# To Learn or Not to Learn: Replaced Token Detection for Learning the Meaning of Negation

**Gunjan Bhattarai, Katrin Erk**
University of Texas at Austin
2515 Speedway, Austin, Texas 78712, United States
gunjan.bhattarai@utexas.edu, katrin.erk@utexas.edu

**Abstract**

State-of-the-art language models perform well on a variety of language tasks, but they continue to struggle with understanding negation cues in tasks like natural language inference (NLI). Inspired by Hossain et al. (2020), who show under-representation of negation in language model pretraining datasets, we experiment with additional pretraining with negation data for which we introduce two new datasets. We also introduce a new learning strategy for negation building on ELECTRA's (Clark et al., 2020) replaced token detection objective. We find that continuing to pretrain ELECTRA-Small's discriminator leads to substantial gains on a variant of RTE (Recognizing Textual Entailment) with additional negation. On SNLI (Stanford NLI) (Bowman et al., 2015), there are no gains due to the extreme under-representation of negation in the data. Finally, on MNLI (Multi-NLI) (Williams et al., 2018), we find that performance on negation cues is primarily stymied by neutral-labeled examples.

**Keywords:** negation, inference, natural language inference, replaced token detection, sentence semantics

## 1. Introduction

With the adoption of the Transformer (Vaswani et al., 2017) and ever larger models trained on gigantic datasets (Devlin et al., 2018; Raffel et al., 2020; Brown et al., 2020; Fedus et al., 2022; Chowdhery et al., 2022; OpenAI, 2024), we have seen massive improvements in neural models' ability to master natural language understanding tasks. However, these models still struggle with some linguistic phenomena, including negation (Ettinger, 2020)–the phenomenon that we focus on in this paper. As an example of neural language models' difficulties with negation, BERT-Base (Devlin et al., 2018) outputs "Berlin" for both the inputs "[MASK] is the capital of Germany" and "[MASK] is not the capital of Germany" (despite the latter statement being factually incorrect). This problem extends to downstream tasks as well, including sentiment analysis (Tejada et al., 2021) and NLI (Hossain et al., 2020). Prior work has consistently held that negation is a fundamentally hard problem for models to learn, particularly during pretraining. Kassner and Schütze (2020) show that models are simply unable to learn negation using the masked language modeling (MLM) objective, and Hosseini et al. (2021) only achieve modest improvements on NLI with a pretraining task focused on negation. Moreover, unlike other phenomena like math or multi-step reasoning that language models have historically struggled on, scaling up models to the billions of parameters still does not solve the negation issue (García-Ferrero et al., 2023).

In this work, we investigate the degree to which additional pretraining on negated examples can shrink the gap between neural language model's performance on NLI (natural language inference) sequences with and without negation cues. Our method involves taking ELECTRA (Clark et al., 2020) and continuing pretraining on two new datasets that we call Expanded NLI and Expanded LAMA. ELECTRA's pretraining differs from that of BERT in that it also uses replaced token detection (RTD): it trains a discriminator to distinguish tokens that were in the original, unmasked parts of the sentence ("Original", represented as a 0) from those where the generator replaced [MASK] tokens ("Replaced", represented as a 1).

More generally, can RTD training teach language models about the meanings of words that MLM cannot? Gastaldi and Pellissier (2021) view language models with MLM pretraining from a structuralist point of view, in which linguistic units are characterized through co-occurrence (syntagmatic relations) and substitutability (paradigmatic relations). From that perspective, our question becomes: Can the meaning of negation be easily described in terms of syntagmatic and paradigmatic regularities? We are not aware of work arguing either for or against this, but we think it is possible that the meaning of negation is hard to pick up solely from word co-occurrence. With the RTD training regime, we can demonstrate more directly what negation "does", and this may help the model pick up the right inferences. If that proves successful, then RTD training could also be helpful for other phenomena, such as antonymy.

We use ELECTRA's RTD objective to teach the model about inferences involving negation cues as follows. We want ELECTRA to output [0 0 0 0 0 0]



*LREC-COLING 2024*, pages 16237–16250

– all original – for the sentence "Berlin is the capital of Germany" (we make the simplification here that each word is its own token) because the statement is factually correct. In contrast, for its direct negative "Berlin is not the capital of Germany", we would want the system to predict either "Berlin" ([1 0 0 0 0 0 0]), "not" ([0 0 1 0 0 0 0]), or "Germany" ([0 0 0 0 0 0 1]) as "Replaced", thereby creating a representation in which adding "not" directly changes the output.

After pretraining, we finetune our models on a collection of RTE datasets (Recognizing Textual Entailment)[1] (Dagan et al., 2006; Bar-Haim et al., 2006; Giampiccolo et al., 2007; Bentivogli et al., 2009; Wang et al., 2018), MNLI (Multi-NLI)[2] (Williams et al., 2018), and SNLI (Stanford NLI)[3] (Bowman et al., 2015) and evaluate both on their standard development sets as well as on held-out negated data from Hossain et al. (2020).

Overall, we make the following contributions. First, we demonstrate that ELECTRA's discriminator can be used to learn negation cues within NLI, finding a 19.9% increase in accuracy on our Negated RTE evaluation dataset. Second, we show that negation is not learned well in SNLI, likely because it contains very few negated examples. Finally, we provide evidence that negation is not understood in MNLI because models have difficulty learning the impact of negation cues in neutral-labeled examples. If neutral examples are excluded, even our baseline ELECTRA-Small model learns the impact of negation cues on entailment and contradiction in MNLI with relative ease.

## 2. Related Work

Hossain et al. (2020) provides evidence that neural models' dismal performance on negation in NLI is because sentences with negations are underrepresented in popular pretraining datasets. For example, just 8.69% of sentences in Wikipedia have negation cues. Many finetuning datasets suffer from the same problem - for instance, just 1.19% of SNLI (Bowman et al., 2015) examples and 7.16% of RTE with examples include negation cues. As a result, models do not get much experience with handling negation. To remedy this, Hossain et al. (2020) create 1,500 additional hand-annotated NLI examples containing "not" for each of RTE, MNLI, and SNLI, making 450 examples of each set visible during finetuning. For example, when finetuning on RTE, 450 Negated RTE examples were added to the finetuning dataset, and the model was evaluated on the 1050 held-out examples. While this was enough to bring performance on negated examples to the same accuracy as non-negated examples for RTE and reduce the non-negative/negative gap by about 50% for SNLI, MNLI (a multi-domain natural inference dataset) only saw small gains. Hossain et al. (2020) speculated that this was due to the wide diversity of the MNLI dataset, which was difficult to cover from just 450 new examples. Our work offers a different explanation–we find that weaker performance on negation in MNLI is because the model has not learned how to handle negation cues in examples labeled "neutral".

