Learning Robust Negation Text Representations

Anonymous ACL submission

Abstract

Despite rapid adoption of autoregressive large language models, smaller text encoders still play an important role in text understanding tasks that require rich contextualized representations. Negation is an important semantic function that is still not properly captured by such methods, affecting many downstream applications relying on text embeddings. We propose a strategy to improve negation robustness of text encoders, by distilling data from large language models using diverse patterns of negation and hedging. We adopt a standard contrastive learning strategy to finetune a strong BERT-based model, and observe large improvement in negation understanding capabilities while maintaining competitive performance on general benchmarks. In addition, we also show that our method can be adapted to LLMs, leading to improved performance on negation benchmarks.

1 Introduction

Modeling negation is an ongoing problem that text encoders still struggle with. For instance, embedding vectors of minimal negation pairs (*I go to school* vs. *I do not go to school*) have high cosine similarity (Ettinger, 2020; Anschütz et al., 2023), despite their contradictory meaning. This is due to the "distributional hypothesis" (Harris, 1954) underlying text embedding methods, which learn the representation of words based on surrounding context. While highly effective in general, resulting models are insensitive to negation and related phenomena such as antonymy (Mrkšić et al., 2016). This can lead to semantic anomalies in downstream applications, e.g. when searching for products lacking certain properties (Merra et al., 2023) and for exclusion-type queries (Zhang et al., 2024). In a broader sense, negation is closely related to hedging, used to expressed ambiguity, probability, or uncertainty rather than completely refute a premise like negation. Hedging is an even less explored topic in embedding research but is crucial to many language understanding tasks. For instance, hedging is ubiquitous in scientific publications (Crompton, 1997; Pei and Jurgens, 2021), where precise stipulation of the degree of certainty in hypotheses, findings and conclusions (e.g., *clear/weak/no evidence for* . . .) is a critical component of scientific discourse, but again is not generally captured well in embedding vectors, as demonstrated in Figure 1.

	MPNet	H-MPNet
Global warming is a hoax.	0.81	0.39
There is not enough evidence to claim that global warming is real.	0.72	0.58
There is no doubt that global warming is real.	0.78	0.85

Figure 1: Cosine similarities between the sentence *Global warming is real.* and topically-similar sentences conveying different levels of modality, as obtained by MPNet, a strong sentence transformer, and HedgeMPNet (H-MPNet), our model finetuned on HedgeTriple.

Modern large language models (LLMs) are highly effective across a wide range of tasks (OpenAI Team, 2024; Gemini Team, 2024, *inter alia*). Despite this, text encoders (e.g. BERT-based models) are still widely used for text understanding tasks, as: (1) the autoregressive nature of LLMs makes them sub-optimal for learning rich contextual text representations (cf. conditioned on surrounding contexts in bidirectional encoder models); (2) for classification tasks with some amount of labeled data, smaller finetuned text encoder models tend to perform better than LLMs; and (3) text encoders are a critical component of RAG systems, where embedding vectors from text encoders are used in the text retrieval stage to enhance robustness and reduce hallucination (Lewis et al., 2020).

LLMs have the ability to reliably follow instruc-

tions to generate fluent text outputs, which kick-started a line of research on synthetic data distilled from large LLMs to improve smaller, customized models (Eldan and Li, 2023; Wang et al., 2023). In this work, we explore the use of synthetic data to make text encoder models more robust to negation and hedging by further finetuning text embedding models on contrastive triples distilled from a LLM. Our contributions in this work are:

- We propose a data synthesis method that is well-grounded in the linguistics literature on negation and hedging.
- We show that finetuning a text encoder model on synthetic data can significantly improve its performance on negation benchmarks while preserving comparable performance on general benchmarks. Moreover, results with comparable models show the importance of data diversity over quantity.
- We adapt the method to decoder-only LLMs, showing improved negation understanding, with local degradation on benchmarks.

2 Related work

To obtain better text representations, a common and effective strategy is large-scale contrastive finetuning. Specific to improving negation understanding, two works are most relevant, both following the method of first creating minimal pairs which differ only in negation cues, then finetuning a general-purpose text encoder to better differentiate between these pairs. Anschütz et al. (2023) employed a rule-based negator to add verbal negation, modifying the main clause of the sentences by leveraging part-of-speech information (or removing negation cues if they were found in the original). Instruction-tuned LLMs have also facilitated large-scale generation of synthetic data. Günther et al. (2023) used GPT-3.5 to negate sentence from NLI samples, with specific direction to keep the pairs "syntactically very similar", also resulting in verbal negation minimal pairs.

Although not directly related to adding negation, Rezaei and Blanco (2024) also use GPT-3.5 to paraphrase negated samples in NLP benchmarks into affirmative versions, with the motivation that models process affirmative texts better than negation. Jang et al. (2023) explore the abilities of LLMs to follow prompts containing negation, based on manual prompt modifications from a small subset of common benchmarks. To ensure coverage of diverse negation types, Truong et al. (2022b) manually created a small testbed for a broad class of different negation types. To our knowledge, our work is the first to explore a taxonomy-based approach with the aim of generating negation and hedging data at large-scale.

There has also been research on improving the negation understanding of transformer models by modifying the pre-training objective. Hosseini et al. (2021) use unlikelihood training to penalize the likelihood of tokens that are false in a negated sentence. Truong et al. (2022a) add a new mask token to explicitly mask the negation cue in sentences to learn better representations.

