.31
5	0.33	0.88	0.31	0.94
6	0.86	0.41	0.94	0.31

Table 10: Accuracy for $BART_M$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.77	0.36	—	—
3	0.34	0.89	0.33	—
4	0.85	0.33	0.97	0.32
5	0.32	0.88	0.33	0.95
6	0.89	0.34	0.97	0.33

Table 11: Accuracy for $RoBERTa_M$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.56	0.62	—	—
3	0.4	0.61	0.32	—
4	0.84	0.64	0.96	0.5
5	0.3	0.51	0.32	0.96
6	0.88	0.77	0.96	0.36

Table 12: Accuracy for $DeBERTa_S$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.74	0.32	—	—
3	0.4	0.89	0.3	—
4	0.84	0.35	0.94	0.39
5	0.34	0.88	0.3	0.74
6	0.83	0.33	0.93	0.53

Table 13: Accuracy for $RoBERTa_S$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.77	0.37	—	—
3	0.33	0.91	0.31	—
4	0.84	0.34	0.94	0.29
5	0.3	0.85	0.31	0.92
6	0.86	0.41	0.94	0.28

Table 14: Accuracy for $BART_{SMFA}$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).

m-fold test acc.	No inoc.	1-fold inoc.	≤2-fold inoc.	≤3-fold inoc.
2	0.79	0.35	—	—
3	0.32	0.93	0.32	—
4	0.83	0.31	0.95	0.32
5	0.32	0.94	0.32	0.94
6	0.84	0.32	0.95	0.32

Table 15: Accuracy for $RoBERTa_{SMFA}$ on (m>n)-fold external negation after ≤n-fold inoculation ($n \in \{1, 2, 3\}$).