Despite the clear gains in performance found by Hossain et al. (2020), their approach has its limits since finetuning can result in catastrophic forgetting, meaning that knowledge from pretraining is lost when weights are changed to learn the finetuning task (French, 1999; McCloskey and Cohen, 1989; Kirkpatrick et al., 2017). More specifically, because the model is only learning negation in the finetuning stage (i.e., when it loses knowledge not specific to learning NLI), the knowledge it picks up cannot be transferred to other downstream tasks.

Hosseini et al. (2021) propose resolving this by further pretraining BERT with an unlikelihood loss on factually incorrect statements with negation cues. For instance, for the training example "[MASK] is not the capital of Germany," the aim would be to minimize the probability of BERT predicting "Berlin." They combine this continued pretraining approach with knowledge distillation on non-negated examples from the original BERT to preserve performance on examples without negation. Our work borrows heavily from their training regimen, except that we use ELECTRA's replaced token detection (RTD) objective for greater flexibility in adding new pretraining examples.

While their approach leads to tangible gains (albeit much smaller than Hossain et al. (2020)'s gains from finetuning data augmentation) when finetuning RTE, unlikelihood pretraining only increases accuracy by 1.5 points on SNLI and performs within the margin of error for MNLI. Hosseini et al. (2021) hypothesize that this is due to catastrophic forgetting. Our work largely replicates their findings, though we find different causes for the results on MNLI and SNLI. For SNLI, we find that models immediately overfit on non-negated cues due to severe underrepresentation of negated examples. For MNLI, we find that the neutral-labeled examples confuse the model; when choosing between just entailment and contradiction, ELECTRA-Small already performs well on MNLI.

[1] RTE-1, RTE-2, RTE-3, and RTE-5, the datasets use, are all freely available without restriction.

[2] Most of MNLI is available under the OANC license. The remainder is available under the Creative Commons share-Alike 3.0 Unported License, Creative Commons Attribution 3.0 Unported Licenses, and the public domain in the USA.

[3] SNLI is released under the Creative Commons Share-Alike Unported License.

Kassner and Schütze (2020) find for multiple pretrained language models that they are not well suited to making factual predictions in the presence of negation. To demonstrate this, they compare the models' predictions between the LAMA (LAnguage Model Analysis) (Petroni et al., 2019) and Negated LAMA (Kassner and Schütze, 2020) datasets[4]. LAMA consists of factual statements, such as "As early as the 14th century there was domestic hard-coal processing" where a word piece is masked to test for factual knowledge. Negated LAMA consists of sentence pairs where the first sentence states a fact and the second sentence repeats that statement, but adds "not" to falsify it. The token that is masked is the one that would need to change to make the sentence correct. One example from the dataset is "The waters of the river Brahmaputra are shared by China, India, and Bangladesh. The [MASK] of the River Brahmaputra are **not** shared by China, India, and Bangladesh." In this case, [MASK] can be filled by the word "delta", which is only in Bangladesh. The authors find the tested language models do not respond to the presence of negation cues: they have similar predictions in both standard and Negated LAMA.

Kassner and Schütze also suggest that the masked language modeling (MLM) task itself is not well-suited to learning negation. They pretrain a BERT-Base model from scratch on a custom dataset, with 50% of examples in the form "$x_j$ is $a_n$" and 50% of examples in the form "$x_j$ is not $a_n$" (where $x$ is a subject and $a$ is an adjective). While their model performed well on both negated and non-negated examples in training, performance collapsed on the test set, showing that MLM is unable to teach generalizable negation knowledge. However, when this BERT model is finetuned on factuality classification (i.e., determining whether a given statement is true or false), both training and test accuracy reach 100%. Our work uses RTD instead and finds that the task is sufficient to learn how to handle negation during pretraining when the negation cue flips the sentence's polarity.

García-Ferrero et al. (2023) show that negation remains a problem even when scaling language models to the billions of parameters. They showcase this by taking a dataset built using WordNet to do factuality classification (e.g., predict either true or false for a given sentence) on both negated and non-negated examples. They find that foundation models ranging from T5-XXL (Raffel et al., 2020) to Falcon-40B (Almazrouei et al., 2023) to LLaMA-65B (Touvron et al., 2023) struggle to classify negative sentences, with performance worse than random guessing. While instruction-tuned LLMs like Flan-T5 (Chung et al., 2022) and Vicuña (Chiang et al., 2023) (i.e., those that are fine-tuned on data that includes negated cues, including sentiment analysis) are able to classify negated examples accurately if they don't have a distractor, they struggle to do so if a distractor is present. Thus, we get the same picture in the world of large language models as both Kassner and Schütze (2020) and Hossain et al. (2020) find with BERT: Pretraining alone does not give sufficient information on how to work with negation, thought fine-tuning helps.

## 3. Methods

We create two datasets to further train ELECTRA based on existing NLI and LAMA data. While NLI prediction is normally a sequence classification task, we convert it to a task where ELECTRA classifies each token as "original" or "replaced".

### 3.1. Pretraining on Wikipedia and Books

To our knowledge, ELECTRA has not been previously used to test negation performance in NLI, so we establish our baselines by finetuning the ELECTRA-Small[5] Discriminator on RTE, MNLI, and SNLI. For all other experiments, we continue pretraining the discriminator on the RTD task on two datasets, which we call Expanded NLI and Expanded LAMA. The two datasets comprise sentences involving negation adapted from NLI and LAMA datasets (see Sections 3.2 and 3.3). Both datasets also contain 174,808 sequences from the Books corpus and 144,496 examples from English Wikipedia[6] (all of size 128 tokens as measured by ELECTRA's tokenizer). The purpose of adding Wikipedia and Books data (the datasets ELECTRA was originally trained on) is to ensure that our models do not overfit on the much smaller set of NLI and LAMA examples we introduce.

To combat the paucity of sentences containing negation within Wikipedia, we upsample them. We use NegBERT (Khandelwal and Sawant, 2020) to identify negation cues in Wikipedia, and sample 1/3 of our examples from sequences without negation, 1/3 from sequences with 1 negation token, and 1/3 from sequences with multiple negation tokens.