Due to the prevalance of hedging in scientific communication, it is mostly explored in the science domain as an uncertainty detection task. The BioScope dataset (Vincze et al., 2008) includes negation and hedging annotations. Ghosal et al. (2022) curate a large scale uncertainty detection dataset from open-access reviews available in the open review platform, containing the most unqiue hedge cues. To model hedging, Pei and Jurgens (2021) introduce a dataset containing sentence- and aspect-level certainty in scientific findings. The work reveals that hedge words alone are not enough to model certainty. For instance, "*Further research is necessary to understand whether this is a causal relationship*" contains 0 hedges but has a high level of uncertainty. This motivates us to employ LLMs to obtain more diverse patterns rather than a template-based approach.

Our work builds on two core ideas from previous work: (1) we use LLMs to create synthetic data, but ground the generation step with clear linguistic instructions; and (2) we adopt a simple contrastive learning strategy to finetune a strong text encoder using the generated data. Beyond achieving large improvements on negation benchmarks, we demonstrate that the strategy retains general capabilities.

3 Method

3.1 HedgeTriple dataset

To make the encoder more sensitive to negation and hedging, we adopt contrastive learning. The crucial part for any contrastive learning algorithm is to collect positive and negative samples. As detailed below, given an affirmative sentence (e.g. *I will go to school*), we assume that a hedged variant (e.g. *I will probably go to school*) is more similar in meaning to the original than the negated text

(e.g. *I will not go to school*). This relationship motivates the use of contrastive learning, minimizing the distance between an affirmative anchor and its hedged variant, while maximizing the distance to the anchor's negated variant in the latent space.

3.1.1 Selecting anchor sentences

We select 50K anchors from the negation triples dataset[1] which was used to train Jina Embedding (Günther et al., 2023), a competitive BERT-based encoder model. The anchors are sourced from five common datasets used in training embeddings — SNLI (Bowman et al., 2015), Multi-NLI (Williams et al., 2018), sentence-compression (Filippova and Altun, 2013), Simple Wikipedia (Coster and Kauchak, 2011), and COCO Captions (Lin et al., 2014) — to ensure general representations across different text types. Most sentences are affirmative and around 50% of the anchors have simple grammatical structures with less than 15 words while complex samples with multiple clauses (> 30 words) account for around 10%.

3.1.2 Synthesizing negation and hedging

Negation For negation, we adopt the taxonomy of negation from Pullum and Huddleston (2002). It provides linguistic definitions along with a sample of typical constructions corresponding to each category of negation. We made explicit how a sentence can be modified to exhibit each type of negation, to compensate for lack of LLM knowledge about some negation types. The chosen prompt, along with an example **`input text`** and `LLM generated text`,[2] is as follows:

```
Text: A yellow and black plane is
flying in the clouds and blue sky.

Negate the text. The types of
negation:

1. "verbal": verbal negation:
when the negation is grammatically
associated with the verb, the head
of the clause.
2. "absolute": Absolute negator:
no (including compounds nobody,
nothing, etc., and the independent
form none), neither, nor, never.
3. "affixal": Affixal negators:
un-, in-, non-, -less, etc.
4. "lexical": Lexical negation:
when the negation is added by
substituting the main predicate of
the sentence with its antonym or
word carrying negative meaning.

Modified text:
1. "verbal": The yellow plane is
not flying in the clouds and blue
sky
2. "absolute": There is no
yellow and black plane flying in
the clouds and blue sky.
3. "affixal": The non-flying
plane is not in the clouds and
blue sky.
4. "lexical": The yellow plane
is grounded.
```

Hedging For hedging, we adopt the hedging taxonomy proposed by Crompton (1997) for scientific findings, which classifies hedges based on their part-of-speech. We simplify the taxonomy into two categories: single-word cues (e.g. *probably*), and multi-word cues (e.g. *it is unclear if*). Even so, we found that LLMs fall back to using specific cues for each type, resulting in very limited diversity of hedging in the generated texts. To address this, we curated a list of cues (134 single-word cues and 45 multi-word cues) from the HedgePeer dataset (Ghosal et al., 2022) and explicitly included a random cue in the prompt for each call to the LLM (full list in Appendix A). An example prompt with input and output is as follows:

```
Text: A yellow and black plane is
flying in the clouds and blue sky.

Add hedging to the text. Two
types of hedging:
1. "word": single-word cue such
as reportedly
2. "phrase": multi-word cue such
as not entirely clear

Modified text:
1. "word": A yellow and black
plane is reportedly flying in the
clouds and blue sky.
2. "phrase": It's not entirely
clear what's happening, but a
yellow and black plane appears to
be flying in the clouds and blue
sky.
```

3.1.3 Constructing triples

We perform a post-processing step to filter out all samples where the generated text is too different from the anchor text (based on Levenshtein distance, with the upper threshold of 60, equivalent to 10 words) and retain only minimal pairs. This is an essential step to ensure that the triples are

[1] https://hf.co/datasets/jinaai/negation-dataset-v2

[2] We use GPT-3.5 for all prompts in Section 3.1.2.

still topically similar. For instance, the pair *{'anchor': 'Swiss bank UBS announced it would cut about 1,600 more jobs at its investment bank after it posted a 8.1 billion Swiss franc loss in the fourth quarter, missing forecasts.', 'positive': "According to UBS's announcement, the bank will likely specify cutting around 1,600 more jobs at its investment bank."}* is technically correct, but half of the main content of the anchor is omitted in the positive sentence. In another instance, the pair does not maintain the contradiction relationship, such as *{'anchor': 'The Red Cross reported that 400 were dead , but this was disputed by Mexican officials .', 'negative': '400 were not dead.'}*. The final dataset consists of 31K anchors, each with 4 negation and 2 hedging generated outputs. We construct triples for contrastive learning by treating anchor–negation pairs as negative instances and anchor–hedging pairs as positive instances, resulting in 248K samples. We name this dataset HedgeTriple, and have made it publicly available at `https://hf.co/ANONYMOUS`.

