Leveraging Natural Language Inference for Automatic Feedback Assignment

A Masters thesis

Martin Hanzel
École Polytechnique Fédérale Lausanne

Abstract

Natural Language Inference (NLI) is a sub-task of NLP with the goal of determining entailment relations between two texts. A premise p entails a hypothesis h if the meaning of h can reasonably be inferred given p. NLI finds an application in the field of digital learning, where it can be used to automatically evaluate free-form student responses to open-ended questions, a task which has largely been infeasible until the advent of deep language models. Using NLI, an entailment relationship is determined between the student's response and a set of correct and incorrect reference answers. Immediate, personalized feedback can therefore be given by associating a piece of feedback to each entailed reference answer and returning the most relevant feedbacks to the student after response submission.

NLI models are intended to identify a semantic relationship between two texts, but can be fooled by adversarial examples containing confounding knowledge, incorrect word associations, and more. While NLI models were found to work well in a digital learning setting with some modification and under non-adversarial circumstances, they are not a panacea for answer verification and feedback assignment, and work best when augmented with other techniques such as classical NLP methods or premise engineering.

Master of Computer Science
Submitted 26 August 2022
Supervising lab: ML4ED
Supervising company: Taskbase AG
Supervisors: Anette Hunziker, Thiemo Wambsganss

Contents

1 Introduction

Dear reader, do you recall your life as a schoolchild or student? Surely you must remember the experience of handing in an assignment, then anxiously waiting days for your grade. Or getting a big, red $\boxed{\text{F}}$ on a test because you didn't understand something but then receiving no feedback on what you did wrong?

Education is one of the best investments one can make in their life, and yet our teachers and teaching assistants are constantly overworked and spend too much time correcting assignments [26, 44, 83]. Giving each individual student the feedback they deserve is a time-consuming affair that is seldom realized. Many times, we learn from trial and error, and from failure, but if we fail and don't know why, frustration overshadows the joy of learning.

Today, computers are as abundant as books, and a lot of learning has moved from the traditional pen-and-paper to the modern world of tablets, e-books, videos, PowerPoints, and online courses. The early 2010s saw the rise of massive open online courses (MOOCs), such as Coursera or Udacity, that serve a huge number of students over the Internet. More recently, the COVID-19 pandemic highlighted the need and value of digital learning resources as students worked from home. Digital learning is becoming more commonplace.

Digital learning platforms are not yet fully autonomous. Human correctors who evaluate students' learning progress are still needed. While the correction of "closed-ended" questions, such as multiple choice, fill-in-the-blank, or mathematics, is easily automated, the task of correcting free-form, "open-ended" responses in real time is much more difficult. An even greater challenge is parsing such a free-form response and, based on mistakes that the student made, assigning personalized feedback that addresses the mistakes and gives hints on how to succeed.

The purpose of this project is to evaluate an AI method of matching open-ended responses with personalized feedback items so that students can get tailored, immediate feedback on open-ended questions that require higher-order thinking.

Receiving tailored feedback to open-ended questions undoubtedly improves the learning experience and creates a more interactive and engaging learning environment [13, 125, 137].

1.1 This work

This thesis details an exploratory research project at Taskbase AG, a educational technology (EdTech) company. The goal of the project is to study the feasibility and value of employing artificial intelligence, specifically machine learning (ML) and natural language inference (NLI) techniques, to feedback assignment in digital learning. It will explore NLI's benefits and drawbacks in digital learning, how to adapt digital learning methods to take better advantage of the power of NLI, and how NLI can be built upon to specialize it for the field of digital learning.

1.2 Outline

You are now reading Section 1. Welcome! Section 2 presents the basics of digital learning platforms and introduces NLI as a language processing task. Section 3 describes prior work in technology and feedback in the (digital) classroom, recalls the history of NLI from its inception to the state-of-the-art, and presents past methods of using NLI in digital learning. Section 4 introduces the goals, expected contributions, and research questions of this work. Section 5 outlines the data collection, model selection, and data-gathering processes. It also presents an empirical evaluation into how NLI models work (and how they fail) on various open and education-specific corpora, through a series of experiments that explore the sensibilities and failure cases of several different NLI models. Section 6 brings together all the findings to suggest a framework on how to employ NLI models in digital learning platforms to maximize the usefulness of feedback given to students. Section 7 lists many additional topics and questions that were raised by this work.

In the appendices, the reader shall find a list of tables and figures of raw data for each quantitative experiment, datasets developed for this project, and a glossary of common terms.

2 Foundations

> If you wish to make an apple pie from scratch, you must first invent the universe.
>
> *Carl Sagan, Cosmos*

This work does not attempt to invent the universe. However, it will be useful to introduce some basic concepts that the reader should know about digital learning platforms and the task of Natural Language Inference (NLI).

2.1 A model for tasks and tailored feedback

For the purposes of this work, a **digital learning platform** is a tool in education where instructors can create and assign exercises, assignments, quizzes, or tests; and where students can complete these on-line . Digital learning platforms also have a component that automatically evaluates responses to many kinds of questions, lifting some of the burden off the shoulders of correctors. In contrast to traditional pen-and-paper approaches, students can complete exercises and correctors can grade them anytime, anywhere. The Moodle software at EPFL satisfies this definition of a digital learning platform.

Within an assignment, quiz, exercise, or test on a digital learning platform, there is a list of **tasks**, or questions to respond to. Students respond to one task at a time. When a response is submitted, the digital learning platform may save the response and move on to the next task, or offer immediate formative feedback, motivational and/or corrective, using some automated mechanism. How immediate feedback is generated depends on whether the task is *open-ended* or *closed-ended*.

2.1.1 Tasks are open-ended or closed-ended

Tasks can be **open-ended** or **closed-ended**. In the context of this work, a closed-ended task is one with a clear expected response that can be checked for correctness with a rubric or key. Closed-ended tasks are typically answerable by multiple-choice or fill-in-the-blank, and correction is easily automatable. Many tasks in mathematics are also closed-ended tasks, since programs exist that can parse a math expression and precisely evaluate its structure and result.

Open-ended tasks are much more interesting because they require the student to formulate an answer in their own words. The response takes the form of natural language, usually in the form of one or several sentences. There is no definite answer key — because there are so many ways to formulate an answer to an open-ended question, responses from one student to another will vary.

Open-ended tasks
Explain what intention could be associated with Barack Obama's "hope" and "change" campaign slogans in 2008.
Explain what a successful company should offer to its customers.
What does the wolf do by inviting Little Red Riding Hood into the house?
Why must you know the molarity of a solution in order to efficiently perform a chemical reaction?
Explain in one sentence how Mozart's works shaped contemporary music.
Closed-ended tasks
What does the acronym "NLP" mean?
In what year was Otto von Bismark born?
Which of these countries have *not* adopted the Euro as their currency? Check all that apply.
Who was the president of France in 2000?
Fill in the blank: "Sur nos monts, quand le soleil annonce un ______ réveil,"
Write the formula for calculating the roots of a parabola.

Table 1: Examples of open- and closed-ended tasks.

Ways in which a response may vary are sentence structure, word substitution (e.g. synonyms), presence/absence of capitalization, presence/absence of punctuation, inconsistent grammar, etc. There may also be several correct responses to a task, for example:

Question: *What are the effects of climate change?*
Answer 1: *Earth's weather patterns will be disrupted.* (T 2.1)
Answer 2: *The average temperature will rise.*

Open-ended tasks provide greater didactical value since they require students to synthesize a response from scratch, instead of ruling out incorrect options as in multiple-choice, or blindly guessing in a narrow context as in fill-in-the-blank. Open-ended tasks challenge students' higher-order reasoning and critical thinking skills [7, 39, 93, 97, 86, 144].

Table 1 lists some examples of closed-ended and open-ended task prompts.

2.1.2 Students should receive feedback

Receiving feedback is a crucial component of the learning process. Based on feedback to responses, students alter their conceptions in order to better understand the material being studied — this is the principle of learning. Feedback can take many forms: a “correct”/“incorrect” label to a response, a numeric grade, a motivational message (e.g. “Well done!”), a (partial) correct answer to a task, a hint on reaching the correct answer, an explanation of a student’s misconception, and so on [125]. When feedback is given during the learning process, as opposed to at the end, it is called **formative feedback**, and its purpose is to change the way a student thinks and learns about a particular subject [124].

Feedback has a property of timing. Feedback given after a delay is the norm for pen-and-paper assignments and formal tests/exams. Feedback given immediately can be achieved *en masse* by automating the assignment process and evaluating student responses with a machine. Of course, in a traditional assignment, the more assistants an instructor has, the speedier the marking. On the other hand, some learning environments, such as many MOOCs, seldom offer feedback due to the amount of learners, unless a learner pays for the course.

Good feedback has three important traits that are relevant in this work:

Elaborative Good feedback elaborates on the student’s response or gives hints on how to approach the problem. Simple feedback like a “correct”/“incorrect” label or numeric grade is not elaborative [125].

Immediate Good feedback is given right after the student submits their answer to the task.

Specific Good feedback addresses specific elements of a student’s answer, why those elements might be wrong, and how they can be improved [125].

This work will focus on the concept of **personalized** or **tailored** feedback. Such feedback is elaborate and specific, and ideally personalized to a particular student and their learning level. The ultimate goal of digital learning platforms is to make the assignment of tailored feedback immediate for all types of tasks.

Assigning good feedback in closed-ended tasks is quite easy. Verification is automatable in real time using a computer. Since the space of possible and relevant answers is small, instructors can anticipate mistakes and provide elaborated and specific feedback ahead of time.

Assigning good feedback in open-ended tasks, however, is much more difficult. Because the student answers in natural language, there is a huge space of possible answers, and so feedback cannot be assigned based on a simple lookup table or even a set of rules. It would be impossible for a human instructor to exhaustively create feedback for every possible way a student might phrase their response, as some of this work’s examples will demonstrate. Yet, open-ended questions have enormous didactical value over closed-ended since they challenge students' higher-order thinking skills [7, 39, 93, 97, 86, 144].

2.1.3 The task model

How can an immediate, tailored feedback assignment system be developed that works for open tasks? To ponder this, it is essential to define a **task model** that prescribes the elements of a task:

Prompt A prompt is a short piece of text that poses a question or instructs the student to give an answer to something. Optionally, the prompt may provide context of the question. For examples of open-ended and closed-ended prompts, see Table 1.

Hypotheses Hypotheses are reference responses to the task, typically written by the instructor or task author. These are two types of hypotheses. **Correct hypotheses** are reference correct responses to the task, i.e. expected responses. **Mistake hypotheses** are ones that capture some mistake that the student makes or a misconception that they have about the learning material. Hypotheses should resemble actual student responses, whether they be correct or incorrect.

Feedback Each hypothesis can be associated with one or more feedback items, which are shown to the student after they submit a response. Feedback items should address the misconception in their associated mistake hypotheses.

Student responses A task accumulates responses as students attempt the task.

Feedback is assigned to students, roughly speaking, by picking which hypotheses are closest to a certain response (there can be more than one). If any of the matched hypotheses are mistake hypotheses, their corresponding feedback items are shown. Otherwise, the response is correct and the student passes the task.

Every hypothesis should have a piece of associated feedback. In the case of a mistake hypothesis, the feedback should address the mistake made and give the student some instruction or background as to why the response is wrong. If the response is correct, motivational feedback (e.g. *Great job! Climate change is actually predicted to make weather patterns more extreme across the globe.*) is also helpful to heighten student satisfaction and engagement [125].