Finally, to simulate ELECTRA's pretraining, we mask 15% of the tokens in our selected Wikipedia and Books sequences at random, using the open-sourced ELECTRA-Small Generator to replace the

[4]Both are licensed under the Creative Commons Attribution-Noncommercial 4.0 International License

[5]To our knowledge, only ConvBERT (Jiang et al., 2021) and DeBERTa-v3 (He et al., 2021) also trained with RTD. As they are trained on identical datasets as ELECTRA, we believe that ELECTRA's performance can be generalized to them.

[6]Licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported License (CC BY-SA) and the GNU Free Documentation License (GFDL)

tokens. Unlike in standard ELECTRA training, we don't train the generator and discriminator jointly–we only train the discriminator. This choice is designed to limit any potential gains the model may get on standard Wikipedia and Books such that it won't have an unfair advantage over the baseline ELECTRA-Discriminator model.

### 3.2. Pretraining on Expanded NLI

The new Expanded NLI dataset comprises our curated Books and Wikipedia datasets along with subsets of the negation examples Hossain et al. (2020) created for each of RTE, MNLI, and SNLI, reformatting these training examples in the ELECTRA discriminator pretraining format.

An NLI data point typically consists of a premise and a hypothesis, where the task is to predict whether the premise means that the hypothesis is true ("entailment"), whether the premise has no bearing on the hypothesis ("neutral"), or whether the hypothesis outright contradicts the premise ("contradiction"). RTE differs from MNLI and SNLI in that there is no "neutral" category. Each data point from Hossain et al. (2020) is a set of three NLI data points. Each triplet contains one NLI data point with a negation in the premise, one data point with negation in the hypothesis, and one data point with negation in both. Here is a simplified example:

**Premise:** Subia was not nominated by President Bush to the board of directors of the Legal Services Corp. **Hypothesis:** Subia was Bush's first nominee to the Board of Directors. **Label:** Contradiction

**Premise:** Subia was nominated by President Bush to the board of directors of the Legal Services Corp. **Hypothesis:** Subia was not Bush's first nominee to the Board of Directors. **Label:** Neutral

**Premise:** Subia was not nominated by President Bush to the board of directors of the Legal Services Corp. **Hypothesis:** Subia was not Bush's first nominee to the Board of Directors. **Label:** Entailment

We convert such a triplet to data that we can use to train ELECTRA's discriminator as follows. We (1) convert an NLI premise/hypothesis pair to a coherent text, and (2) use the triplets to automatically create original/replaced labels to teach ELECTRA about negation. To address (1), we take inspiration from cloze-based prompting (Schick and Schütze, 2020a,b), connecting the premise and hypothesis by one of the connecting words (discourse markers) from Table 1. Here is the contradiction from above, reformulated using *thus*: "Subia was not nominated by President Bush to the board of directors of the Legal Services Corp. **Thus**, Subia was Bush's first nominee to the Board of Directors." To avoid overfitting on sentence order, we format 50% of examples as having premise then hypothesis and the other 50% as having hypothesis then premise.

To address (2), we check each triplet to see if it contains any item labeled as "entailment". If there is exactly one, this becomes our "comparison sentence". In our example above, this is the third member of the triplet. We then rewrite the other two members of the triplet with the aim of tagging any word that is the same as in the comparison sentence as 0/"original," and any mismatched words as 1/"replaced". Roughly, the process is as follows: to assign labels of 0 and 1 to a target sequence, we track that sequence and the comparison sequence in parallel. As long as the sequences align, we mark words in the target sequence as 0/original. When we come upon a mismatched word, we attempt to look for the next possible alignment after the mismatch, labeling words as 1/replaced until we find the alignment. Below is the full procedure:

**1.** Split the NLI example into a premise and hypothesis. For each of them:

**2.** Does the input sequence have the same number of tokens as the comparison sequence?

**2a.** If yes: Check whether the $i^{th}$ token of the input sequence is equal to the $i^{th}$ token of the comparison sequence, outputting a 0 for index $i$ if they are equivalent and 1 if they are not.

**2b.** If no: We need to track two different indices - $i$ for input and $j$ for the comparison sequence. $i$ and $j$ both start as 0.

**2b1.** Is the $i^{th}$ token of the input sequence the same as the $j^{th}$ token of the comparison sequence? If yes, increment both $i$ and $j$ by 1 (i.e., move one token to the right). Keep doing this until the two tokens are found to be different.

**2b2.** If the two tokens are found to be different: Skip to the next token for whichever sequence (input or comparison) is longer. That is, increment $i$ if the input sequence is longer, and $j$ if the comparison sequence is longer. Keep doing this until the tokens are equivalent, in which case we will return back to Step 2b1 with our new values of $i$ and $j$. All input tokens that were traversed during this step are marked as a "1", except if both (1) the comparison sequence is being traversed and a match is found between input and comparison and (2) the previous token in the input sequence was tagged as a "1". In this case, since no tokens were traversed in the input sequence, the token would be tagged as a "0".

**2b3.** If no equivalent pair of tokens is found after moving k steps to the right (where k is a parameter; we chose 4 to mitigate the risk of finding two identical words in very different parts of the sentence), tag the current token as "1" on the input sequence and add 1 to both $i$ and $j$ (i.e., move one token to the right for both sequences) from where they were when you last began step 2b2. Try Step 2b2 once more.

**3.** Connect the converted premise and hypothesis with a vector of 0's matching the number of tokens in the connecting phrase.

As an example[7], we consider the first member of the triplet above as the target sequence. The first mismatch is at the word "Bush's" in the hypothesis part of the target sequence, which does not match the word "not" in the comparison (entailment) sequence. To find a match, we traverse "not Bush's" in the comparison sequence, and "Bush's" in the target sequence, so we mark "Bush's" in the target sequence as 1/replaced. Here is the reformulated first triplet member, where the bolded word is marked replaced:

"Subia was not nominated by President Bush to the board of directors of the Legal Services Corp. Thus, Subia was **Bush's** first nominee to the Board of Directors."

Overall, our procedure shares similarity with both traditional RTD-based matching as well as token-level edit distance, though in the case of the latter it differs in two crucial aspects. First, token-level edit distance represents a sum, whereas we return a vector of values with 1s and 0s denoting "replaced" or "original" for each token. Second, our algorithm also has to handle the placement of those 1s and 0s, which is important since unlike in standard RTD, the two sequences being compared do not have the same length.