3.2 Contrastive triple finetuning

Large-scale contrastive finetuning has been shown to be an effective strategy for improving general text representations (Reimers and Gurevych, 2019; Wang et al., 2022). The key idea works by minimizing the distance between an anchor and positive samples, and maximizing the distance between an anchor and negative samples. We adopt the commonly-used Multiple Negative Ranking Loss (MNRL) (Henderson et al., 2017), which contrasts a positive sample against multiple negative samples. In its original form, MNRL only requires anchor–positive pairs and randomly samples positives from other instances which are considered as negatives. In our case, the negatives are generated explicitly, as defined above, to represent linguistic negation. The loss function is as follows:

$$\mathcal{L} = -\sum_{q \in \mathcal{D}} \log \left(\frac{e^{\text{sim}(q,p^+)}}{e^{\text{sim}(q,p^+)} + \sum e^{\text{sim}(q,p^-)}} \right) \quad (1)$$

where q is the query or anchor drawn from dataset $\mathcal{D}$, p^+ and p^- are the positive and negative sample corresponding to q, sim() is a similarity function (cosine similarity between `CLS` embeddings).

To help the model learn to distinguish between closely-related but different text, we explicitly provide hard negative samples which have high lexical overlap but contradictory meaning. As the aim of this paper is to demonstrate the applicability of the generated triples, we did not extensively explore other contrastive learning methods but hypothesize that other contrastive losses would also work well.

4 Experiments

4.1 Baseline

Base model We evaluate several leading general text encoders, namely: Sentence Transformer (Reimers and Gurevych, 2019) and all-mpnet-base-v2[3] (hereafter, **MPNet**). Our model is based on MPNet, which is a BERT-based model pretrained with masked and permuted language modeling objectives, which was further finetuned on 1B sentences pairs for embedding tasks (NLI, text similarity).

Negation-aware model We evaluate two negation-aware encoder models: (1) **Jina**,[4] a T5-based model finetuned on 50K triples focusing on negation;[5] and (2) **NegMPNet**,[6] which is the all-mpnet-base-v2 model further finetuned on 80K pairs of sentences curated from different negation-focused datasets.

Our model We also base our method on the all-mpnet-base-v2 model, which allows for a direct comparsion. Our method works by finetuning the MPNet model using the contrastive loss from Eq (1) applied to our HedgeTriple dataset (see §3.1.2), and name the resulting model **HedgeMPNet**. We release the model at `hf.co/ANONYMOUS`.

4.2 Benchmarks

4.2.1 Negation-focused benchmarks

NevIR (Weller et al., 2024): an information retrieval benchmark, based on CONDAQA (Ravichander et al., 2022). Each sample consists of a pair of contrasting queries, each with one relevant document. The goal is to correctly rank the two documents with respect to each query. We report the Right Rank (RR) metric, which is the percentage of time the models correctly produce the correct rank for the pair of queries, with chance performance of 25%, as for each data sample, the model needs to correctly rank 2 queries.

[3] `https://hf.co/sentence-transformers/all-mpnet-base-v2`

[4] `https://hf.co/jinaai/jina-embedding-l-en-v1`

[5] This is the same dataset we use for selecting anchors.

[6] `https://hf.co/tum-nlp/NegMPNet`

	MPNet	Jina	NegMPNet	HedgeMPNet
Negation benchmark				
NevIR	8.10	14.61	18.08	**40.56**
ExcluIR	69.29	57.36	46.76	**73.09**
Cannot	34.91	30.62	**69.44**	55.68
M3-Counterfactual	16.20	41.91	**51.29**	47.34
Average	32.13	36.13	46.39	**54.17**
General benchmark				
MTEB-Classification	65.07	67.76	**70.83**	69.74
MTEB-PairClassification	83.04	**84.80**	79.05	82.20
MTEB-Reranking	**68.83**	56.42	68.24	66.85
MTEB-Clustering	**43.69**	37.15	38.45	36.88
MTEB-Retrieval	43.10	**44.81**	36.12	35.75
MTEB-STS	80.28	**80.96**	77.58	77.49
MTEB-Summarization	27.49	29.58	27.49	**30.98**
Average (56 datasets)	**58.79**	57.38	56.82	57.14

Table 1: Results on negation and general benchmarks. The reported score for each task is the main metric to evaluate that task; higher is better. **bold** and underline denotes the best and second-best scores respectively.

ExcluIR (Zhang et al., 2024): a benchmark focusing on exclusion queries (e.g. *Apart from Old & Kumar Go to White Castle, what other films has actor Errol Sitahal appeared in?*). The dataset is a modified version of HotpotQA (Yang et al., 2018). We also use RR here, with chance performance of 50% as each query is separately evaluated.

Cannot (Anschütz et al., 2023): an MT evaluation dataset, where negation is a common cause of error. The dataset includes sentence pairs and their semantic similarity scores. We report Spearman's correlation ρ between our model predictions (cosine similarity) and the ground truth.

M3-Counterfactual (Otmakhova et al., 2022): a subset of the M3 dataset, constructed by manually corrupting statements in biomedical literature to evaluate model's robustness in a counterfactual setting. The modification includes adding negation to statements, changing statements into non-evidential sentences (*There is no evidence that ...*), or changing the modality (e.g. by adding hedging words such as *might* or intensifiers such as *certainly*). We reformat the data into text-similarity-style task and assign original–negation pairs a score of -1, original–no evidence pairs a score of 0, and original–hedged pairs a score of 1. Similar to the Cannot dataset, we evaluate the models' performance using Spearman's correlation ρ against the cosine similarity estimates.