The set of responses which match only the correct hypotheses is the **correct class** of responses. The set of responses which contain the same misconception

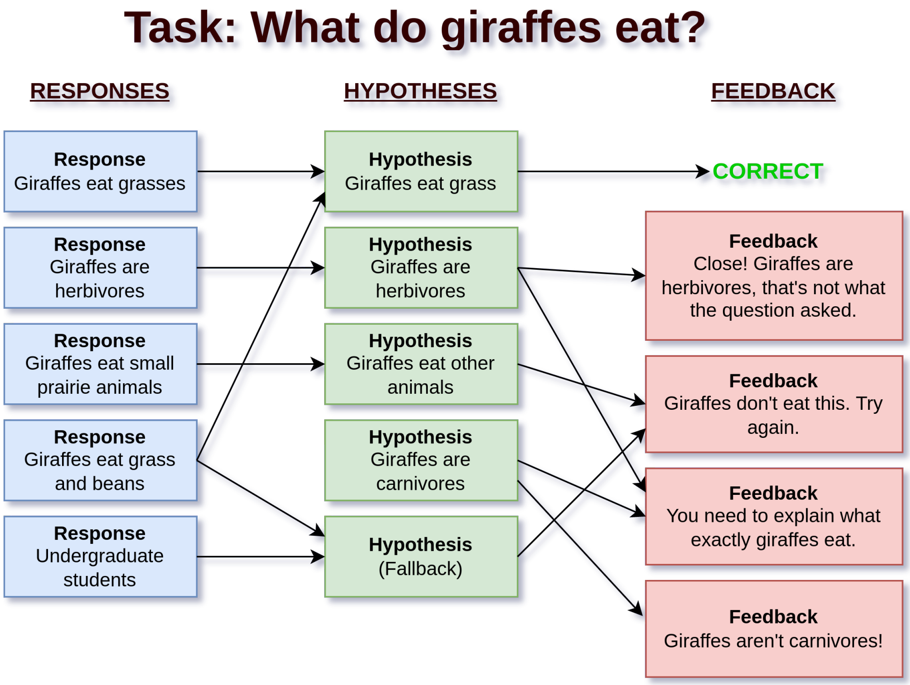

Task: What do giraffes eat?
RESPONSES
HYPOTHESES
FEEDBACK
Response Giraffes eat grasses
Response Giraffes are herbivores
Response Giraffes eat small prairie animals
Response Giraffes eat grass and beans
Response Undergraduate students
Hypothesis Giraffes eat grass
Hypothesis Giraffes are herbivores
Hypothesis Giraffes eat other animals
Hypothesis Giraffes are carnivores
Hypothesis (Fallback)
CORRECT
Feedback Close! Giraffes are herbivores, that's not what the question asked.
Feedback Giraffes don't eat this. Try again.
Feedback You need to explain what exactly giraffes eat.
Feedback Giraffes aren't carnivores!

Figure 1: Illustration of the task model. Student responses (left) are associated with hypotheses (centre), which cause feedback to be returned to the student (right).

or mistake is a **mistake class**. A response may belong to several mistake classes, and even to correct *and* mistake classes, but no response belonging to a mistake class may be truly correct.

Figure 1 illustrates the task model for a single task. Note the presence of a hypothetical fallback hypothesis, which is matched if a response contains some knowledge not matched by the other hypotheses. For example, “Giraffes eat grass and beans” could possibly match the first hypothesis, but “beans” isn’t something that giraffes eat, so this misconception must be captured in some way.

2.1.4 Creating a task

When creating a task, it is the instructor’s responsibility to create hypotheses, feedback items, and assign hypotheses to feedback items. This implies that the instructor needs to know of every possible misconception ahead of time so that all students can get immediate feedback. However, it is also possible to create missing hypotheses and feedback items later, once there are responses, at the cost of delaying feedback assignment. Once the hypotheses and feedback items are created, they can be assigned immediately and future students will benefit.

Researchers have explored ways to ease feedback creation and optimize the quality of feedback during the creation phase (for example, [92] — more in Section 3 on page 20). Creating feedback is outside the scope of this work — it is assumed that good feedback has already been created and that the only problem that remains is assigning it.

To achieve immediate feedback assignment, a machine must automatically match responses to hypotheses. Thus, it is important for the instructor to design tasks in such a way that they be easy for machines to understand. If the prompt is too complex, then the hypotheses and responses will also be complex, raising the likelihood that the machine makes a mistake. Algorithms that "understand" natural language are still in their infancy, with state-of-the-art language models reaching a human level of performance only on very specific language understanding tasks [17, 107].

2.1.5 Assigning feedback

This work deals with the problem of matching student responses to task hypotheses. As mentioned above, enumerating every possible response is impractical. Responses containing tricky language elements such as negation, or variable sentence structure, might also be too challenging for classical language processing tasks to solve. Consider the following hypothesis and responses:

Task: *Write a sentence about Charlie Chaplin and what he is known for.*
Hypothesis: *Charlie Chaplin was a popular film star.*
Response A: *Everyone loved seeing Charlie Chaplin in movies.* (T 2.2)
Response B: *As a film star, Charlie Chaplin was very popular.*

Perhaps one can see how a computer might naively match the response B to the hypothesis using classical NLP techniques, but what about A? Apart from the name, the sentence is entirely different in structure even though it conveys the same information.

Sentence similarity approaches such as bag-or-words or word embeddings fail here. Because of its different structure, response A would be judged as being far from the hypothesis. Worse, two contradicting sentences may appear very similar:

Task: *How is shopping related to financial responsibility?*
Hypothesis: *Shopping teaches you to be careful with money.* (T 2.3)
Response: *Not shopping teaches you to be careful with money.*

The addition of the negation makes the sentences very similar but distinct in meaning. A sentence similarity algorithm would score them as very close. Or consider modifying Response A to the Charlie Chaplin task by replacing the word "loved" with "avoided". These words aren't exactly antonyms, yet completely change the semantics of the sentence while preserving perfectly its structure.

A response should match with a hypothesis not by its structure or syntax, but by its *meaning*. The algorithm should *understand* the response and hypothesis and draw conclusions about whether or not they mean the same thing. This is the task of Natural Language Understanding, a crossing of NLP and AI that aims to endow machines with reading comprehension and reasoning. This work shall explore precisely how these AI-driven methods can be applied to verify the correctness of student responses and extract any mistakes the student might have made by matching responses to correct and mistake hypotheses.

2.2 Natural Language Inference

Natural Language Inference (NLI) is a broad sub-task of Natural Language Understanding (NLU) and Natural Language Processing (NLP) that deals with recognizing entailment relationships between two texts. A **premise** text p is said to **entail** a **hypothesis** text[1] h if h can (most likely) be inferred from p. In similar words, p semantically implies h, which we denote mathematically with the relation $p \vDash h$, following MacCartney's and Manning's notation [78, 79]. The entailment relation is an if-and-only-if relation. The simplest form of NLI is therefore a classification task that results in one of two outcomes:

$$NLI_{\text{2-WAY}}\langle p, h\rangle = \begin{cases} \textsc{entailment} & \text{if } p \vDash h \\ \textsc{not entailment} & \text{if } p \nvDash h \end{cases}$$

We can complicate the NLI problem further and split the NOT ENTAILMENT class into two other cases: CONTRADICTION and NEUTRAL:

$$NLI_{\text{3-WAY}}\langle p, h\rangle = \begin{cases} \textsc{entailment} & \text{if } p \vDash h \\ \textsc{contradiction} & \text{if } p \vDash \neg h \\ \textsc{neutral} & \text{if } p \nvDash h \land p \nvDash \neg h \end{cases}$$

[1] To avoid confusing the meaning of "hypothesis", this report shall use the term "NLI hypothesis" or "language hypothesis" to refer to a hypothesis in the context of NLI, and "task hypothesis" to refer to a hypothesis within a learning task, when the meaning is not clear.

The three-way NLI definition discriminates between contradictory texts and non-contradictory texts. Roughly speaking, if given p then h certainly[2] cannot be true, p is said to **contradict** h. An alternate formulation is that the opposite of h follows from p.

In the final case, p implies neither h nor $\neg h$. The outcome is NEUTRAL. The truths of both p and h are independent, and the statements are **compatible** [78]. This outcome may happen when the texts are unrelated, or if the texts are related but there is some knowledge in h that is not fully covered by p. For example, the premise

p: *The blue fox goes swimming*

entails the hypothesis

h_1: *The fox goes swimming*

but using the hypothesis

h_2: *The fox goes swimming in the lake*

produces the prediction NEUTRAL since the premise cannot prove or disprove that the fox goes swimming specifically in the lake.

Examples

Let us consider three examples from the MultiNLI corpus [147].

p: *While it's probably true that democracies are unlikely to go to war unless they're attacked, sometimes they are the first to take the offensive.* (T 2.4)

h: *Democracies probably won't go to war unless someone attacks them on their soil*

p: *Harlem was our first permanent office, he said.* (T 2.5)

h: *Harlem did a great job*

p: *What's truly striking, though, is that Jobs has never really let this idea go* (T 2.6)

h: *Jobs never held onto an idea for long.*

Text T 2.4 is an example of ENTAILMENT. The meaning of h can be directly inferred from p. A human reading p would in all likelihood say

[2] or most likely, depending on one's interpretation of the NLI task

that h is a logical consequence of p. Text T 2.5 shows a premise and hypothesis which are NEUTRAL. Although the subject matter is similar, there is nothing in p that would imply that h is true or untrue. Text T 2.6 shows a CONTRADICTION. The premise contains knowledge that is at odds with the hypothesis. A human reader, having read p, would assume that h is false.

The duty of an **NLI model** is to implement the above definitions in a computer program. An NLI model takes a text pair $\langle p, h \rangle$ and returns a tuple of probabilities for each 2-way or 3-way outcome.

Table 2 shows the 2-way and 3-way NLI problems and their corresponding outcomes depending on the relationship between p, h, and $\neg h$.

Note the special case where $p \vDash h \wedge p \vDash \neg h$. Such a case is a logical contradiction and should never happen in the real world (however, because natural language is ambiguous, it sometimes does! See Text T 6.1). If such a case is indeed encountered, an NLI model is free to choose an outcome based on the knowledge that it has. The outcome is undefined. In the world of machine learning, NLI datasets are labeled by human annotators who may choose only among the valid labels, therefore an ML model would never encounter an example where the gold truth is this undefined condition.

$\mathbf{p \vDash h}$	$\mathbf{p \vDash \neg h}$	**2-way outcome**	**3-way outcome**
✓	×	ENTAILMENT	
×	✓	NOT ENTAILMENT	CONTRADICTION
×	×	NOT ENTAILMENT	NEUTRAL
✓	✓	UNDEFINED	

Table 2: Table of NLI outcomes for 2-way and 3-way entailment. The last row is a don't-care case that should never arise in practice since it a logical impossibility — the outcome for this case is undefined.

While most present work in NLI research deals with the 3-way definition of NLI, this work deals mostly the 2-way definition. In the context of digital learning, whether a response entails a task hypothesis or not is a yes/no decision — no distinction needs to be made between NEUTRAL and CONTRADICTION.

A variant of entailment, called **bidirectional entailment**, is a property of a text pair that indicates semantic equivalence:

$$p \equiv h \text{ if and only if } p \vDash h \wedge h \vDash p$$

Premise	Hypothesis	Label
People are watching while a construction crew builds a bridge.	The bridge is under construction.	Entailment
A cricket batsman has been bowled out middle stump.	A cricket game is taking place.	Entailment
An old man with a braided beard wears a tie dye shirt.	A 70 year old man with a beard.	Neutral
A silhouette of a man walking.	The man is very tall	Neutral
A person in jeans is standing up and doing a wheelie on the back of a motorcycle.	a dog is doing a wheelie	Contradiction
A child splashes in a lake.	A child plays soccer.	Contradiction

Table 3: Example premise-hypothesis pairs from the SNLI dataset. SNLI was constructed from the Flickr30k dataset [150] of image captions.