If a triplet contains no entailed sentences, then we discard it from pretraining. When there is more than one entailed sequence, we compare each entailed sequence to itself. For all other members of the triplet, we randomly select one of the possible entailed sequences to compare the example to. We also remove neutral examples from Hossain et al. (2020)'s MNLI and SNLI datasets.

Finally, to avoid the situation in which the model would learn to overfit on the negation of our NLI-like examples, we create non-negative versions of the premise/hypothesis pairs, pseudo-label them with pretrained models on each dataset available on Huggingface (textattack/roberta-base-RTE for RTE, microsoft/deberta-base-mnli for MNLI, and boychaboy/SNLI_distilroberta-base for SNLI) (Wolf et al., 2019) and use the same methods outlined above to add them to our training set.

Ultimately, we add 600 RTE-style examples (450 negated, 150 non-negated examples), 447 MNLI-style examples (384 negative, 63 non-negated), and 401 SNLI-style examples (348 negated, 63 non-negated). We add all unused negated examples to our evaluation sets used to test our models after finetuning.

[7]To see more examples of this procedure, please reference Appendix A.

| Sentence 1 | Connecting Words |
|---|---|
| Premise | therefore; as a result; thus; we can thus conclude that; based on this, we can assume that; from this, we can deduce that; this means that; hence; accordingly; ergo |
| Hypothesis | this is because; rather; we know this because; in reality; in fact, we know this since; |

Table 1: Discourse markers used to connect Hossain et al. (2020)'s examples for pretraining. "Sentence 1" refers to whether the premise or the hypothesis was chosen to be the first sentence.

Non-negated Sentence

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| The | waters | of | the | river |

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| Brahmaputra | are | shared | by. |

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| China, | India, | and | Bangladesh. |

Negated Sentence

| | | | | |
|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 |
| The | waters | of | the | river |

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| Brahmaputra | are | not | shared | by |

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| China, | India, | and | Bangladesh. |

Table 2: An example of a negated and non-negated sentence used to create Expanded LAMA. Variant 1 combines both sentences into one data point; Variant 2 only uses the non-negated sentence; Variant 3 only uses the negated sentence; and Variant 4 uses the negated sentence twice.

| 0 | 1 | 0 | 0 | 0 |
|---|---|---|---|---|
| The | sketch | of | the | river |
| 0 | | 0 | 0 | 0 |
| Brahmaputra | | are | shared | by. |
| 0 | 0 | 0 | 0 | |
| China, | India, | and | Bangladesh. | |

Table 3: Using the same sentence pair as Table 2, we show an example for Variant 2.

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| As | early | as | the |
| 0 | 0 | 0 | 0 |
| 14th | century | there | was |
| 0 | 0 | 0 | 0 |
| domestic | hard | -coal | processing. |

Table 4: An example of a sentence pair in Expanded LAMA constructed from Standard LAMA. Variant 1 uses the sentence as-is, while Variant 2 duplicates it.

### 3.3. Pretraining on Expanded LAMA

Our Expanded LAMA dataset consists of our curated Books and Wikipedia datasets plus preprocessed combinations of Standard (Petroni et al., 2019) and Negated LAMA (Kassner and Schütze, 2020) corpora. Because Kassner and Schütze (2020) originally created Negated LAMA for testing masked language models, they did not create gold labels for the [MASK] tokens. However, we can always replace them by either random words or the original word from the non-negated sentence.

Given transformers' well-studied tendency to memorize training datasets and struggle to generalize unless the training set is large and diverse, we create 4 different variants of Negated LAMA and 3 of LAMA. Given each variant has 20,000 examples apiece, we ultimately add a total of 140,000 training examples to Expanded LAMA. All of these are added to training only.

#### 3.3.1. Variants of Negated LAMA

First, we use the entire sequence (both negated and non-negated sentences). To protect against overfitting on non-negated/negated sentence order, we put the non-negated sentence first for 50% of examples, and the negated sentence first for the other 50%. We replace the [MASK] token by its equivalent in the non-negated example and tag it as a 1 for "replaced", with all other tokens tagged as 0. Table 2 shows an example.

Second, we take the non-negated sentences only, changing each masked token to a random token and label it as 1/"replaced" (the other tokens will be tagged as 0/"original"). This is designed to help the model not overfit on "tag random noun if 'not' is present". Table 3 has an example.

Third, we take the negated sentences only, substituting the [MASK] token with its equivalent in the non-negated example and marking the token with a 1. All other tokens are labeled 0. An example of this is the negated sentence in Table 2. The goal of this variant is to help the model understand how a sentence with negation would look without also needing a non-negated sentence in the sequence.

Fourth, we take the negated sentence and duplicate it to further ensure the model doesn't overfit to learning negation only when a non-negated sentence is also present. An example of this would be "The waters of the river Brahmaputra are not shared by China, India, and Bangladesh. The waters of the river Brahmaputra are not shared by China, India, and Bangladesh." Both instances of the word "waters" will be tagged as 1/"replaced" and the others will be tagged as 0/"original".

#### 3.3.2. Variants of Standard LAMA

From Standard LAMA, we create data as follows. First, we include all of the sentences and output all 0's (for "original"). Second, we include all of the sentences twice and output all 0's for the label. This is meant to hedge against overfitting on the 1 non-negated / 1 negated and 2 negated cases from our Negated LAMA dataset. Third, we randomly select 15% of tokens to mask, using ELECTRA-Small Generator to replace them, and mark the replaced token as 1 ("replaced") as in standard ELECTRA training. Tables 4 (for the first two variants) and 5 (for variant 3) provide examples corresponding to the sentence "As early as the 14th century there was domestic hard-coal processing."

For cases where we link two sentences, we use the connecting words from Expanded NLI. Given that there is no real "premise" or "hypothesis", we randomly selected 50% of cases where the first sentence would be the "premise" (and thus the sequence's connector would be sampled from the corresponding linkage words), with the first sentence considered the "hypothesis" for the rest.