4.2.2 General benchmarks

Aside from negation benchmarks, we also evaluate the general capabilities of finetuned models on standard English benchmarks. Specifically, we use the comprehensive general benchmark set of text understanding tasks **MTEB** (Muennighoff et al., 2022), spanning 7 subtasks with 56 datasets.

5 Main findings

5.1 Negation and general benchmark results

As can be seen in Table 1, in general our model ("HedgeMPNet") outperforms all similar-sized text embedding models on negation benchmarks, while maintaining similar performance on general benchmarks. On the negation side, we see large increases on both NevIR and ExcluIR over both general (all-mpnet-base-v2) and negation-focused models (Jina and NegMPNet). Note that the high performance of NegMPNet on Cannot is because it is in-domain data for this model, in that the model was fine-tuned on the training portion of the same dataset. Over general benchmarks, we can observe increases on classification and summarization tasks, and drops on other tasks. One interesting pattern is the large increase on sentiment classification datasets inside MTEB-Classification, showing that this strategy is especially helpful for sentiment-related tasks where people tend to express opinions subjectively (using more hedging) and using terms associated with negation to express negative sentiment.

We further conducted additional experiments to ablate the impact of the HedgeTriple dataset. To save time and resources, for subsequent ablation experiments, we only evaluate on a subset of MTEB that has been shown to correlate highly with overall model performance, as introduced in

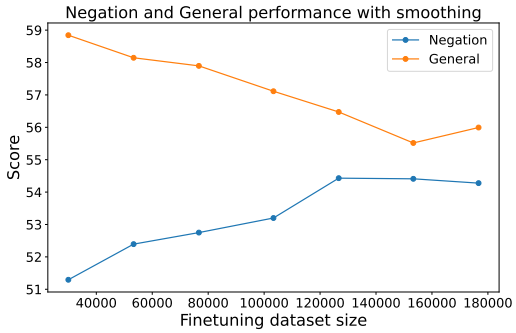

Figure 2: Negation–general performance tradeoff when finetuned on different dataset sizes, smoothed with moving average with window size 3

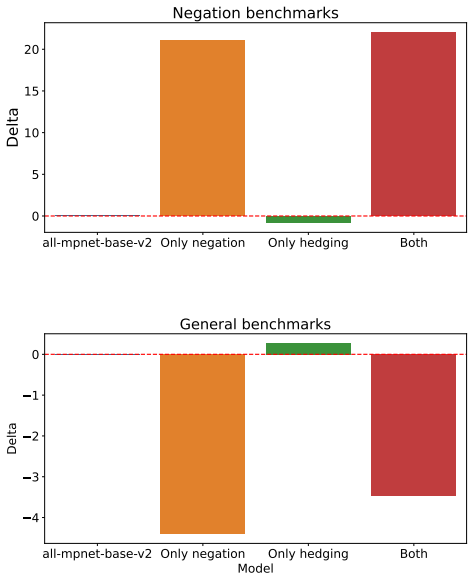

Figure 3: Relative difference wrt. all-mpnet-base-v2 using different portions of data

BehnamGhader et al. (2024).

Balancing negation–general capability tradeoffs Catastrophic forgetting, where a model loses some of its original capabilities, is inevitable when models are further finetuned to adapt to new tasks or domains. Thus, we experiment with finetuning using different data sizes ranging from 10K to 200K instances, to observe the impact of training data size. Results show that finetuning on more HedgeTriple samples leads to larger performance gains on negation benchmarks at the cost of general capabilities. From Figure 2, we can see that performance on negation benchmarks is observed with as few as 10K samples on HedgeTriple. The optimal point to balance out the tradeoff is around 150K training samples. We hypothesize that retention of general capabilities is thanks to exposure to hedging data, and conduct an ablation analysis with respect to data attribution, i.e. finetuning only with either hedging or negation data, to further investigate this.

Data attribution We conducted an ablation study to evaluate the impact of each portion of the data: only using negation data ("Only negation"), only using hedged data ("Only hedging"), or both (Table 2). When only using negation data, we used the original positive sentences from the negation dataset which we sampled the anchors from; while for only hedged data, we used the original negative sentences. The results show (Figure 3) that combining both data types leads to the best performance, and that negation data plays a more important role. Only using hedging data is not beneficial as all the benchmarks considered are more focused on negation, and do not have any explicit measure for hedging. However, finetuning on hedging data is beneficial in retaining general capabilities, with high results on the MTEB-lite set, surpassing even the base model without additional finetuning.

Data contamination We found no exact matches between any of the negation benchmarks and HedgeTriple.[7] N-gram analysis reveals that overlap happens for less than 4% of the samples in all datasets (noting all samples have a minimum length of 5 words). Moreover, the overlap here happens in the retrieval corpus, not on the query set. This is standard in IR and is not considered data contamination. Hence data leakage is negligible.

Diversity vs. quantity As detailed in Section 2, both negation and finetuning using contrastive learning to improve SBERT have been explored in previous work. However, they only consider the most straightforward types of negation: syntactically adding *not* to the main verb either by rules in CANNOT (Anschütz et al., 2023), or using GPT-3.5 in Jina Embedding (Günther et al., 2023). Instead, a main contribution of this work is the adoption of linguistically-sound taxonomies to create more diverse negation data. Our models finetuned on similar data sizes outperform both NegMPNet (∼80K samples) and JinaAI Embedding model (∼50K samples) on the negation benchmarks. This shows that diversity in negation and hedging patterns plays a bigger role than quantity.

[7] Defined as when a sample has a text field (query, doc, text, etc.) that is included in HedgeTriple, or vice versa.