Or, p is **equivalent** to h if both texts entail each other. In this case, both p and h contain exactly the same information. We can say that p is a *paraphrase* of h and vice versa. Bidirectional entailment is explored deeper in Section 5.5.

NLI is also known by the name Recognizing Textual Entailment, or RTE. Many papers explicitly state that the two terms are interchangeable, and preference for one over the other seems to be subjective [102]. There is no consensus yet in the NLP community on the precise difference between the two terms; this work shall use the term NLI in order to be consistent with published language models and datasets, which overwhelmingly prefer to use NLI in their names.

Table 3 shows a few premise-hypothesis examples from the SNLI dataset [16].

2.2.1 NLI has broad applications

The authors of the first NLI challenge [32, 33] suggest many different applications of NLI in real-world settings. A few applications of note:

Information Retrieval (IR) A user query is submitted to an IR system, which considers its documents as premises. Documents which entail the user query are returned.

Comparable documents Given two documents, the system identifies sentences in document A that are lexically similar to sentences in document B. If these sentences entail, then the documents are comparable. The applications of this are finding related news articles, or this approach could also find a niche in plagiarism detection.

Question Answering (QA) Candidate answers from an open-book QA system can be validated or ranked by recognizing entailment between the document containing the answer (p) and a cloze statement augmented with the retrieved answer (h).

Answer assessment This is the application which is most relevant to this work. Student responses to open-ended questions are check for entailment against a reference answer. The answers are scored depending whether they entail or do not entail the reference.

The answer assessment of NLI ties in very well to the task model proposed in Section 2.1.3 on page 10. Student responses to tasks are matched with task hypothesis by submitting to an NLI model (a) the student response r as an NLI premise, and (b) a task hypothesis h as the NLI hypothesis. If the model predicts ENTAILMENT, then r belongs to h's class of responses. Each task hypothesis should have some associated feedback, and the feedback of entailed hypotheses are shown to the student. If r entails *only* the correct task hypothesis, then the response is graded as correct and the student can move on to the next question. Otherwise, the student tries again on the same task. This response-feedback cycle is illustrated in Figure 2.

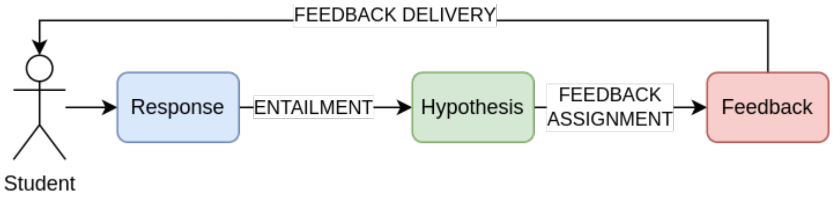


Figure 2: The feedback cycle based on the task model in Section 2.1.3 on page 10 using an NLI approach to feedback assignment.

2.2.2 NLI is a vague task

NLI is not a precise science. For a premise-hypothesis pair $\langle p, h \rangle$, determining the gold label can be difficult or even impossible, depending on the context and wording of the two texts. It therefore follows that the process of determining an entailment relationship between two texts is error-prone and greatly dependent on training data when resolving ambiguous cases.

Section 6.1 explores (non-exhaustively) several sources of ambiguity and how they make the task of NLI challenging and sometimes non-deterministic.

2.3 Taskbase

Taskbase AG is a Zürich-based company that develops tools for online learning. Its major product is the Taskbase Learning Application Platform (which we will call the **Taskbase Platform** throughout this report, see Figure 3 on the following page for a screenshot), a web application that facilitates evaluating students' knowledge and learning progress. The Taskbase Platform allows instructors to create digital assignments for students using a variety of question types (e.g. multiple choice, mathematics, fill-in-the-blank, short answer), evaluate student responses using different classical and AI methods, and return personalized feedback to the student based on mistakes that the student makes in their response.

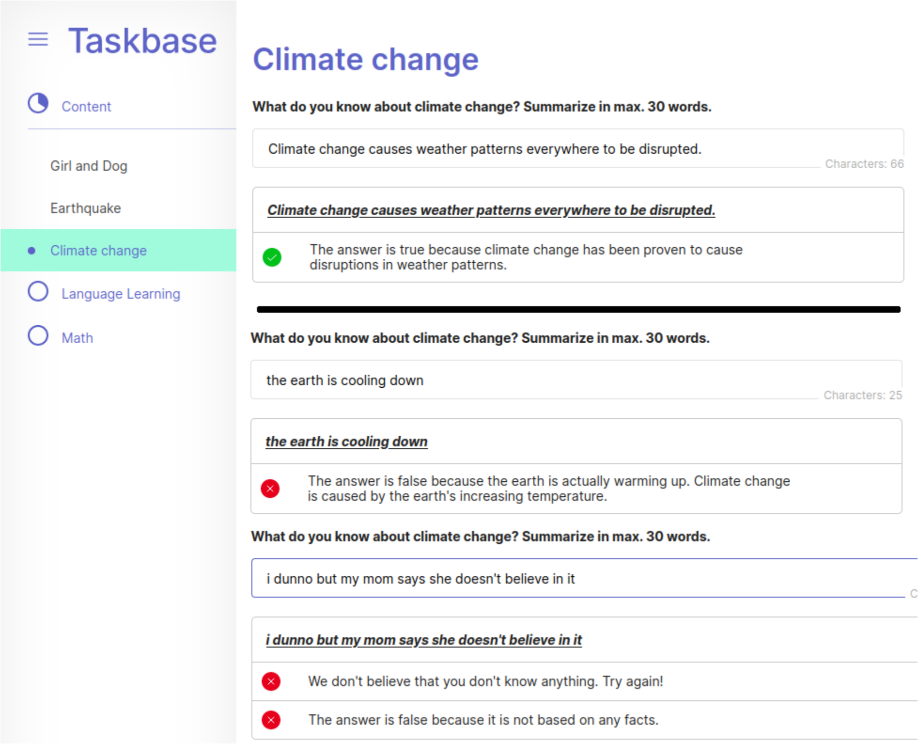
Taskbase's users speak many languages. Datasets collected from the Taskbase Platform include texts in English, German, French, and Italian. It is important that NLP solutions adopted by Taskbase can handle different languages, or even a combination of languages in the same task.

The feedback model in the Taskbase Platform is similar but not identical to the task-response-hypothesis model introduced in section 2.1.3 on page 10.

This Masters project was devised directly from Taskbase's use case in researching a "universal AI" that can handle natural language responses to tasks in a variety of domains. This project helps Taskbase answer the following questions:

- How do state-of-the-art NLI models behave?
- What are the shortcomings of current NLI models on mono- and multilingual corpora?
- Can current NLI models be used directly for feedback assignment in the real world?
- How can current NLI models be improved so that they perform better in a digital learning setting?

For reasons of confidentiality and data privacy, this report will include no original data submitted by users on the Taskbase Platform.



Climate change

What do you know about climate change? Summarize in max. 30 words.

Climate change causes weather patterns everywhere to be disrupted.

Climate change causes weather patterns everywhere to be disrupted.

✓ The answer is true because climate change has been proven to cause disruptions in weather patterns.

What do you know about climate change? Summarize in max. 30 words.

the earth is cooling down

the earth is cooling down

✗ The answer is false because the earth is actually warming up. Climate change is caused by the earth's increasing temperature.

What do you know about climate change? Summarize in max. 30 words.

i dunno but my mom says she doesn't believe in it

i dunno but my mom says she doesn't believe in it

✗ We don't believe that you don't know anything. Try again!

✗ The answer is false because it is not based on any facts.

Figure 3: A modified screenshot of Taskbase's learning platform showing a task about climate change. The bottom two responses below the black line are edited in. They show how relevant feedback is assigned to incorrect responses.

3 Prior work

This section presents prior work and lessons learned in (a) feedback assignment in digital education and (b) Natural Language Inference.

3.1 Feedback

The research of how to give constructive feedback to students is a long-standing effort, which unfortunately has lost steam due to the limitations (of time, effort, and otherwise) of assessing students in a paper-and-pencil setting. Computer-aided instruction and automation of student assessment opened the door to previously-unknown branches of research, such as large-scale feedback assignment and real-time feedback delivery.

Naturally, traditional pen-and-paper methods differ from computer-based methods in how instruction is delivered, monitored, and evaluated, but both still require some mechanism of creating, assigning, and delivering feedback to students. In fact, computer-based methods appear to be preferred among students, with learning outcomes being equal or greater than with pen-and-paper instruction (a fact quickly learned during the recent COVID-19 pandemic). In one example by Singleton [128], primary school students had a similar level of success when computer-based methods were introduced but preferred computers as a learning tool.

However, this work is not about the generalities of digital learning; It is about how to provide constructive feedback to students, and so some knowledge of digital learning tools and their significance in the world is assumed.

The ability to conveniently deliver instruction to larger audiences and incorporate digital tools in tracking/evaluating students' work has outpaced the ability to personalize each student's learning experience. In the majority of learning tools today, closed-ended questions are the norm because they can be evaluated by machine easily.

Prior research in improving the marking and feedback process appear to focus on three areas: constructing, assigning, and delivering feedback. Most literature appears to focus on applying feedback strategies in only a few specific fields: mathematics, reading comprehension, computer science, and engineering. However, we shall assume that the lessons learned apply to all subjects of learning equally; This work does not assume any specific subject.

3.1.1 Constructing feedback

Constructing feedback involves creating and organizing feedback items. Many different types of feedback exist, and not all have equal usefulness and didactical value to students. Feedback creation is also a process, with specific procedures and guidelines which help the feedback creator to design feedback that generates positive emotions from students. Feedback additionally has different characteristics which affect its utility. This subsection also covers the *organization* of feedback into useful schemes that helps instructors recall feedback items during manual feedback assignment and feedback improvement.

Valerie Shute has already explored the properties of formative feedback [125]. These properties are summarized on page 9 of this work. To review, good feedback should be elaborative (i.e. elaborates on the student's response in the context of the given task), and specific (i.e. addresses particular elements of the response, in particular misconceptions).

Shute describes 12 types of feedback, of which a few are most common in digital learning for closed-ended tasks:

Verification Also known as "correct"/"incorrect" feedback. Also can take the form of "X% correct".

Correct response Provides the correct response to a task with little additional information.

Try again If the student's response is incorrect, prompts them to try again. If it is correct, no feedback is given and the student moves on.

Some further points that Shute makes are:

- Feedback items should not be *normative*, that is, they should not compare a student to others. This disadvantages poor performers by demotivating them from performing well on later tasks and from seeking tailored learning paths. However, normalizing a feedback item to the student themself may improve motivation by focusing a student on a challenging topic, attributing future successes to effort, and creating a tailored learning path by highlighting a direction of study [84].

- The effect of feedback is more significant if is provides the correct answer instead of a correct/incorrect label, according to Bangert-Drowns *et al.* [6]. This approach could be combined with automated test construction [113] to create a nearly limitless corpus of feedback, where a student receives a correct answer feedback item to a task, and is then asked a similar question with a different answer to check learning.

Shute also summarizes in a table 31 guidelines of effective feedback construction and delivery [125].

In general, Shute and Rahimi [124] conclude that feedback should be constructed to be (a) not overly complex, (b) relevant and unlikely to be ignored by the student, and (c) elaborative. Black and Wiliam [12] also agree that feedback is most useful when it is related to the student's answer (elaborative and/or specific), and when it is instructive and provides the student with a way forward.

Feedback creation is a time-consuming process, and one which does not always have benefits. Mirmotahari *et al.* [89] and Moons *et al.* [92] claim that the process of creating re-usable feedback items is not time-saving in the short-term, as teachers spend as much time constructing and organizing feedback than is saved with the automating feedback assignment. However, benefits do begin to emerge if feedback is re-used across multiple offerings of the same course.