## 4. Experiments

### 4.1. Continued Pretraining

We use three different pretrained models, each with roughly 14 million parameters. The first is the standard ELECTRA-Small discriminator checkpoint open-sourced by Clark et al. (2020) (we refer to this as "Base"). The second is additionally trained for

| | | | |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| As | early | as | the |
| 0 | 0 | 0 | 0 |
| 14th | century | there | was |
| 0 | 0 | 0 | 1 |
| domestic | hard | -coal | production. |

Table 5: Using the same sentence pair as Table 2, we show an example for Variant 3.

| Model | RTE | MNLI | SNLI |
|---|---|---|---|
| Base | 0.614 | 0.819 | 0.895 |
| + NLI | 0.581 | 0.813 | 0.896 |
| + LAMA | 0.625 | 0.814 | 0.893 |

Table 6: Accuracy results when finetuning our Base, +NLI, and +LAMA models on RTE, MNLI, and SNLI and evaluating on their standard dev sets.

1 epoch on Expanded NLI (referred to as ("+NLI"), while the third is trained for 1 epoch on Expanded LAMA (denoted by "+LAMA").[8] The latter two take roughly 1 hour on a T4 GPU to train. We then finetune the models on the RTE, MNLI, and SNLI datasets for 5 epochs. For RTE, we run all experiments three times and report the median accuracy; otherwise, we run experiments once per category. We use the AdamW optimizer with learning rate 2e-5, weight decay 0.01, and batch size 64 for both continued pretraining and finetuning.

Our results are in Tables 6 and 7. Relative to finetuning ELECTRA-Small, finetuning the model after continued pretraining on Expanded NLI decreased performance on standard RTE (-3.3%) while preserving standard MNLI and SNLI performance. However, it improved Negated RTE performance (+4.1%). The additional pretraining appeared to have little effect on Negated SNLI (+0.7%). Overall, adding Expanded NLI seemed to have little benefit.

On the other hand, continued pretraining on Expanded LAMA led to dramatic improvements in Negated RTE (+19.9%) while effecting little change on standard RTE (+1.1%)[9]. On MNLI we see effectively no change (+0.6% on Negated MNLI, -0.5% on Standard MNLI). In contrast, Negated SNLI saw a considerable performance drop (-4.6%). Overall, our results on Expanded LAMA closely track Hosseini et al. (2021)'s: solid gains on RTE with little to no improvement (or even decrease) on MNLI and SNLI.

[8]The two non-baseline models have been trained longer on Wikipedia and Books than Clark et al. (2020)'s ELECTRA-Small, but the net increase in training time amounts to less than 0.03% of ELECTRA-Small's pretraining time. Given that going from 50% trained to 100% trained on ELECTRA-Small only led to a 0.9 point increase on the GLUE benchmark, this difference should have no significant impact on the results.

| Model | RTE | MNLI | SNLI |
|---|---|---|---|
| Base | 0.504 | 0.610 | 0.469 |
| + NLI | 0.545 | 0.607 | 0.476 |
| + LAMA | 0.703 | 0.616 | 0.423 |

Table 7: Accuracy results when finetuning our Base, +NLI, and +LAMA models on RTE, MNLI, and SNLI and evaluating on their negated development sets.

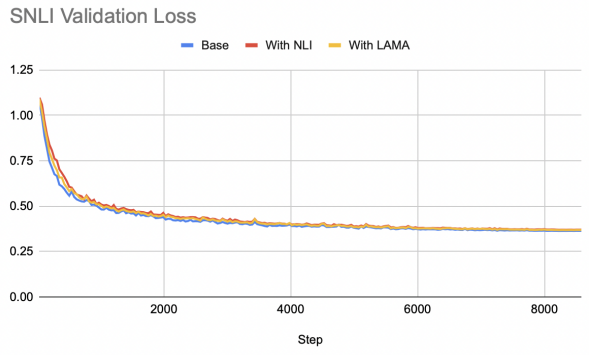
## 4.2. Analyzing MNLI and SNLI Results

To investigate what might be blocking gains on MNLI and SNLI, we finetune each of our three pretrained models for 1 epoch on each dataset, evaluating results every 39 steps (the number of steps in 1 epoch in RTE). Figure 1 shows our validation loss curve for the first epoch of SNLI training. There is near-immediate divergence on the Negated SNLI validation set. Indeed, the lowest validation loss was reached on 156, 156, and 273 steps when finetuning the baseline model, the Expanded NLI-trained model, and the Expanded LAMA-trained model, respectively. Given that just 1.19% of the SNLI training set even has negation cues, this suggests that the models are almost immediately learning to overfit on patterns unrelated to negation.

In contrast, MNLI experiences fluctuation within a narrow range on its negated validation set (see Figure 2). The model appears to neither learn negation nor overfit on other cues on negated examples. To explore why this occurred, we replace every neutral example in both training and Negated MNLI evaluation with an entailment or contradiction example. (With a batch size of 100, we pick the first non-neutral example in a given batch to replace all neutral examples in the batch.)

Our results are shown in Figure 3. When there are only "entailment" and "contradiction" labels and no "neutral" labels, we see the standard picture of (albeit a little noisy) convergence. Indeed, when finetuning for 5 epochs, the dev set accuracy on Negated MNLI without neutral labels reaches 88.92% (Base), 89.30% (+NLI), and 88.44% (+LAMA), showing that the model easily learns this simplified task. In short, the failure to learn that we saw on Negated MNLI only happens when there are neutral examples with negation.

[9]Given these dramatic improvements, it is worth noting that RTE and LAMA do not share any overlap, so the improvements are not due to data leakage.



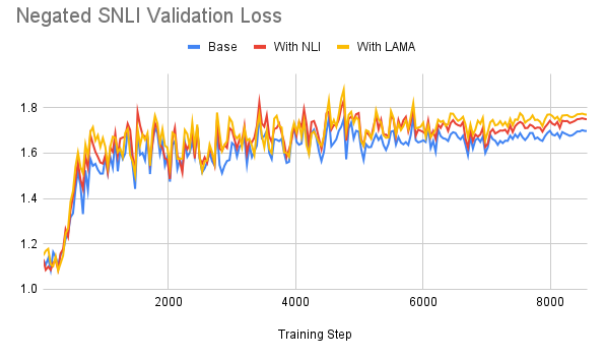
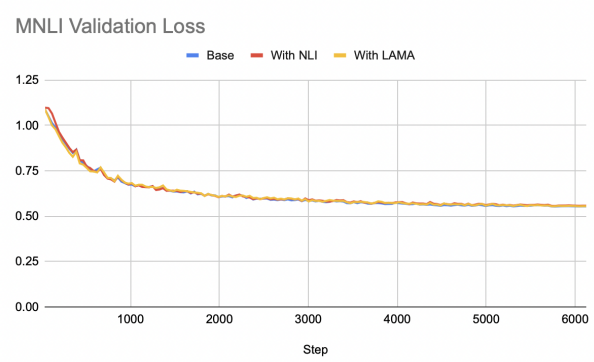




Figure 1: Models trained on SNLI converge normally on the standard development set, but immediately diverge on the Negated SNLI development set. Note that the y-axis scale ranges from 1 to 2.