	MPNet	Only negation	Only hedging	HedgeMPNet
Negation benchmark				
NevIR	8.10	35.72	9.83	**40.56**
ExcluIR	69.29	**76.10**	64.83	73.09
Cannot	34.91	54.89	15.28	**55.68**
M3-Counterfactual	16.20	**53.17**	31.31	47.34
Average	32.13	**55.57**	29.98	54.17
General benchmark				
MTEB-Classification-lite	64.56	65.24	63.9	**66.30**
MTEB-PairClassification	90.15	85.17	**94.58**	88.92
MTEB-Reranking	**70.32**	67.65	69.11	68.25
MTEB-Clustering	**39.27**	36.22	36.90	32.79
MTEB-Retrieval	48.46	30.88	**48.79**	34.61
MTEB-STS	**84.87**	78.36	83.58	79.07
MTEB-Summarization	27.49	30.77	30.17	**30.98**
Average (16 datasets)	60.73	56.33	**61.00**	57.27

Table 2: Ablation results on negation and general benchmarks. The reported score for each task is the main metric to evaluate that task; higher is better. "Only negation" and "Only hedging" refer to the setting of finetuning MPNet on only negation data and hedging data, respectively.

	HedgeMPNet	Llama-3-8B-Instruct	Only negation	Only hedging	Llama-3-8B-Hedge
Negation benchmark					
NevIR (0shot)	40.56	74.04	73.13	68.13	**78.13**
ExcluIR (0shot)	73.09	91.71	**93.83**	92.50	93.40
CANNOT (0shot)	55.68	44.16	**63.99**	53.22	60.89
M3-Counterfactual	47.34	56.05	68.59	69.64	**76.27**
Average	54.17	66.85	75.15	70.61	**77.17**
General benchmark					
MMLU (5shot)	N/A	**65.68**	63.59	63.65	63.03
HellaSwag (0shot)	N/A	75.77	**77.17**	76.80	75.31
GSM8K (5shot, CoT)	N/A	**75.36**	66.41	70.96	67.10
Average	N/A	**72.27**	69.06	70.47	68.48

Table 3: LLM results on negation and general benchmarks in comparison with the best performing model from the previous experiment. The reported score for each task is the task-specific main metric "Only negation" and "Only hedging" refer to the setting of finetuning Llama-3-8B-Instruct on only negation data and hedging data, respectively.

5.2 Effect of HedgeTriple on LLMs

We also look at the performance of a current-gen decoder-only LLM (Llama-3-8B-instruct) on negation benchmarks, and whether finetuning it on HedgeTriple can improve its handling of negation and hedging (Table 3). We treat the task as ranking between two documents, with the following prompt:

```
Document 1: doc 1
Document 2: doc 2
Query: q

Which document is more relevant to
the query? Please choose 1 or 2.
Answer:
```

For the CANNOT similarity task, we ask the model to score the sentence pairs:

```
Determine the similarity between
the following two sentences (S1,
S2). The score should be ranging
from -1.0 to 1.0, and can be a
decimal.
S1: sentence 1
S2: sentence 2
Score:
```

Simply applying the LLM in a zero-shot manner, we immediately see much higher performance than HedgeMPNet on both NevIR and ExcluIR. However, Llama-3-8B-instruct is several orders of magnitude larger in parameter size, and much more expensive to apply as a text encoder. Regardless, there is active research on deriving text embeddings from LLMs, such as via bidirectional text encoders (BehnamGhader et al., 2024; Wang et al., 2024).

Next, we convert HedgeTriple into pairs to fine-

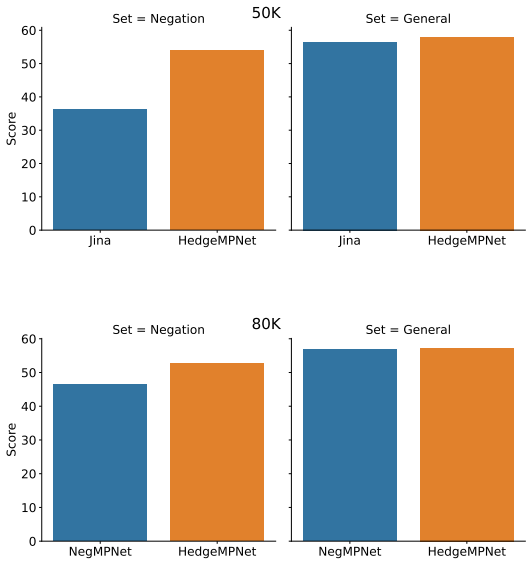

Figure 4: HedgeMPNet compared with similar negation-focused models finetuned on similar-sized datasets.

tune Llama-3-8B-Instruct with LoRA (Hu et al., 2022) (finetuning details in Appendix B). For instance, a triple is converted into two pairs:

```
Sentence 1: A boy holding his skateboard behind him and covering his behind.
Sentence 2: The boy is sitting comfortably without his skateboard and with his behind exposed.
Do the two sentences have opposite meaning? Yes or No.
Answer: Yes

Sentence 1: A boy holding his skateboard behind him and covering his behind.
Sentence 2: The boy, it seems, held his skateboard behind him and covered his behind.
Do the two sentences have opposite meaning? Yes or No.
Answer: No
```

We observe further improvements in the fine-tuned model (Llama3-8B-Hedge) over the base version, showing that the HedgeTriple is also beneficial for current-gen LLMs. Despite there still being room for improvement, overall, LLMs appear to be able to distinguish between negated and non-negated contexts quite well when evaluated in a pairwise setting. However, this finding may not generalize to other negation benchmarks, which LLMs still struggle with (Truong et al., 2023).