Moons *et al.* [92] propose a framework for creating and organizing so-called "atomic feedback", that is, feedback items which focus on one thing and one thing only from a student's response. They argue that atomic feedback is re-usable, modular, and organizable into a hierarchy of categories which makes retrieving the proper feedback item easier in by-hand feedback assignment. Atomic feedback has the advantage of being composable and combinable when a response necessitates more than one feedback item. However, the authors focused on tasks in mathematics. It is unclear whether the same strategy works for other subjects, especially writing- or language-oriented ones.

Recently, AI has been applied to automate the process of creating feedback. Bernius *et al.* [10] present a system called *CoFee* which uses a machine learning approach to suggest feedback to open-ended exercises. The authors claim that 85% of its suggestions were accepted by teachers, and 5% were accepted with a minor modification.

Table 4 on the following page summarizes this section's cited literature.

3.1.2 Assigning feedback

Assigning feedback means determining which feedback items are most relevant to a certain student response. This section focuses foremost on the automation of this process.

Much literature in this field focuses on the grading of essays instead of open-ended questions, but since the goal of semantic analysis is similar, papers on automatic essay grading are included.

The first essay-scoring system, Project Essay Grade (PEG) was developed by Ellis B. Page [99], who also gives the lofty claim that his system produced outputs indistinguishable from those of human essay scorers on technical and creative benchmarks. Even earlier, in 1966, he published a popular science

Reference	**Summary**	**Meta-analysis**
Bangert-Drowns *et al.* [6]	Correct-answer feedback is more valuable than correct/incorrect label.	
Bernius *et al.* [10]	AI can assist in feedback creation.	
Black and William [12]	Feedback should be related to the student's answer and instruct a way forward.	
McColskey *et al.* [84]	Feedback should be relative to students' own progress, not other students.	
Mirmotahari *et al.* [89]	Constructing feedback does not immediately save time.	
Moons *et al.* [92]	Feedback should be atomic. Atomic feedback can be organized hierarchically. Constructing feedback does not immediately save time.	
Rodrigues and Oliveira [113]	Creating of exercises and tests can be automated.	
Shute [125]	Feedback should be elaborative and specific.	✓
Shute and Rahimi [124]	Feedback should not be complex, should be relevant, should be unlikely to be ignored, should be elaborative.	✓

Table 4: Summary of literature cited on how to effectively construct feedback items.

article called "The Imminence of Grading Essays by computer" [98], which even described for the first time the possibility of giving tailored feedback to essays! Here is what Arthur Daigon suggested as a tailored feedback item to an essay, re-printed in Page's article:

> John [we are told that using first names softens criticism], please correct the following misspellings: beleive, recieve. Note the ie, ei problem. You overuse the words interesting, good, nice; then was [sic] repeated six times. Check trite expressions. All of your sentences are of the subject-verb variety and all are declarative. Re construct. Check subject-verb agreement in second paragraph. You had trouble with this in

> your last paper. Title lacking. Do the following related assignment for tomorrow, etc.

Page's early efforts were controversial at first, and not a commercial success. The greatest failure in PEG was not technical, but rather stemmed from the social restrictions and lack of interest in using expensive computers for mere essay-grading. Page's work also failed to spark any technological revolution in NLP, but he continued to work in this field for many years. Page's later experiments in 1995, based on the PEG system, would go on to supposedly become "more reliable than a 6-judge panel" [146].

Whittington and Hunt [146] provide a review of methods for assessing "free-text" or "free-form" responses in 1999 or earlier, including Page's PEG, Latent Semantic Analysis [69], similarity scores, and grammar-parsing techniques.

As early as 1998, Larkey [70] explored automatic essay grading using primitive machine learning techniques: Bayesian independence classifiers, k-nearest neighbour classifiers, and linear regression. The next few years would produce similar papers in automated grading of free-form answers, many claiming that automated essay grading produces similar accuracy to human correctors [18, 19, 41, 83, 90, 117, 118, 131, 135] (see Table 5 on page 26).

More advanced techniques came later. Noorbehbahani and Karden in 2011 [96] proposed a modified BLEU algorithm [100] for free text assessment. Rodriguez and Oliviera [113] in 2014 proposed "a system for formative assessment... of students' progress" which automatically creates "practice" exams based on questions from previous exams. These questions can be open-ended, and are matched to reference answers by classical syntactic and semantic similarity. He, Hui, and Quan [51] proposed ensemble methods of previously-seen systems.

Approaches to grading short-answer questions that explicitly mention the NLI task or entailment-based methods appeared around 2007. Even prior work has been moving (perhaps unknowingly) towards NLI. Harabaigu, Hickl, and Lacatsu [49] exemplified an NLI approach to text summarization. In their method, texts are split up into "Semantic Content Units" (SCUs), which represent individual atomic propositions. While not directly related to digital learning, such method may be useful for extracting pieces of knowledge from the premise and hypothesis and comparing the two sets to determine entailment.

Following exactly this thread, Nielsen *et al.*, [95] proposed methods of assessing student answers by breaking down texts into "facets" that can be compared to facets in other texts. Sukkarieh and Stoyanchev [132] built an entailment engine (based on a previous automatic grading system [71]) from classical NLP techniques such as grammar parsing, morphology analysis, and tree matching. Dagan *et al.* cited both of these works in a later review of recognizing textual entailment [33].

In 2009, Mohler and Mihalcea published a review [91] of various NLI-based approaches to short answer grading. They concluded that, at the time, the best approaches used LSA.

What all of these more modern works have in common is that they compare a student text to some reference text provided by the instructor, and providing a grade as feedback, instead of relying on heuristics like PEG. Very little mention is made to assigning personalized feedback, except for Mitchell *et al,* [90], who mention the possibility of matching responses against "specifically invalid" answers, and even somewhat identify the Too Much Information problem (presented later, in Section 5.5.5 on page 50), where some incorrect knowledge in an otherwise correct response nullifies its correctness. However, where there is a correct reference answer, there can be mistake reference answers, and personalized feedback can theoretically be given by entailing with the mistake hypotheses, a realization that was seldom published.

Deep learning approaches to short-text scoring emerged in 2016. The first efforts were based on LSTMs [3, 68, 103, 110]. After Vaswani's *et al.* attention paper [140], transformer-based methods appeared [21, 46, 75, 133, 142]

Other approaches appeared as well, including clustering-based methods [8, 87] and a stacked neural network method [108].

Table 5 on the next page shows a summary of these prior works and the methods used therein.

3.1.3 Delivering feedback

The process of *delivering* feedback follows feedback assignment — after feedback is assigned to a response, there are several facets of delivering it to a student. One of these, perhaps the most obvious one, is the timing of the feedback relative to the response submission, on which this section will focus.

Feedback has two aspects of timing:

Delay The time between when a student submits their response to a task and the time they receive feedback. Feedback delay can be immediate, or delayed by some time (as is the case with paper-and-pencil assignments). The only way to achieve truly immediate feedback is via a personal tutor or an automated system. Feedback delay is not only tied to time of correction — feedback can be artificially delayed for a time period, or until a student has completed a certain number of tasks (or an entire assignment).

The jury is still out on whether immediate or delayed feedback is preferred [125]. Support for immediate feedback argues that errors are immediately

Ref.	Authors	Year	Method
[41]	Foltz *et al.*	1999	LSA
[18]	Burstein *et al.*	2001	Discourse parsing
[90]	Mitchell *et al.*	2002	Knowledge extraction, pattern-matching
[83]	Mason and Grover-Stephensen	2002	Knowledge extraction, tree-matching
[118]	Rudner and Liang	2002	Bayes' theorem
[117]	Rosé *et al.*	2003	Syntactic analysis and Naive Bayes
[131]	Sukkarieh *et al.*	2003	Information extraction and IR
[135]	Thomas *et al.*	2004	LSA
[51]	He *et al.*	2009	Ensemble methods
[96]	Noorbehbahani and Karden	2011	Modified BLEU
[8]	Basu *et al.*	2013	Clustering
[3]	Alikaniotis *et al.*	2016	LSTM
[68]	Kumar *et al.*	2017	LSTM
[110]	Riordan *et al.*	2017	LSTM
[46]	Gong and Yao	2019	Attention
[75]	Liu *et al.*	2019	Attention
[103]	Prabhudesai and Duong	2019	Siamese LSTM
[133]	Sung *et al.*	2019	Attention
[142]	Wang *et al.*	2019	Attention; meta-learning
[21]	Camus and Filighera	2020	Attention
[108]	Rajagede and Hastuti	2021	Stacked neural networks

Table 5: A non-exhaustive summary of works applying NLI or NLI-like techniques to feedback assignment.

corrected and not written to a student's memory. Support for delayed feedback suggests that errors are often forgotten anyway and do not interfere with the formation of correct knowledge once the feedback is received.

A few examples: Lemley *et al.* [72] at Birmingham Young University, in an internal study, found that students receiving immediate feedback perform better on final assessments, but students receiving delayed feedback (by mail) complete courses faster. Dihoff *et al.* [36] found that immediate but not delayed feedback enhances learning. Corral *et al.* [31] on the other hand, found an advantage in delayed and correct-answer type feedbacks. Fyfe *et al.* [42] do not find any advantage between the two across 38 different school classes but hint towards benefits of delayed feedback.

Evidence for the superiority of immediate and delayed feedback is present for both sides, but every study appears to use delayed and immediate timings as independent variables, with little regard for the types of feedback and the types of tasks that are being applied. It is possible that certain feedback types or certain tasks are better-suited for either immediate or delayed feedback [67] — a larger meta-analysis should be conducted to determine this. Overall, literature suggests that immediate feedback may be more palatable to the education community — evidence for immediate feedback appears to be more concrete and well-explained.

Cycle timing Bangert-Drowns *et al.* [6] organized learning into a five-stage cycle, where students are most receptive to feedback at certain stages only. The stages are:

1. The initial state, in which the student is ready to receive a task.
2. A search and retrieval state, in which the student receives a question and recalls information to answer it.
3. A response state, in which a learner formulates a response to the question and formulates an expectation about what the feedback will say.
4. An evaluation state, in which a student has received feedback and evaluates their answer.
5. An adjustment state, in which a student modifies their knowledge and goals based on the evaluation. The adjusted knowledge determine the next initial state.

According to Bangert-Drowns *et al.*, feedback contributes to learning only if delivered “mindfully”, that is, in the correct state in the cycle and with the correct presuppositions. For example, if some feedback is available before the search and retrieval state (e.g. by a question that leaks the answer, or a simple lookup-type task), the student “mindlessly” fills in the answer and does not learn [125]. The feedback should also correspond to the student’s expectations and cognitive needs (e.g. shouldn’t be too trivial, shouldn’t be too complex). The combination of feedback and expectation also affects how the adjustment state works — a student receiving “correct” feedback to a confident answer will feel differently about their studies than a student receiving “incorrect” to an unconfident answer.

Shute and Rahimi [124] stress that feedback should be delivered in “manageable units” that do not overwhelm the learner. This can be difficult to balance in a setting like the “atomic feedback” paper [92] where several atomic feedback items could be relevant, or in the situation of Daigon’s quote on page 23. Depending on the task, there is a balance to be struck between listing as much feedback as possible to avoid iterating on the same task (i.e. the student submits an altered answer over and over again but different feedback items pop up, which can be frustrating), and suppressing certain feedback items to avoid confusing and overwhelming the student.

Shute's table of 31 guidelines [125] also lists several *do*s and *do not*s relating to the delivery of feedback.

Table 6 summarizes this section's cited literature.