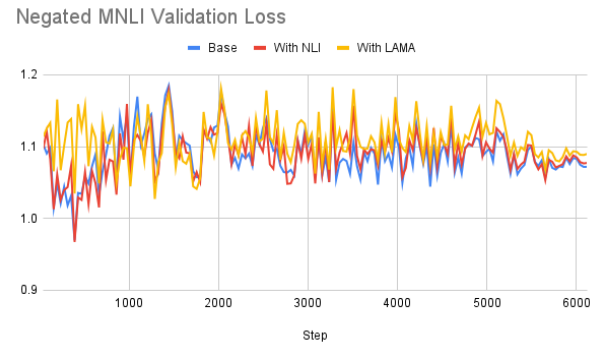




Figure 2: Models trained on MNLI converge normally on the standard development set, but oscillate within a narrow range when evaluated on the Negated MNLI development set without converging. Note that the y-axis scale ranges from 0.9 to 1.2.

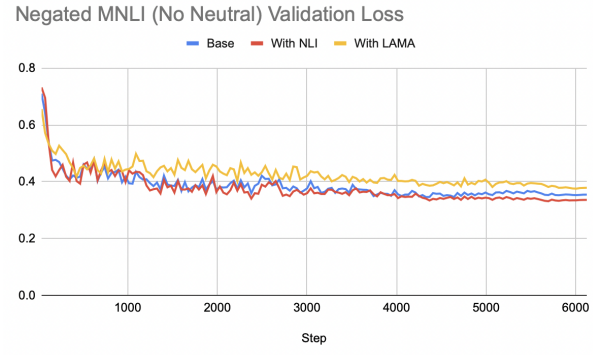



Figure 3: When removing examples labeled "neutral" in both the training and negated development sets, MNLI converges normally on the negated development set.

## 5. Discussion

Like previous work, we have focused on NLI examples where overt negation converts an entailment to a contradiction (and vice versa). By doing so, however, we have exposed a new problem; namely, that neural models have not learned to handle cases where a negation cue does not change an NLI label or changes it from or to the "neutral" label. Future research will have to explore how to teach negation to a model in the multi-class setting, which will likely require a new pretraining task.

However, our work provides a generally effective paradigm for handling binary entailment classification when negation is present. Adding Negated LAMA's 20,000 examples proved sufficient to get good performance in binary RTE despite just 7.16% of the finetuning dataset including negation cues. We conjecture that this was because Negated LAMA examples were similar enough to RTE to generalize to improved downstream performance. On the other hand, Negated LAMA wasn't similar enough to SNLI's negated examples (at least to counteract the severe under-representation of negation cues at 1.19% of the dataset) to secure similarly large gains from continued pretraining. As for Negated NLI, our work demonstrates improvements relative to standard ELECTRA for Negated RTE, weaker performance on standard RTE, and roughly the same results for all other datasets. We theorize that the limited changes might be due to the small quantity of NLI examples in training.

## 6. Conclusion

In this work, we explore continuing pretraining ELECTRA's discriminator on data containing negation to analyze the impact on performance on Negated RTE, SNLI, and MNLI. We find dramatic performance improvements on Negated RTE with

additional pretraining, showing ELECTRA's ability to learn negation cues in binary NLI classification. We also demonstrate that overfitting on non-negated cues due to the lack of negation data in SNLI is the main hurdle in improving Negated SNLI accuracy (showing the importance of having considerable representation of negation in downstream data). Finally, existing models actually perform well on Negated MNLI if "neutral" examples are excluded, though struggle with negation otherwise.

## 7. Limitations

The forms of negation we focused on were "not", noun and verb phrases containing "not" ("cannot", "did not do", etc.). "never", and "n't". This was done to better fixate on why and in what way negation is difficult to learn for language models in a way that would be useful for NLI. Most likely, the system has not learned to react to other negation cues (e.g., affixes like "-less"). Thus, we recommend that future versions of Expanded NLI and Expanded LAMA include a wider variety of different negation cues (see e.g. Morante and Blanco (2012)). In general, negation has a *scope* – the negated text section – and a *focus*, the part of the scope that is explicitly negated. The focus of negation looks to be particularly suited to improving the generation of pretraining data.

Second, the large difference in size between our new NLI examples (1,448) and our new LAMA examples (140,000) could be a confounding factor in our findings. While there is reason to doubt this given that the number of negated Wikipedia examples in the original pretraining dwarfs both datasets, we believe that future work should look into creating additional negated NLI examples.

Third, we obtained good results on Negated MNLI only after removing neutral-labeled data. Future work should refine our continued pretraining task to include neutral-labeled NLI examples.

Fourth, our datasets are English-only, so we recommend that future work extend our datasets to cover multiple languages and explore how much transfer of negation knowledge to other languages occurs when continuing pretraining multilingual models on Expanded NLI and Expanded LAMA.

## 8. Acknowledgements

We would like to thank our reviewers for their valuable feedback. This work was supported by the DARPA AIDA program under AFRL grant FA8750-18-2-0017. We thank the Texas Advanced Computing Center, as well as the Chameleon testbed supported by the National Science Foundation, for providing grid resources that contributed to these findings. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either express or implied, of the Air Force Research Laboratory (AFRL), DARPA, or the U.S. Government.

## 9. Ethics Statement

For the continued pretraining of ELECTRA, we have created two additional datasets based on existing NLI and LAMA data as well as existing Wikipedia and Books data. Thus, we add no new risk of adding personally protected information, offensive material, or biases that could discriminate against marginalized groups.

## 10. Bibliographical References

Ebtesam Almazrouei, Hamza Alobeidli, Abdulaziz Alshamsi, Alessandro Cappelli, Ruxandra Cojocaru, Mérouane Debbah, Étienne Goffinet, Daniel Hesslow, Julien Launay, Quentin Malartic, Daniele Mazzotta, Badreddine Noune, Baptiste Pannier, and Guilherme Penedo. 2023. The falcon series of open language models.