In addition, we evaluate the general capabilities of the fine-tuned LLM on three common benchmarks — MMLU (Hendrycks et al., 2021), HellaSwag (Zellers et al., 2019), and GSM8K (Cobbe et al., 2021) — to determine if the fine-tuning has led to catastrophic forgetting. We use the default settings for each benchmark in lm-evaluation-harness (Gao et al., 2023). Overall, we observe comparable performance with and without fine-tuning for MMLU and HellaSwag, but a drop on GSM8K (which contains grade school math problems). We also notice a degradation over the MMLU subset related to mathematics (e.g. high school/college/elementary mathematics, statistics). This finding implies that that negation robustness can negatively impact the arithmetic reasoning abilities of models. We conducted an error analysis on GSM8K and found that most of the errors are due to wrong calculations—even though the equations are correct—and the loss of quantitative commonsense knowledge (see Appendix C for details).

We also performed ablation to see the impact of negation and hedging data on the LLM's capabilities (Only negation and Only hedging in Table 3). Similar to the encoder models experiments, we notice a larger effect of negation data in improving negation understanding capabilities, but combining both negation and hedging leads to the best scores. Over the general benchmarks, hedging data also leads to best retention of general model capabilities. Interestingly, combining both data types leads to worse results compared with using either alone.

6 Conclusion

Negation and hedging are important phenomena that have huge impact on language understanding but are often overlooked when evaluating models' capabilities. In this work, we propose a strategy to improve text embedding robustness to negation and hedging based on contrastive finetuning on synthetic data distilled from LLM. Our prompts are carefully crafted with well-defined linguistic taxonomies to ensure diversity in the negation and hedging patterns. We conducted extensive experiments and observed drastic improvements on negation benchmarks while retaining general capabilities. Furthermore, finetuning an LLM on the generated triples is also beneficial in improving negation understanding abilities, at the cost of a small degradation in mathematical performance.

7 Limitations

Prompting For data generation, we iteratively update the prompts based on manually inspecting the output of LLMs until observing the desired behaviour. Employing automatic prompt optimization technique suchs as DSPy (Khattab et al., 2023) would result in better prompts but we decided not to explore this as the current results are satisfactory.

Other languages As a starting point, we focused exclusively on English, but the same strategy can be readily adapted to other languages. Thus, we claim that the findings of this work are generalizable to a multilingual setting.

Finetuning strategies In both contrastive finetuning of text encoders and LLM supervised finetuning, we experimented with a relatively simple and straightforward strategy and data format. For the LLM, finetuning using more diverse instructions with a reasoning step would likely unlock more sophisticated negation reasoning abilities.

References

Miriam Anschütz, Diego Miguel Lozano, and Georg Groh. 2023. This is not correct! negation-aware evaluation of language generation systems. In *Proceedings of the 16th International Natural Language Generation Conference*, pages 163–175, Prague, Czechia. Association for Computational Linguistics.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2Vec: Large language models are secretly powerful text encoders. *arXiv preprint*.

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*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

William Coster and David Kauchak. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 665–669, Portland, Oregon, USA. Association for Computational Linguistics.

Peter Crompton. 1997. Hedging in academic writing: Some theoretical problems. *English for Specific Purposes*, 16(4):271–287.

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *Preprint*, arXiv:2305.07759.

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.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1481–1491, Seattle, Washington, USA. Association for Computational Linguistics.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2023. A framework for few-shot language model evaluation.

Gemini Team. 2024. Gemini: A family of highly capable multimodal models. *Preprint*, arXiv:2312.11805.

Tirthankar Ghosal, Kamal Kaushik Varanasi, and Valia Kordoni. 2022. Hedgepeer: a dataset for uncertainty detection in peer reviews. In *Proceedings of the 22nd ACM/IEEE Joint Conference on Digital Libraries*, JCDL '22, New York, NY, USA. Association for Computing Machinery.

Michael Günther, Louis Milliken, Jonathan Geuter, Georgios Mastrapas, Bo Wang, and Han Xiao. 2023. Jina embeddings: A novel set of high-performance sentence embedding models. *Preprint*, arXiv:2307.11224.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

Matthew Henderson, Rami Al-Rfou, Brian Strope, Yunhsuan Sung, Laszlo Lukacs, Ruiqi Guo, Sanjiv Kumar, Balint Miklos, and Ray Kurzweil. 2017. Efficient natural language response suggestion for smart reply. *Preprint*, arXiv:1705.00652.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2021. Measuring massive multitask language understanding. *Proceedings of the International Conference on Learning Representations (ICLR)*.

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.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Joel Jang, Seonghyeon Ye, and Minjoon Seo. 2023. Can large language models truly understand prompts? a case study with negated prompts. In *Transfer learning for natural language processing workshop*, pages 52–62. PMLR.

Omar Khattab, Arnav Singhvi, Paridhi Maheshwari, Zhiyuan Zhang, Keshav Santhanam, Sri Vardhamanan, Saiful Haq, Ashutosh Sharma, Thomas T. Joshi, Hanna Moazam, Heather Miller, Matei Zaharia, and Christopher Potts. 2023. Dspy: Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv:2310.03714*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Felice Antonio Merra, Omar Zaidan, and Fabricio de Sousa Nascimento. 2023. Improving the relevance of product search for queries with negations. In *Companion Proceedings of the ACM Web Conference 2023*, pages 86–89.

Nikola Mrkšić, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina M Rojas Barahona, Pei-Hao Su, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Counter-fitting word vectors to linguistic constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–148.

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

OpenAI Team. 2024. GPT-4 technical report. *Preprint*, arXiv:2303.08774.