Reference	**Summary**	**Meta-analysis**
Corral *et al.* [31]	Delayed feedback enhances learning.	
Bangert-Drowns *et al.* [6]	Feedback must be delivered in appropriate stage of learning.	
Dihoff *et al.* [36]	Immediate feedback enhances learning.	
Fyfe *et al.* [42]	No advantage between immediate and delayed feedback.	
Shute [125]	There is support for both immediate and delayed feedback.	✓
Kulik and Kulik [67]	Formative feedback is better immediate; Summative feedback is better delayed. Best timing depends on features of studies.	✓
Lemley *et al.* [72]	Immediate feedback yields better performance; Delayed feedback yields faster course completion.	
Shute and Rahimi [124]	Feedback should be delivered in manageable units	✓

Table 6: Summary of literature cited on how to effectively deliver feedback to students.

3.2 Language models and NLI

Faithful folk could argue that Natural Language Processing has roots in biblical times, after Yahweh split up humanity's one common language and people suddenly had to find ways to understand each other (King James Bible; Genesis 11:1-9). Shortly afterward, in 1947, the idea of a mechanized translation machine was suggested by Warren Weaver [60]. Weaver, a war researcher, was inspired by advancements in cryptography to "decode" foreign text (specifically Russian documents) into legible English. He also correctly doubted this mechanized method's feasibility due to "semantic difficulties because of multiple meanings, etc.". To the credit of his foresight, the first Russian-to-English translation machine was demonstrated in 1954 [60, 61].

However old NLP might be, the fields of **Natural Language Understanding (NLU)** and **Natural Language Inference (NLI)** are relatively young. NLU is a sub-task of NLP, concerned with teaching machines to truly "understand" the meaning of natural language and its context, beyond restricted syntactic or lexical meanings of language's constituent parts. NLU encompasses several sub-sub-tasks itself, of which NLI is one.

3.2.1 Early NLI

The roots of NLI lie in scientists' efforts to analyze the semantics of natural language. The *syntax* of language was by the end of the 20th century very well understood, in large part thanks to Noam Chomsky's works [24, 25], but it was well-established at the time that semantics were difficult to formalize.

NLI emerged as a popular field of study during the first PASCAL RTE challenge [32]. This challenge presented, for the first time, the general task of recognizing entailment between two texts. The challenge was a success, receiving 17 distinct submissions, and continued for 6 more iterations until 2011[3]. However, NLI was found to be tremendously difficult for machines, with "good" performance producing around 55% to 65% accuracy in the first few challenges[4]. Each iteration of the PASCAL RTE challenge changed the themes of the datasets, with many participants considering subsequent challenges to be "easier" than the year before.

In the earliest efforts to develop NLI [22], systems relied on simple approaches based on word-to-word associations [45], syntax-level analysis [139], knowledge extraction [114], formal logic [79, 80], or combinations thereof[5]. Word-to-word approaches to NLI were augmented by rich word-association datasets such as WordNet [88] and FrameNet [5]. These datasets were complete enough that, despite being composed of a fixed number of hard-coded associations, early NLI models were able to achieve better-than-chance performance by exploiting a wide corpus of word relations and inferring their meaning from a limited textual context.

3.2.2 Machine learning models emerge

At the second RTE challenge (RTE-2), Bos and Markert argued that logical inference techniques have a ceiling of usefulness and presented one of the first machine learning techniques for NLI [14, 15]. Several other machine learning submissions have also appeared during this time [52, 62]. These models worked

[3] `https://tac.nist.gov//data/`

[4] The first three RTE challenges were on 2-way entailment, so this performance was hardly better than chance.

[5] Early RTE challenges and literature in this field seemed to have dropped off the map — many papers are not indexed anymore in journals or archives.

primarily on generating sentence or word embeddings using classical NLP techniques, then classifying using non-neural learning algorithms like SVMs or k-means clustering.

Bowman *et al.* were one of the first to apply deep learning techniques to NLI using a classifier neural network fed by RNN and LSTM RNN networks which generate sentence embeddings [16]. Liu *et al.* [76] and Conneau *et al.* [28], among others, followed with increasingly complex LSTM and convolutional architectures.

Around this time, the concept of attention[6] was introduced [4]. This point began a shift from complex NLP pipelines with distinct elements for features like negation detection, synonymy, antonymy, PoS tagging, etc., to end-to-end deep learning systems. Rocktäschel *et al.* [112] presented a fully-neural, end-to-end LSTM approach to NLI that did not rely on an independent sentence mapping step, and extended that approach with attention. In 2017, Vaswani *et al.* proposed the Transformer architecture for sequence-to-sequence tasks (like, for instance, reading a sentence) based solely on attention instead of recurrent networks, in the highly-cited paper *Attention is all you need* [140]. Since then, the world of language models has undergone a dramatic Transformation[7], with all new state-of-the-art models since then being based on this new architecture.

In 2015, Dai and Le [34] achieved state-of-the-art using a "semi-supervised pre-training" approach in recurrent networks and NLP. In this approach, a language model is trained on a corpus of text in an unsupervised or semi-supervised fashion (by allowing the training algorithm to generate its own labeled examples) before being trained on a smaller set of labeled, domain-specific examples. This finding paved the way for the *pre-train-then-fine-tune* paradigm that is most common today — *pre-train* a language model to a generalized form, then *fine-tune* on a downstream task quickly, since neural network weights do not need to be learned from scratch [40]. The concept of pre-training was subsequently applied in ELMo [101], ULMFiT [57], and OpenAI in GPT-1 [104]. Pre-training, however, is not itself a new concept. It dates back to 2010, when Erhan *et al.* demonstrated that pre-training adds robustness and better generalization capabilities to a deep network architecture [40].

How is pre-training relevant to NLI? A pre-trained General Language Model (GLM) has learned weights to "understand" natural language some degree, in that it can predict the next word or a masked word in a sequence based on contextual cues. Fine-tuning a GLM on a task-specific dataset enables the

[6] Attention in NLI is, roughly, a non-recurrent and somewhat fully-connected mechanism where a neural network learns which words or tokens are most relevant to each other. For example, the word "her" might attend strongly to "girl", but not to "running".

[7] pun intended

model to more quickly learn about the downstream task, since it doesn't need to learn the language anymore. In fact, models can be fine-tuned on almost any downstream task quite cheaply, since the pre-training knowledge *transfers* to the downstream task.

At the end of 2018, Google's BERT [35] was open-sourced. This model would have important consequences in the next few years, as it inspired many similar models such as RoBERTa in 2019 [77], XLM-RoBERTa in 2019 [29], Sentence-BERT in 2019 [109], DistilBERT in 2019 [119], BART in 2020 [73], and DeBERTa in 2021 [50][8]. Google then followed up with T5 [107] and mT5 [149], novel and larger Transformer-based architectures.

The thing that all these models since 2018 have in common is that they are pre-trained on *massive* text corpora collected from digital, print, and spoken sources. In many cases, these datasets are so large that it is physically possible only for large companies with enormous compute power to handle them and train models on them. Section 3.2.3 discusses datasets more closely.

In the beginning, these pre-training text corpora were English-only, although researchers quickly investigated the possibility of multi- and cross-lingual language models by aligning word and sentence representations using paired sentences in various languages. Conneau and Lample at Facebook [27] provide a good summary of this prior work and also propose XLM, a training methodology to produce cross-lingual GLMs that led to the development of XLM-RoBERTa [29], a significant cross-lingual GLM which has been extended for many different tasks. Google has also moved to only providing their BERT model in a multilingual variation[9].

Today, the key in deep language models is "bigger is better." Language model sizes are growing very fast in a very short time (see Figure 6 on page 92) [17, 107, 116], to the point where they are impossible to use without owning (or renting, in the case of cloud computing) specialized hardware. Table 7 on the next page and Figure 6 on page 92 illustrate this growth. This trend was prophesied by Shazeer *et al.* [122] in 2017, who warned the NLP world of the coming of "outrageously large neural networks."[10]

3.2.3 Hand-curated datasets became machine-collected and crowd-sourced

The first few NLI datasets were small-scale. The first dataset introduced in the PASCAL RTE challenge consisted of 1367 pairs. In 2014, the SICK dataset

[8] DeBERTa would go through two more major versions

[9] urlhttps://github.com/google-research/bert/blob/master/multilingual.md

[10] Hopefully we won't run out of superlatives.

Model	**Year**	**# of parameters**
OpenAI GPT-2 [105]	2019	1.5B
Google T5 [107]	2020	11B
OpenAI GPT-3 [17]	2020	175B
BigScience BLOOM [11]	2022	176B
Deepmind [106]	2021	230B
NVIDIA Megatron-Turing NLG [129]	2022	530B
Google GLaM [37]	2021/2022	1.2T

Table 7: A non-exhaustive list of today's latest big language models and their sizes.

was introduced [82], composed of about 10 000 pairs, which were constructed by taking several grammatical variations of English sentences. Deep learning had not yet been introduced to NLI yet, so early NLI datasets were test-only or contained a limited number of examples for training.

In 2015, Bowman *et al.* were the first to create an NLI dataset that makes the leap to the large scale: the Stanford Natural Language Inference (SNLI) corpus [16]. This dataset was groundbreaking in that it was the first dataset of sufficient size to train "data-intensive, wide-coverage" models. All of the texts in this dataset were written by humans, annotated by humans, and collected by machine from the Flickr 30k corpus (a dataset of image captions from the Flickr image-hosting service; [150]). It has, however, been criticized for being made up exclusively of image captions, which makes it ideal for describing scenes but limits its utility in other areas, like understanding conversational language. Nevertheless, SNLI was a major milestone in the development of large-scale datasets.

In 2018, the MultiNLI corpus was developed by Williams, Nangia, and Bowman with the express intent to address the shortcomings of SNLI [147]. MultiNLI (or MNLI) collects sentences from 9 text sources in many formats: face-to-face and telephone conversations, reports, letters, public domain texts from governmental websites, and open-access non-fiction works from print and digital media. Though smaller than SNLI (MultiNLI has 433k examples compared to SNLI's 570k), MultiNLI covers natural language more broadly. It has been so successful that most NLI models published on HuggingFace, including NLI versions of state-of-the-art models such as BERT, DeBERTa, T5, etc., have been fine-tuned on MultiNLI or derivative datasets.

Both SNLI and MultiNLI collect premises from text sources. Hypotheses were collected by presenting a premise to a crowdsourcing worker, who conceives one entailing, one contradicting, and one neutral hypothesis.

In the same year, Conneau *et al.* published the Cross-lingual Natural Language Inference corpus (XNLI) [30]. XNLI extends the MultiNLI corpus by translating it to 15 different languages: English, French, Spanish, German, Greek, Bulgarian, Russian, Turkish, Arabic, Vietnamese, Thai, Chinese, Hindu, Swahili, and Urdu. Texts were translated by professional translators. XNLI allows texts to be written in any of the supported languages, even in multiple languages in a single text. The dataset is published with each premise-hypothesis pair having the same language. Cross-language pairs are not provided but can be constructed by sampling the appropriate text from the 15 languages. The vast majority of cross-language NLI models are trained on XNLI or derivatives.

It is also worthwhile to mention the Cross-lingual TRansfer Evaluation of Multilingual Encoders (XTREME) corpus, published by Hu *et al.* in 2020 [59]. XTREME is a broad, multilingual benchmark consisting of several different tasks, including (a subset of) XNLI. It is most often used to fine-tune and validate general or multi-task cross-language LMs.