Roy Bar-Haim, Ido Dagan, Bill Dolan, Lisa Ferro, Danilo Giampiccolo, Bernardo Magnini, and Idan Szpektor. 2006. The second pascal recognising textual entailment challenge. *Proceedings of the second PASCAL challenges workshop on recognising textual entailment, volume 6.*

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared 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 M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher 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.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *ICLR*.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2006. The pascal recognising textual entailment challenge. *Proceedings of the First International Conference on Machine Learning Challenges: Evaluating Predictive Uncertainty Visual Object Classification, and Recognizing Textual Entailment, MLCW'05, pages 177–190*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

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.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135.

Iker García-Ferrero, Begoña Altuna, Javier Álvez, Itziar Gonzalez-Dios, and German Rigau. 2023. This is not a dataset: A large negation benchmark to challenge large language models.

Juan Luis Gastaldi and Luc Pellissier. 2021. The calculus of language: explicit representation of emergent linguistic structure through type-theoretical paradigms. *Interdisciplinary Science Reviews*, 46(4):569–590.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and Bill Dolan. 2007. The third PASCAL recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 1–9, Prague. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. *CoRR*, abs/2111.09543.

Md Mosharaf Hossain, Venelin Kovatchev, Pranoy Dutta, Tiffany Kao, Elizabeth Wei, and Eduardo Blanco. 2020. An analysis of natural language inference benchmarks through the lens of negation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9106–9118, Online. Association for Computational Linguistics.

Arian Hosseini, Siva Reddy, Dzmitry Bahdanau, R Devon Hjelm, Alessandro Sordoni, and Aaron Courville. 2021. Understanding by understanding not: Modeling negation in language models. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1301–1312, Online. Association for Computational Linguistics.

Zihang Jiang, Weihao Yu, Daquan Zhou, Yunpeng Chen, Jiashi Feng, and Shuicheng Yan. 2021. Convbert: Improving bert with span-based dynamic convolution.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Aditya Khandelwal and Suraj Sawant. 2020. NegBERT: A transfer learning approach for negation detection and scope resolution. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 5739–5748, Marseille, France. European Language Resources Association.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526.

Michael McCloskey and Neal J. Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press.

Roser Morante and Eduardo Blanco. 2012. *SEM 2012 shared task: Resolving the scope and focus of negation. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 265–274, Montréal, Canada. Association for Computational Linguistics.

OpenAI. 2024. Gpt-4 technical report.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Timo Schick and Hinrich Schütze. 2020a. Exploiting cloze questions for few-shot text classification and natural language inference. *Computing Research Repository*, arXiv:2001.07676.

Timo Schick and Hinrich Schütze. 2020b. It's not just size that matters: Small language models are also few-shot learners. *Computing Research Repository*, arXiv:2009.07118.

Giorgia Nidia Carranza Tejada, Johannes C. Scholtes, and Gerasimos Spanakis. 2021. A study of bert's processing of negations to determine sentiment. *BNAIC/BeneLearn 2021*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

## A. Examples of Sentence Labeling for Expanded NLI

We demonstrate how we create Expanded NLI data using simple (made-up) triplets styled in Hossain et al. (2020)'s format. Here is the first one, which we call Sample Triplet 1.

**Premise:** It is not raining. **Hypothesis:** It is drizzling. **Label:** Contradiction
**Premise:** It is raining. **Hypothesis:** It is not drizzling. **Label:** Neutral
**Premise:** It is not raining. **Hypothesis:** It is not drizzling. **Label:** Entailment

As mentioned above, we convert each of the three NLI examples into a contiguous text. For all examples below, we start with the premise and use "Therefore," as the connecting word for simplicity. We exclude any consideration of punctuation, [CLS], and [SEP] tokens, and consider every word to be its own token.

Because the third member of the triplet is an entailment, it becomes our comparison sentence. Every word in it is labeled as 0/original. The second member of the triplet is labeled neutral, so it will not be included in our dataset. Finally, the first member of the triplet is a contradiction, so we use the preprocessing algorithm. Because the premises of the first and third example are the same, the words "It is not raining." are all labeled as 0/original. However, in the hypothesis, the word "drizzling" in the first example is found to align with the word "not" in the third example. We follow Step 5 to skip to the next word on the third example (since it is longer), which is "drizzing". Because these are equivalent, we label the word "drizzling" as 1/replaced.

We now check to see what the next word after "drizzling" is in both sequences. Because there isn't one, we can stop adding 0s and 1s to our label vector. So the labels for the first triplet member is a vector of all 0s (since every word in the premise and every word up to "is" in the hypothesis are the same) except for the last entry which is a 1 for the word "drizzling". Tables 8 and 9 show how we label the first and third example.

Tables 10 through 14 showcase four additional labeled triplets, this time from data created from Hossain et al. (2020)'s SNLI data. Here is another example, Sample Triplet 2, which we use for Tables 10 and 11:

**Premise:** This little girl is not riding her bike. **Hypothesis:** A young girl rides her bike in the grass. **Label:** Contradiction
**Premise:** This little girl is riding her bike. **Hypothesis:** A young girl does not ride her bike in the grass. **Label:** Neutral

Sentence 1

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| It | is | not | raining. |

Sentence 2

| 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|
| Therefore, | it | is | not | drizzling. |

Table 8: Labeling the third triplet member of Sample Triplet 1.

Sentence 1

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| It | is | not | raining. |

Sentence 2

| 0 | 0 | 0 | 1 |
|---|---|---|---|
| Therefore, | it | is | drizzling. |

Table 9: Labeling the first triplet member of Sample Triplet 1.

**Premise:** This little girl is not riding her bike. **Hypothesis:** A young girl does not ride her bike in the grass. **Label:** Entailment

Like the previous triplet, the third triplet member is an entailment and serves as our comparison sentence. Every word is labeled as 0/original. The second triplet member is neutral and is discarded. The first triplet member is a contradiction, so it is processed by comparing it to the comparison sequence. The premises of both sequences are the same, so all words of the premise are tagged as 0/original. The hypotheses are the same up until we get to "rides" on the contradiction and "does" on the entailment. Because the entailment hypothesis is longer than the contradiction hypothesis, we iterate through the entailment sequence (specifically "not", "ride", "her", and "bike").