Yulia Otmakhova, Karin Verspoor, Timothy Baldwin, Antonio Jimeno Yepes, and Jey Han Lau. 2022. M3: Multi-level dataset for multi-document summarisation of medical studies. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3887–3901, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jiaxin Pei and David Jurgens. 2021. Measuring sentence-level and aspect-level (un)certainty in science communications. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9959–10011, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Geoffrey K. Pullum and Rodney Huddleston. 2002. *Negation*, chapter 9. Cambridge University Press.

Abhilasha Ravichander, Matt Gardner, and Ana Marasovic. 2022. CONDAQA: A contrastive reading comprehension dataset for reasoning about negation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 8729–8755, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

MohammadHossein Rezaei and Eduardo Blanco. 2024. Paraphrasing in affirmative terms improves negation understanding. *arXiv preprint arXiv:2406.07492*.

Thinh Truong, Timothy Baldwin, Trevor Cohn, and Karin Verspoor. 2022a. Improving negation detection with negation-focused pre-training. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4188–4193, Seattle, United States. Association for Computational Linguistics.

Thinh Hung Truong, Timothy Baldwin, Karin Verspoor, and Trevor Cohn. 2023. Language models are not naysayers: an analysis of language models on negation benchmarks. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics (*SEM 2023)*, pages 101–114, Toronto, Canada. Association for Computational Linguistics.

Thinh Hung Truong, Yulia Otmakhova, Timothy Baldwin, Trevor Cohn, Jey Han Lau, and Karin Verspoor. 2022b. Not another negation benchmark: The NaN-NLI test suite for sub-clausal negation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 883–894, Online only. Association for Computational Linguistics.

Veronika Vincze, Gy"orgy Szarvas, Rich'ard Farkas, Gy"orgy M'ora, and J'anos Csirik. 2008. The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*, 9(11):1–9.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024. Improving text embeddings with large language models. *Preprint*, arXiv:2401.00368.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13484–13508, Toronto, Canada. Association for Computational Linguistics.

Orion Weller, Dawn Lawrie, and Benjamin Van Durme. 2024. NevIR: Negation in neural information retrieval. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2274–2287, St. Julian's, Malta. 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, New Orleans, Louisiana. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

Wenhao Zhang, Mengqi Zhang, Shiguang Wu, Jiahuan Pei, Zhaochun Ren, Maarten de Rijke, Zhumin Chen, and Pengjie Ren. 2024. Excluir: Exclusionary neural information retrieval. *Preprint*, arXiv:2404.17288.

A List of hedge cues

Single-word cues ['wish', 'conjecture', 'wonder', 'implying', 'unlikely', 'likely', 'slight', 'likelihood', 'possibly', 'sufficient', 'question', 'whether', 'believe', 'wouldnt', 'expect', 'hinting', 'hope', 'suspect', 'if', 'afraid', 'necessarily', 'thinking', 'expecting', 'might', 'apparent', 'felt', 'apparently', 'seem', 'may', 'certainly', 'propose', 'probable', 'imply', 'potentially', 'shouldnt', 'nearly', 'suggestive', 'impression', 'clear', 'can', 'or', 'hesitant', 'probability', 'specify', 'hopefully', 'clean', 'sure', 'ought', 'wrong', 'why/if', 'argue', 'somewhat', 'unsure', 'plausible', 'doubtful', 'must', 'anticipate', 'uncertainty', 'feel', 'clearly', 'either', 'specifying', 'appreciate', 'appear', 'indication', 'couldnt', 'hoping', 'possibility', 'cant', 'suggesting', 'proposing', 'notion', 'presumably', 'potential', 'seemingly', 'doubt', 'uncertain', 'probably', 'assume', 'undoubtedly', 'assumption', 'sense', 'surely', 'arguing', 'cannot', 'clearer', 'should', 'debatable', 'indicating', 'indicate', 'strange', 'speculate', 'weird', 'suggestion', 'think', 'suppose', 'arguably', 'questionable', 'would', 'imagine', 'claim', 'theoretically', 'maybe', 'suggest', 'presume', 'idea', 'like', 'unclear', 'implication', 'almost', 'unknown', 'possible', 'appearence', 'rather', 'implicit', 'puzzling', 'supposedly', 'suspicion', 'impossible', 'wondering', 'argument', 'vague', 'thought', 'hypothesize', 'seeming', 'could', 'guessing', 'tend', 'say', 'wether', 'maynot', 'slightly', 'feeling', 'assuming']

Multi-word cues ['not very clear', 'not surely', 'cannot claim', 'seeming like', 'not clear', 'on the fence', 'not so sure', 'not very sure', 'hard to pin down exactly', 'look like', 'felt like', 'not also sure', 'not really sure', 'not totally sure', 'cannot imagine', 'isnt clear', 'not completely sure', 'not exactly sure', 'no idea', 'not entirely clear', 'could not figure out', 'not at all sure', 'wonder if', 'do not convincingly', 'mostly clear', 'feel like', 'cannot hope', 'not 100 % sure', 'sound like', 'not clearly', 'not convincing', 'not at all clear', 'not conclusive', 'not quite sure', 'not entirely sure', 'can not', 'not totally clear', 'not all are clear', 'somewhat unclear', 'not even sure', 'very unclear', 'seem like', 'can imagine', 'not certain', 'not sure']

B LLM finetuning details

C A closer look at the GSM8K output

We compare the CoT output of Llama-3-8B-Instruct and our finetuned version, Llama-3-8B-

Parameter	Value
Epoch	20
Rank	16
Target modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
Alpha	16
Dropout	0
Bias	None

Table 4: Caption

Hedge, to identify the reasons for the drop in arithmetic reasoning skills. We randomly sample a set of 100 samples where Llama-3-8B-Hedge get the answer wrong, and classify the error types into different categories and summarize the results in Table 5. From this analysis, we can see that the finetuned model regresses somewhat in calculation and commonsense knowledge. Another large number of cases come from a bug in the evaluation toolkit rather than actual errors of the model. One interesting pattern is that model tends to do some inverted calculations, messing up the correct position of the two operands, affecting subtraction and division calculations. Although not very prevalent, we also notice a small portion of cases where the base model gets the correct answer despite incorrect reasoning steps.