A drawback of current NLI datasets is that they do not represent the space of NLI texts found in digital learning. Both MultiNLI and SNLI consist of premise texts collected from open corpora. While SNLI collected them from image captions, MultiNLI collected from various sources, with the intention of generalizing well over the English language. On the other hand, NLI for digital learning deals with texts which are responses to questions — these texts typically present one or several facts and should not contain bias from “colloquial” or “conversational” language such as slang or missing punctuation. For example, this NLI pair appears in the MultiNLI dataset which, admittedly would be useful in building a general language processing system, does not provide much value in as an example in digital learning:

p: yeah it’s a nice way to relax i mean in a way i mean i find it
anyway although sometimes watching the news isn’t very
relaxing i get home from from (T 3.1)
h: Watching the news isn’t always relaxing.

The most recent major NLI dataset is Adversarial NLI (ANLI), published by Nie *et al.* in 2021 [94]. ANLI is unique in that its examples were constructed by hand to be adversarial. That is, human annotators were given a premise text and tasked with making a hypothesis that fools the language model into making an incorrect prediction. It is also unique in its “Human-And-Model-in-the-Loop Enabled Training”, in which human writers created NLI pairs with constant feedback from a chosen NLI model, as well as feedback from human verifiers. This process took place in 3 rounds. In each round, adversarial examples are written with feedback from the model, the examples are verified by humans, and a new model is trained using the adversarial examples. As the model learns, each round consists of more and more difficult examples.

ANLI appears to be the most effective dataset for training robust NLI models. The authors claim that models trained on ANLI are state-of-the-art on several existing NLI benchmarks.

Table 8 shows a summary of each NLI dataset.

Dataset name	**Year**	**Size**	**2-way/3-way**
SNLI [16]	2015	570k	3-way
MultiNLI [147]	2018	433k	3-way
XNLI [30]	2018	400k × 15 languages	3-way
ANLI [94]	2021	170k over 3 rounds	3-way

Table 8: Summary of open NLI datasets. All of these datasets are freely available.

Domain-specific NLI datasets also exist. Specifically, the study of law and legal text processing is a major field of research, including some research on applying NLI to those problems. Two examples on using NLI on legal texts are COLIEE [63] and ContractNLI [66]. NLI is also used in the medical field, for example in the MedNLI dataset [115, 123] containing patient histories. A dataset also exists for evaluating science knowledge in schools [65]. Unfortunately, these domain-specific are still niche — most NLI evaluation is done on general-purpose datasets such as SNLI or MultiNLI, which several researchers lament as being an insufficient benchmark for most real-world NLI work and giving very little regard to domain-specific applications [102, 145].

4 Goals and research questions

4.1 Goal

The goal of this thesis is to obtain an understanding of the intimate workings of NLI models for the purpose of feedback assignment in digital education, specifically, on datasets from the Taskbase Platform. It seeks to use these findings to devise a framework of: themes that must be considered when creating NLI-friendly tasks; techniques that can be used to adapt existing tasks to be more NLI-friendly; and possible solutions to major pitfalls that NLI models display.

4.2 Research questions

This thesis aims to answer the following questions:

- How do various NLI models perform on NLI corpora? What are the contributions of model architecture and fine-tuning datasets to performance?
- What are the shortcomings of current NLI models on multilingual, open-ended tasks in the setting of digital learning?
- How can the NLI task be modified or augmented to perform better in digital learning tasks?
- How can digital learning tasks be adapted in order to better take advantage of the capabilities of NLI?
- Does NLI exhibit certain emergent behaviours that can be exploited for digital learning?

5 Characterizing NLI models

Many different experiments were conducted to determine how current NLI models behave on freely-available NLI datasets and datasets from the Taskbase Platform. First, a benchmark was performed over the widest range of models and datasets. Later experiments were focused on examining a particular property or technique. When an experiment revealed some interesting behaviour, it was explored further in each experiment's discussion section.

5.1 Experimental setup

5.1.1 Libraries and infrastructure

This work used Python[11] versions 3.7.11 and 3.10.5. Major libraries used were Numpy[12], Scikit Learn[13], Pandas[14], spaCy[15], PyTorch[16], Huggingface Transformers[17], Huggingface Datasets[18], Matplotlib[19], Seaborn[20]. NLI models and datasets were obtained from Huggingface[21]. Inference was performed on an AWS EC2 `g4dn.xlarge` instance with a single NVIDIA T4 GPU and 12 GB of VRAM.

5.1.2 Datasets

Datasets are in English unless otherwise specified. Twelve NLI datasets were used throughout this work:

`Taskbase SimpleK` A dataset from Taskbase's corpus, in which both premises and hypotheses are keywords or sentence fragments. `Taskbase SimpleK` consists of approximately 65 entailing pairs and 1325 non-entailing pairs. The dataset is unbalanced because the list of non-entailing pairs was constructed roughly by taking the Cartesian product of a set of premises and non-entailed hypotheses.

`Taskbase Buyer-Seller` A dataset from Taskbase's corpus, containing responses and hypotheses from a single task: *What are the responsibilities of the*

[11] https://www.python.org/
[12] https://numpy.org/
[13] https://scikit-learn.org/
[14] https://pandas.pydata.org/
[15] https://spacy.io/
[16] https://pytorch.org/
[17] https://huggingface.co/docs/transformers/index
[18] https://huggingface.co/docs/datasets/index
[19] https://matplotlib.org/
[20] https://seaborn.pydata.org/
[21] https://huggingface.co/

buyer and seller in a transaction? Premises and hypotheses are all in sentence format and share similar vocabulary. This dataset contains 135 entailing pairs and 18 non-entailing pairs.

`Taskbase Evil Regular DE` A dataset of challenging ("evil") examples from Taskbase's corpus, all in German. It contains 4695 entailing pairs and 44306 entailing pairs from 34 different tasks. In the dataset's source, each task includes a matrix M of premises on one axis and hypothesis on the other axis. For every premise p and hypothesis h, $M_{p,h} =$ ENTAILMENT if p and h entail, and NOT ENTAILMENT otherwise. The dataset was constructed using the Cartesian product of each matrix, resulting in $\langle \text{premise}, \text{hypothesis}, \text{entailment} \rangle$ tuples. The dataset is imbalanced because the task matrices are sparse.

`Taskbase Evil Regular EN` A dataset constructed from `Taskbase Evil Regular DE` by machine-translating every premise and hypothesis using DeepL[22].

`Taskbase Evil Hard` An "extra-evil" dataset from Taskbase's corpus containing hand-picked adversarial examples and vocabulary designed to fool NLI models, consisting of 340 entailing pairs and 268 non-entailing pairs. The format of the pairs are sentence fragments and full sentences. This dataset is in German.

`SNLI` Derived from the test split of the SNLI corpus [16] from Huggingface[23]. Examples with no gold label were dropped. There are 3368 entailing pairs and 6456 non-entailing pairs. The dataset is balanced for 3-way entailment, but not 2-way.

`MNLI validation combined` Derived from the MultiNLI corpus [147] by concatenating the matched validation and mismatched validation splits from Huggingface[24]. Examples with no gold label were dropped. There are 6942 entailing pairs and 12705 non-entailing pairs. The dataset is balanced for 3-way entailment, but not 2-way.

`XNLI` A truncated version of the XNLI test split [30] from Huggingface[25] containing only examples in German, English, Spanish, and French. Each premise-hypothesis pair is the same language, and each pair is repeated 4 times, once for every language There are 6680 entailing pairs and 13360 non-entailing pairs. The dataset is balanced for 3-way entailment, but not 2-way.

[22] `https://www.deepl.com/translator`

[23] `https://huggingface.co/datasets/snli`

[24] `https://huggingface.co/datasets/multi_nli`

[25] `https://huggingface.co/datasets/xnli`

`XNLI shuffled` A mutation of the `XNLI` dataset. It has the same size, except the premise and hypothesis languages were sampled from the list of allowed languages (German, English, Spanish, French). For each distinct text pair, there are 4 variations of it where the pair's premise and hypothesis may have different languages.

`ANLI R`{`1,2,3`} Three datasets, imported directly from the test R1, test R2, and test R3 splits of the ANLI corpus [94] from Huggingface[26]. Each split has 334 to 402 entailing pairs, and 666 to 798 non-entailing examples. The dataset is balanced for 3-way entailment, but not 2-way.

A quick note: when a dataset (or model) appears in `fixed width type`, it refers to the dataset or model adapted for this work. When a dataset or model appears as normal text, it refers to the dataset or model as described in its original paper. For example, a model may be fine-tuned on XNLI, but tested on `XNLI`, which is the specific subset of XNLI used in this work.

5.1.3 Models

The experiments in this section use up to seven NLI models chosen for this work. Models were selected to best cover the different state-of-the-art neural network architectures since BERT (2019; [35]), as well as the various NLI datasets since SNLI (2015). Only transformer-based models [140] were chosen since these produce better performance than sentence embedding-based models or LSTMs [35].

Seven open-access NLI models were used, each obtained from Huggingface. To avoid having to write the full names of each model, they were given short names which will be used throughout this work:

`AT-mT5` Based on Google's multilingual mT5 architecture [149] and trained on MultiNLI and XTREME XNLI [59] datasets by the Alan Turing Institute[27]. This model was the the first model to be explored. Other models were added thereafter. Freely available[28].

`RoBERTa` Based on Facebook's RoBERTa Large architecture [77] general language model and fine-tuned on the MultiNLI dataset. English-only. Freely available[29].

[26] `https://huggingface.co/datasets/anli`

[27] `https://www.turing.ac.uk/`

[28] `https://huggingface.co/alan-turing-institute/mt5-large-finetuned-mnli-xtreme-xnli`

[29] `https://huggingface.co/roberta-large-mnli`

`ML mDeBERTa` Based on Microsoft's DeBERTa v3-base architecture [50] and fine-tuned on the MultiNLI and XNLI datasets by Huggingface user `MoritzLaurer`. Freely available[30]

`RoBERTa LXA` Based on Conneau's *et al.* XLM-RoBERTa architecture and fine-tuned on the XNLI and ANLI datasets by Huggingface user `vicgalle` (`LXA` = **L**arge, **X**NLI, **A**NLI). Freely available[31].

`RoBERTa LX` Based on Conneau's *et al.* XLM-RoBERTa architecture and fine-tuned on the XNLI dataset by Huggingface user `joedav` (`LX` = **L**arge, **X**NLI). Freely available[32].

`ML DeBERTa MFA` Based on Microsoft's DeBERTa v3-base architecture and fine-tuned on the MultiNLI, FEVER [136], and ANLI datasets by Huggingface user `MoritzLaurer` (**MFA** = **M**ultiNLI, **F**EVER, **A**NLI). Freely available[33].

`RoBERTa Ynie` Based on Conneau's *et al.* XLM-RoBERTa architecture and fine-tuned on the SNLI, MultiNLI, FEVER, and ANLI datasets by Yixin Nie, one of the authors of ANLI [94], also known by his Huggingface handle `ynie`. Freely available[34].

Huggingface supports ready-made "pipelines" for many tasks, e.g. text classification, but not all models are supported. Instead, inference using these models was invoked manually and the output logits analyzed directly.

All of these models are 3-way entailment models. For the purposes of this work, they were converted to 2-way by merging the NEUTRAL and CONTRADICTION outcomes into a single NOT ENTAILMENT outcome.

5.1.4 Statistics

In quantitative experiments, entailment is inferred for examples in some NLI dataset and the statistics in Table 9 on the next page are recorded. These are the "standard statistics".

When determining p-values, statistic means are compared for significance using a simple Z test. Strictly speaking, the binomial test is more correct (since ENTAILMENT-NOT ENTAILMENT is effectively an unbalanced Bernoulli trial), but sample sizes of NLI datasets are large enough to make the Binomial test impractical and to justify a close approximation.