Per Step 6 of the preprocessing algorithm, we have not found a word in the entailment sequence equal to "rides" within 4 steps. Thus, we tag the word "rides" as 1/replaced in the contradiction and set our contradiction pointer to the next word - "her". The entailment pointer is set to "not" (one to the right of "does", which was the first word we compared to "rides"). Since these words are still not the same, we continue iterating through the entailment sequence to "ride" and "her", with "her" being where we stop since it matches the contradiction word. Since we did not have to move the contradic-

Sentence 1

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| This | little | girl | is |
| 0 | 0 | 0 | 0 |
| not | riding | her | bike. |

Sentence 2

| 0 | 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|
| Therefore, | a | young | girl | does | not |
| 0 | 0 | 0 | 0 | 0 | 0 |
| ride | her | bike | in | the | grass. |

Table 10: Labeling the third triplet member of Sample Triplet 2.

Sentence 1

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| This | little | girl | is |
| 0 | 0 | 0 | 0 |
| not | riding | her | bike. |

Sentence 2

| 0 | 0 | 0 | 0 | 1 |
|---|---|---|---|---|
| Therefore, | a | young | girl | rides |
| 0 | 0 | 0 | 0 | 0 |
| her | bike | in | the | grass. |

Table 11: Labeling the first triplet member of Sample Triplet 2.

Sentence 1

| 0 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|
| A | swimmer | is | not | doing |
| 0 | 0 | 0 | 0 | 0 |
| the | breaststroke | in | a | pool. |

Sentence 2

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| Therefore, | a | swimmer | does |
| 0 | 0 | 0 | 0 |
| not | use | the | pool. |

Table 12: Labeling the third triplet member of Sample Triplet 3.

Sentence 1

| 0 | 0 | 1 | 0 |
|---|---|---|---|
| A | swimmer | doing | the |
| 0 | 0 | 0 | 0 |
| breaststroke | in | a | pool. |

Sentence 2

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| Therefore, | a | swimmer | does |
| 0 | 0 | 0 | 0 |
| not | use | the | pool. |

Table 13: Labeling the second triplet member of Sample Triplet 3.

tion pointer, we still tag "her" as a 0 before moving to the next word on both sequences. From there, every word in both sequences is the same, and so the remainder of the first example's hypothesis will be tagged as 0/original.

For Tables 12 and 13, we use the following triplet, which we call Sample Triplet 3:

**Premise:** A swimmer is not doing the breaststroke in a pool. **Hypothesis:** A swimmer uses the pool. **Label:** Neutral
**Premise:** A swimmer doing the breaststroke in a pool. **Hypothesis:** A swimmer does not use the pool. **Label:** Contradiction
**Premise:** A swimmer is not doing the breaststroke in a pool. **Hypothesis:** A swimmer does not use the pool. **Label:** Entailment

The third triplet member is an entailment and thus serves as our comparison sequence. We label each word as 0/original. Given that the first triplet member is marked as neutral, we discard it. The second triplet member is a contradiction, so we will compare it to the entailment sequence.

For the contradiction sequence, the premise is the same as the comparison sequence's until we get to "doing" in the contradiction example and "is" in the entailment example. As the entailment example is longer, we iterate through it to find a match for "doing". We traverse through "not" before hitting the word "doing". We tag "doing" as 1/replaced in the contradiction sequence as per the default case in step 5 of our preprocessing algorithm . The rest of the premise of each as well as the hypotheses are the same, so we tag all other words as 0/original.

The triplet in Table 14 is Sample Triplet 4:

**Premise:** A little girl is not sitting in a seat. **Hypothesis:** She is standing on the seat. **Label:** Neutral

Sentence 1

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| A | little | girl | sitting |

| 0 | 0 | 0 |
|---|---|---|
| in | a | seat. |

Sentence 2

| 0 | 0 | 0 | 0 |
|---|---|---|---|
| Therefore, | she | is | not |
| 0 | 0 | 0 | 0 |
| standing | on | the | seat. |

Table 14: Labeling the second triplet member of Sample Triplet 4.

**Premise:** A little girl sitting in a seat. **Hypothesis:** She is not standing on the seat. **Label:** Entailment **Premise:** A little girl is not sitting in a seat. **Hypothesis:** She is not standing on the seat. **Label:** Neutral

The second example is an entailment and thus each word can be safely tagged 0/original. As the other two sequences are labeled neutral, they are to be discarded.

We provide the following triplet as our final example:

**Premise:** A child is not looking out of a door. **Hypothesis:** The door is open. **Label:** Neutral **Premise:** A child is looking out of a door. **Hypothesis:** The door is not open. **Label:** Contradiction **Premise:** A child is not looking out of a door. **Hypothesis:** The door is not open. **Label:** Neutral

Because no example in the triplet is an entailment, we do not add it to our training data.

## B. Negated MNLI Results for Base ELECTRA-Small Split By Where Negation Was Located

We take a closer look at Neutral-labeled data points in MNLI. First, we separate data points by whether they have negation in the premise, in the Hhpothesis, or both. Results are shown in Table 15. Overall, the model makes the most mistakes when the premise contained negation. The percentage of error is lower when the hypothesis contained negation, especially when the premise did too.

In a further manual analysis of 40 gold-neutral examples, half misclassified by the system and half not, no clear patterns stood out. One thing we noticed is that the misclassified examples, but not the correctly labeled examples, seemed to contain many cases that were pragmatically odd. Here is an example:

| Negated Cue Location | Correct | Incorrect |
|---|---|---|
| Premise Only | 0.206 | 0.253 |
| Hypothesis Only | 0.200 | 0.124 |
| Both | 0.204 | 0.013 |

Table 15: Accuracy with our Base model on Negated MNLI, split by whether the negation cue was located in the premise only, hypothesis only, or in both the premise and the hypothesis. Decimal values are represented such that the sum total of 1 represents the entire Negated MNLI dataset (in order to align with our results in Table 7.

**Premise:** Much that was not said about Japanese management style in the 1980s–with its supposed Zen focus and greater sense of process than outcome–was pure buncombe. **Hypothesis:** The information about Japaneese management in the 1990s was correct. **Gold label:** Neutral

The premise basically says that "much that was not said about X was only for show", which is very odd. Examples that are unusual, from a language modeling point of view, could in principle throw the language model off. However, more analysis is required to see if such subtle patterns of pragmatic oddness are more prevalent in misclassified neutral examples than elsewhere.