Error type	Description	Llama-3-8B-Instruct	Llama-3-8B-Hedge	%
Wrong calculation	When the reasoning step and the final equation to get the result are correct, but the calculated results are wrong	... The total cost of the trip is **$5000 + $5000 + $5400 = $15400** ...	... The total cost of the trip is **$5400 + $10000 = $14000**. ...	20
Fail to include all information	When the logic of the solution is correct but miss one (or more) important steps, leading to incorrect final results	The doll cost as much as 3 action figures, so the doll costs 3 x $5 = $15. One red car costs $4, so 5 red cars cost 5 x $4 = $20. The total cost of all toys is **$15 + $20 + 3 x $5 = $15 + $20 + $15 = $50**.	The doll cost as much as 3 action figures, so the doll costs 3 x $5 = $15. One red car costs $4, so the 5 red cars cost 5 x $4 = $20. The total cost of the toys is **$15 + $20 = $35**	20
Wrong unit	Error in parsing the final answers	Violetta needs 5 crayons and each crayon costs $2, so she needs 5*2 = «5*2=10»10 dollars. She has $20 and will spend $10, so she will get $20-$10 = «20-10=10»**10** dollars in change	Violetta needs 5 crayons, and each crayon costs $2. So she needs 5 * 2 = 10 dollars. She has $20, so she will get $20 - $10 = **$10** in change	19
Fail to reason with multiplication	When the reasoning fails to take into account the quantitative commonsense knowledge (e.g. a week has 7 days, a cow has 4 legs)	Melissa has 8 + 5 + 3 + 12 = «8+5+3+12=28»28 animals to groom over the next week. **Therefore, She will groom 28 / 7 = «28/7=4»4 animals each day.**	She has 8 dogs, 5 cats, 3 birds, and 12 horses that need to be groomed. In total, she has 8 + 5 + 3 + 12 = 28 animals that need to be groomed. **She will split the grooming jobs evenly over the days, so she will groom 28 animals each day of the week**	18
Invert calculation	When the model perform subtraction rather addition or switch positions of the left and right operands	On Thursday, the mechanic earned 6 * 60 =360 for truck tires and 4 * $40 = $160 for car tires. So the total revenue on Thursday was $360 + $160 = $520. On Friday, the mechanic earned 12 * $40 = $480 for car tires. So the total revenue on Friday was $480. The mechanic earned $520 on Thursday and $480 on Friday. **The difference is $520 - $480 = $40**.	On Thursday, the mechanic earned 6 * $60 = $360 for truck tires and 4 * $40 = $160 for car tires. So, the total revenue on Thursday is $360 + $160 = $520. On Friday, the mechanic earned 12 * $40 = $480 for car tires. So, the total revenue on Friday is $480. **The mechanic earned $480 - $520 = -$40** more on the day with lower revenue.	10
Hallucination	When the reasoning chain is not coherent, logical or relevant to the question	Christina had 8 good days, 8 bad days, and 8 neutral days. This is a total of 8 + 8 + 8 = «8+8+8=24»24 days. She also had 3 more days, which are 1 good, 1 neutral, and 1 good. This is a total of 24 + 3 = «24+3=27»27 days. She had 30 days in total, so she has 30 - 27 = «30-27=3»3 days left. Of these 3 days, 2 are good, so she has 2 good days left.	**Christina had 12 good days in total. The first 8 days were good, so that's 8 good days. The next 3 days were good, so that's 3 good days.** 12 - 8 - 3 = 1 good day left in the month.	5
Base model is wrong	When ground truth is incorrect, or when base model gives correct answers but the reasoning chain is wrong,	The interest for the first month is **2% of $100 = «2%*100=2»$2** The interest for the second month is **2% of $100 = «2%*100=2»$2** The interest for the third month is **2% of $100 = «2%*100=2»$2** The total interest is **$2 + $2 + $2 = «$2+$2+$2=6»$6** The total amount Mandy should pay is **$100 + $6 = «$100+$6=106»$106**	After 1 month, Mandy owes Benedict **$100 + 2% of $100 = $100 + $2 = $102**. After 2 months, Mandy owes Benedict **$102 + 2% of $102 = $102 + $2.04 = $104.04**. After 3 months, Mandy owes Benedict **$104.04 + 2% of $104.04 = $104.04 + $2.0816 = $106.1216**. Mandy should give Benedict $106.1216.	4
Repetition	When the model repeats one reasoning step many times.	Let the cost of the lot be x. Then the cost of the house is 3x. The total cost is 120,000, so x + 3x = 120,000. Combine like terms: 4x = 120,000. Divide by 4: x = 30,000. The cost of the house is 3x = 3 * 30,000 = $90,000	The house cost three times as much as the lot, so the house cost 3 x (the cost of the lot) = 3 x (the cost of the house + the cost of the lot - the cost of the house) = 3 x (the cost of the lot) = 3 x (the cost of the house + the cost of the lot - the cost of the house) = 3 x (the cost of the lot) ...	4

Table 5: Error analysis of the sample of 100 errors made by Llama-3-8B-Hedge on GSM8K.