[30] `https://huggingface.co/MoritzLaurer/mDeBERTa-v3-base-mnli-xnli`

[31] `https://huggingface.co/vicgalle/xlm-roberta-large-xnli-anli`

[32] `https://huggingface.co/joeddav/xlm-roberta-large-xnli`

[33] `https://huggingface.co/MoritzLaurer/DeBERTa-v3-base-mnli-fever-anli`

[34] `https://huggingface.co/ynie/roberta-large-snli_mnli_fever_anli_R1_R2_R3-nli`

Stat.	**Definition**
Acc.	Accuracy. Fraction of examples whose predictions match their labels.
Prec. E	Precision of the group labeled ENTAILMENT, i.e. Of the examples predicted as ENTAILMENT, what is the chance that one is truly ENTAILMENT?
Rec. E	Recall of the group labeled ENTAILMENT, i.e. when presented with an example labeled ENTAILMENT, what is the chance that the NLI model will predict ENTAILMENT?
F_1 E	F_1 score of the group labeled ENTAILMENT.
Prec. NE	Precision of the group labeled NOT ENTAILMENT (similar interpretation as above).
Rec. NE	Recall of the group labeled NOT ENTAILMENT (similar interpretation as above).
F_1 NE	F_1 score of the group labeled NOT ENTAILMENT.

Table 9: Summary of statistics collected in benchmarking experiments.

5.2 Benchmarking state-of-the-art language models

The first step of characterizing NLI models to to measure their performance on some datasets. 7 freely-available NLI models were benchmarked on 12 datasets[35], of which 5 come from Taskbase and 7 are open datasets. These models were chosen to represent many different architectures and fine-tuning schemes (a model may be fine-tuned on several datasets).

The goal of this benchmarking experiment is to get an initial overview of which models work best for which datasets, and to discover whether certain models or certain fine-tuning methods have an advantage over others. Close attention is paid to how these models behave on Taskbase's datasets.

5.2.1 Data collection

Twelve datasets were used: `Taskbase SimpleK`, `Taskbase Buyer-Seller`, `Taskbase Evil Regular EN`, `Taskbase Evil Regular DE`, `Taskbase Evil Hard`, `MNLI validation combined`, `SNLI`, `XNLI`, `XNLI Shuffled`, and `ANLI R{1,2,3}`.

[35] The `ML DeBERTa MFA` and `RoBERTa Ynie` were added later in the course of this project and do not appear in other experiments.

5.2.2 Method

Seven NLI models were used: `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `ML DeBERTa MFA`, `RoBERTa LXA`, `RoBERTa LX`, and `RoBERTa Ynie`. Each one was benchmarked against all datasets without modification. There were 172 904 $\langle p, h \rangle$ pairs in total.

Standard statistics were collected.

5.2.3 Results

1 210 328 predictions were collected (172 904 pairs × 7 models). Data were aggregated by model and dataset, and are presented in table form (Table 21 on page 116) and heatmap form (Figure 8 on page 113, Figure 9 on page 114, and Figure 10 on page 115) The three heatmaps show the same data but sliced on different axes, for better interpretability.

5.2.4 Discussion

Immediately, it is evident that there are some datasets where all models perform well. `MNLI`, `SNLI`, and both `XNLI` datasets yielded very good results. This is expected, since most models were trained on these or similar datasets. The exceptions are `RoBERTa` and `RoBERTa Ynie`, which perform poorly on XNLI because they were not trained on it and only understand English.

The SimpleK dataset is unbalanced towards not-entailing examples by about a factor of 10. Therefore, all models have good NOT ENTAILMENT precision and recall. Most models also have good ENTAILMENT recall, suggesting that these models recognize entailing cases better, but sadly, low ENTAILMENT precision suggests that it likely that these models bias predictions towards ENTAILMENT instead. The exceptions to this rule are the two `DeBERTa` models, which have low ENTAILMENT precision *and* recall.

`ML DeBERTa MFA` has a chance to redeem itself, along with `RoBERTa Ynie` on the `Taskbase Buyer-Seller` dataset, where these two models are the strongest performers. This dataset is unbalanced favouring entailing examples. The commonality that these models have is that they were trained on the FEVER dataset.

The two `Taskbase Evil Regular` datasets are challenging because not only are they extremely unbalanced favouring NOT ENTAILMENT examples, but their examples are dirty and extremely variable in structure (e.g. sentence structure, full sentences versus fragments, punctuation, incomplete responses, etc.). No models have good ENTAILMENT recall except `RoBERTa LX` and `RoBERTa LXA` close behind it. However, ENTAILMENT precision and ENTAILMENT F_1 are universally bad.

All multilingual models perform mildly well on the `Taskbase Evil Hard` dataset. The best performers are `RoBERTa LXA` and `RoBERTa LX`. Both models are comparable, with most scores between 0.75 and 0.95, but the former biases more strongly towards NOT ENTAILMENT than the latter. The three `ANLI` datasets show the same pattern, with `ML DeBERTa MFA`, `RoBERTa LXA`, and `RoBERTa Ynie` performing well since they were trained on ANLI.

In an effort to find the "best" model for the five Taskbase datasets, Figure 4 shows the number of times each model achieved a maximum statistic for a single dataset. No single model is best for accuracy. `RoBERTa LX` displays a precision-recall tradeof and likely biases predictions towards ENTAILMENT. `ML DeBERTa MFA` and `RoBERTa LXA` show a slight increase in NOT ENTAILMENT recall, and potentially a precision-recall tradeoff in the opposite direction. Overall, `AT-mT5` achieved the most maximal F_1 scores out of all models.

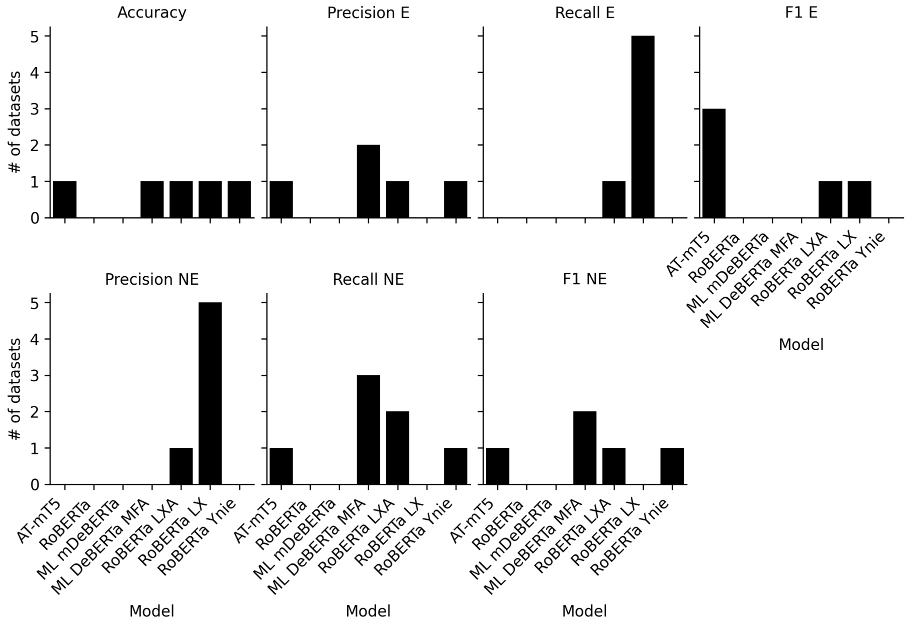


Figure 4: Summary of model performance for each statistic. Each plot shows, for one statistic, the number of Taskbase datasets on which a model achieved maximum performance across all models. The values of some plots may add up to greater than 5 (the number of datasets) if more than 1 model achieved the same maximum.

5.2.5 Wrapping up

No one particular model performs well on all Taskbase datasets. There is evidence of precision-recall tradeoffs in a few models but no single model appears to be more clever than the others on all datasets.

There is some evidence to suggest that the choice of fine-tuning dataset makes a significant difference. Two models trained on FEVER stood out from the rest in the `Buyer-Seller` dataset. However, some evidence is in favour of architecture choice too — the two `RoBERTa` models trained on XNLI did better on `Taskbase Evil Hard`. `RoBERTa` did not, since it was not trained on German.

To have a more precise overview of model performance, and whether they are biased more towards ENTAILMENT or NOT ENTAILMENT, the unbalanced datasets should be re-constructed to be better balanced, or the calculations should be re-done to put greater weight on underrepresented classes.

5.3 Full stops

In open NLI datasets as well as Taskbase datasets, many texts had inconsistent or missing punctuation. This was especially the case with massive or machine-constructed datasets (such as MNLI, where premises were taken from crawled texts and hypotheses were crowdsourced), and texts for tasks which could be answered in keyword form. Various keyword and full-sentence texts were tried with and without full stops at the end, which showed that some clearly entailing pairs were misclassified when a full stop was not present. This finding prompted a larger-scale experiment over several whole datasets where full stops were synthetically added or removed.

5.3.1 Hypothesis

The goal of this experiment is to discover whether full stops at the end of sentences affect model performance.

H_0 Adding full stops at the end of sentences produces no change in performance compared to texts with no full stops.

H_+ Adding full stops at the end of sentences increases performance compared to texts with no full stops.

H_- Adding full stops at the end of sentences decreases performance compared to texts with no full stops.

5.3.2 Data collection

Taskbase's and open datasets were used: `Taskbase SimpleK`, `Taskbase Buyer-Seller`, `Taskbase Evil Regular EN`, `Taskbase Evil Regular DE`, `Taskbase Evil Hard`, `SNLI`, `MNLI combined validation`, `XNLI`, and `XNLI Shuffled`. Each dataset was then processed by a "full-stopifier", which ensured the presence or absence of a full-stop at the end of each premise p or hypothesis h (or both). For each dataset, 4 variations were produced: without full stops on both p and h (the **null variation**), full stops on p only, full stops on h only, and full stops on both p and h.

5.3.3 Method

Models used were `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, and `RoBERTa LX`. The highest-likelihood prediction was recorded. Each variation of each dataset was tested on all models, for a total of 180 runs. All models implement 3-way entailment; results were condensed to 2-way entailment for analysis. Descriptive statistics, precision, recall, and F_1 scores were recorded assuming 2-way NLI.

5.3.4 Results

Table 22 on page 120 shows statistics for each model, dataset, and variation. Table 25 on page 140 shows the p-values.

Significance was calculated on all statistics except F_1 scores by means of a Z-test using the SEM of a Bernoulli distribution. Dataset sizes were large enough (hundreds to tens of thousands of examples) that a Z-test was sufficient over the binomial test of significance.

Table 10 on the next page shows the complete descriptive statistics. Extrema of effect sizes are large, ranging from -59.25% to 83.33%. The most dramatic effect sizes seem isolated to the `ML mDeBERTa` model. Effect sizes using other models generally range from fractions of a percent to a few percent, with an occasional effect size of up to ±9.5%. Most effect sizes range from -4.67% to +6.94% over the null variation. Both mean and median effect sizes for all statistics are mild.

5.3.5 Discussion

Certain patterns are visible in the results.

Precision-recall tradeoff First, many cases display a tradeoff between precision and recall. In most of these, full-stopping the hypothesis or both texts

	count+	count-	mean	min	10%	50%	90%	max
Accuracy	24	23	-0.0028	-0.0936	-0.0380	0.0034	0.0262	0.1207
Precision E	15	30	-0.0277	-0.5926	-0.0735	-0.0116	0.0470	0.2366
Recall E	44	10	0.0733	-0.2093	-0.0659	0.0345	0.1383	0.8333
Precision NE	38	6	0.0144	-0.0095	-0.0025	0.0079	0.0156	0.2414
Recall NE	17	43	-0.0120	-0.1149	-0.0467	-0.0085	0.0199	0.0862
Overall	138	112	0.0100	-0.5926	-0.0467	0.0042	0.0694	0.8333

Table 10: Descriptive statistics of significant effect sizes for the full stops experiment. Values are given as relative change of the statistic over the null variation. *count+* indicates the number of times a statistic increased over all models and datasets. *count-* shows the same for decreases.

leads the model to predict ENTAILMENT with greater likelihood, increasing ENTAILMENT recall and NOT ENTAILMENT precision but decreasing ENTAILMENT precision and NOT ENTAILMENT recall. However, a mixed effect can be observed in a some other instances, for example, with `RoBERTa LX` and the `Taskbase Evil Regular` datasets, among others.

With the models `AT-mT5` and `ML mDeBERTa`, adding full stops to hypotheses on several challenging sentences almost universally improves ENTAILMENT recall, i.e. entailing sentence pairs were more likely to be correctly classified when the hypothesis had a full stop.

Accuracy When accuracy is significantly influenced, it decreases in about half the cases and increases in the rest. There appears to be a correlation between the direction of change in accuracy an the precision-recall tradeoff described above, but whether it is a direct or inverse correlation depends on the model. With the three `RoBERTa` models, accuracy and ENTAILMENT precision are directly correlated, but `AT-mT5` and `ML mDeBERTa` show an inverse correlation.

Language effect and not-understood texts `RoBERTa` tends towards NOT ENTAILMENT on the `Taskbase Evil Regular DE` dataset when full-stopping either or both texts, despite not being trained on German. A similar phenomenon appears on the `XNLI` dataset. This is unsurprising, since the model has no way of understanding non-English languages and will most likely tend to NEUTRAL. The accuracy also increases significantly for the `Taskbase Evil Regular DE` dataset (+3.2% to +6.8%) since the dataset is composed mostly of non-entailing pairs. For all other models (which are multilingual), there is no dramatic difference in the effect of full stops between English and German datasets.

`RoBERTa LXA` is neutral The `RoBERTa LXA` model is quite neutral with respect to full stops. Only a few times did a statistic significantly increase over the null variation; there were no decreases. `RoBERTa LXA` is unique among the models used that it was trained on the ANLI dataset, warranting future experiments with ANLI-trained models on challenging texts. While `RoBERTa LXA` was also trained on XNLI, which could explain its affinity for non-English datasets, `RoBERTa LX` was also trained on XNLI and does not display the neutral effect.

Wrapping up Overall, adding full stops has a mild positive effect, but drives predictions slightly towards NOT ENTAILMENT. There are cases, which shall be explored in Section 5.7, where full stops are tremendously useful.

Whether full stops have a positive or negative effect for a specific model depends mostly on the model, somewhat on the dataset, as well as the location of the full stop (i.e. premise, hypothesis, both).

5.4 Capitalization

In a similar vein to the full stops experiment, this experiment addresses the inconsistent capitalization of text pairs and seeks to discover whether there is any significantly different behaviour when the texts' capitalization is normalized.

5.4.1 Hypothesis

The goal of this experiment is to discover whether capitalization of sentences affects model performance. A capitalized sentence has its first letter uppercased; a non-capitalized sentence has its first letter lowercased but may contain uppercased letters within.

H_0 Capitalizing hypothesis or premise produces no change in performance over uncapitalized texts.

H_+ Capitalizing hypothesis or premise increases performance compared to uncapitalized texts.

H_- Capitalizing hypothesis or premise decreases performance compared to uncapitalized texts.

5.4.2 Data collection

The following datasets were used: `Taskbase SimpleK`, `Taskbase Buyer-Seller`, `Taskbase Evil Regular EN` and `DE`, `Taskbase Evil Hard`, `MNLI validation combined`, `SNLI`, `XNLI`, and `XNLI shuffled`.

From these datasets, four different capitalization variations were constructed:

none The "none" variation has both premise and hypothesis uncapitalized. This will be referred to as the **null variation**.

hypothesis The "hypothesis" variation has the hypothesis capitalized and premise uncapitalized.

premise The "premise" variation has the hypothesis uncapitalized and premise capitalized.

both The "both" variation has both texts capitalized.

5.4.3 Method

Models used were `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, and `RoBERTa LX`. The highest-likelihood prediction was recorded. Each variation of each dataset was tested on all models, for a total of 180 runs. All models implement 3-way entailment; results were condensed to 2-way entailment for analysis. Descriptive statistics, precision, recall, and F_1 scores were recorded assuming 2-way NLI.

5.4.4 Results

Table 24 on page 133 shows statistics for each model, dataset, and variation. Table 25 on page 140 shows the p-values.

Significance was calculated on all statistics except F_1 scores by means of a Z-test using the SEM of a Bernoulli distribution. Dataset sizes were large enough (hundreds to tens of thousands of examples) that a Z-test was sufficient over the binomial test of significance.

Table 11 on the following page shows the complete descriptive statistics. Effect sizes are small, ranging from fractions of a percent to a couple percent for large datasets. Effect sizes are more pronounced for the `SimpleK` dataset, to which `RoBERTa` and `ML mDeBERTa` are rather sensitive, but for a large part, effect sizes are very mild over the null variation. Most effect sizes range from -4.56% to 2.27%.

The mean and median effect sizes were both negative. There are also 90 instances of a statistic significantly decreasing, compared to 63 increasing. Conversely, ENTAILMENT recall and NOT ENTAILMENT precision both had more positive effects than negative effects, suggesting that there is a bias towards ENTAILMENTby capitalizing texts.

5.4.5 Discussion

From these results, a few patterns are visible.

	count+	count-	mean	min	10%	50%	90%	max
Accuracy	13	24	-0.0109	-0.1253	-0.0239	-0.0054	0.0059	0.0578
Precision E	4	21	-0.0515	-0.2705	-0.1793	-0.0344	0.0124	0.1719
Recall E	22	9	0.0230	-0.0551	-0.0232	0.0062	0.0952	0.2069
Precision NE	13	7	0.0191	-0.0026	-0.0018	0.0014	0.0708	0.2000
Recall NE	11	29	-0.0142	-0.1325	-0.0277	-0.0051	0.0063	0.0090
Overall	63	90	-0.0076	-0.2705	-0.0456	-0.0032	0.0227	0.2069

Table 11: Descriptive statistics of significant effect sizes for the full stops experiment. Values are given as relative change of the statistic over the null variation. *count+* indicates the number of times a statistic increased over all models and datasets. *count-* shows the same for decreases.

Precision-recall tradeoff Like the "full stops" experiment, there is a precision-recall tradeoff when capitalizing texts. Overall, capitalization tends to harm accuracy, ENTAILMENT precision, and NOT ENTAILMENT recall; while increasing ENTAILMENT recall and NOT ENTAILMENT precision. The trade-off is most pronounced on the `Taskbase Evil Regular` datasets.

`Roberta LXA` appears to be the least sensitive to capitalization, with only 14 significant changes compared to the null variation.

The tradeoff indicates that capitalization tends to bias predictions towards ENTAILMENT, with a few exceptions: the `RoBERTa LX` model did better on the `XNLI Shuffled` dataset when capitalizing the hypothesis; `ML mDeBERTa` displays opposite trade-offs on both `Taskbase Evil` and `SimpleK datasets`, and `AT-mT5` displays opposite trade-offs on `Taskbase Evil Regular EN` and `DE` datasets when capitalizing the hypothesis. The latter case may be evidence of a language effect in `AT-mT5`,

Language Effect A property of the German language is that all nouns are capitalized. Recall that the null variation consists of texts which are *not* capitalized. When a noun appears at the beginning of the premise or hypothesis, it would be correctly capitalized, possible appearing as more correct to the NLI model, increasing accuracy. AT-mT5 is the only model that displays this kind of effect, with better performance on `Taskbase Evil Regular DE` when capitalizing the hypothesis, while the English version of the dataset displayed worse performance.

Wrapping up Whether H_0 holds or not depends somewhat on the model, and highly on the nature of the dataset and which of the two texts is capitalized. Overall, capitalizing either the premise or hypothesis texts leads to reduced accuracy and a significant shift towards ENTAILMENT in predictions.

5.5 Bidirectional entailment

Bidirectional entailment is a technique where entailment between the NLI premise p and hypothesis h is inferred in both directions, i.e. whether p entails/is compatible with/contradicts h *and* vice versa. The result of (3-way) bidirectional entailment is a pair of outcomes:

$$\text{NLI}_{\text{bidi}} \mapsto \{\textsc{entailment}, \textsc{neutral}, \textsc{contradiction}\}^2$$

where the first element of the tuple is the outcome in the *forward* direction (i.e. does p entil h?) and the second element is the outcome in the *backward* direction (i.e. does h entail p?).

Bidirectional entailment can be useful to determine **equivalence** of two statements. If p and h both contain the same knowledge, they will entail in both directions. Call this relation $A \Leftrightarrow B$, where $A \vDash B \wedge B \vDash A$. In the context of a digital learning platform, a student response being equivalent to a task hypothesis is a stronger condition than entailment. Equivalence can be a useful tool for filtering responses that contain out-of-hypothesis incorrect knowledge, or possibly for enforcing a certain vocabulary.

5.5.1 Hypothesis

The goal of this experiment is to qualitatively determine whether bidirectional entailment is a good tool for testing equality of a premise and hypothesis.

This is a qualitative and explorative experiment; there is no hypothesis.

5.5.2 Data collection

Taskbase's and open datasets were used: `Taskbase SimpleK`, `Taskbase Buyer-Seller`, `Taskbase Evil Regular EN`, `Taskbase Evil Regular DE`, `Taskbase Evil Hard`, `SNLI`, `MNLI combined validation`, `XNLI`, and `XNLI Shuffled`.

5.5.3 Method

5 NLI models were used: `AT-MT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, and `RoBERTa LX`. For each model, dataset, and example, bidirectional entailment was performed on $\langle p, h \rangle$ to get predictions in the forward and backward directions. Results were examined by hand to find failure cases.

5.5.4 Results

Table 12 on the next page summarizes conditional probabilities of backward direction predictions given a forward direction prediction (e.g. $P(\text{backward} =$

$E \mid \text{forward} = N$) over the 5 models used. Table 13 on the following page shows the same, only grouped by each dataset.

Table 26 on page 146 summarizes these probabilities over all models and datasets.

		P(E|F)	P(N|F)	P(C|F)
Model	F			
AT-mT5	E	0.2349	0.7029	**0.0622**
	N	0.0761	0.7690	0.1549
	C	**0.0225**	0.3723	0.6053
RoBERTa	E	0.2340	0.6135	**0.1526**
	N	0.1386	0.6603	0.2011
	C	**0.0752**	0.3267	0.5980
ML mDeBERTa	E	0.2981	0.6190	**0.0829**
	N	0.0924	0.7938	0.1137
	C	**0.0390**	0.3324	0.6286
RoBERTa LXA	E	0.3324	0.5772	**0.0905**
	N	0.2054	0.6698	0.1248
	C	**0.0795**	0.3083	0.6122
RoBERTa LX	E	0.3030	0.6282	**0.0687**
	N	0.1568	0.6983	0.1449
	C	**0.0432**	0.3438	0.6130

Table 12: Summary of probabilities of backward entailment predictions, grouped by model. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F. Bold values are "interesting" cases.

5.5.5 Discussion

There are 9 combinations of forward-backward predictions. Of these, only 2 are interesting: forward = ENTAILMENT $\wedge$ backward = CONTRADICTION and forward = CONTRADICTION $\wedge$ backward = ENTAILMENT. These are "interesting cases" that this section will focus on.

A remark on notation: $P(X|Y)$ will mean the probability of predicting X in the *backwards* direction given a *forward* prediction of Y. For brevity, E shall mean ENTAILMENT, N shall mean NEUTRAL, and C shall mean CONTRADICTION.

The model that is most susceptible to producing an interesting case is `RoBERTa`, although this is not a fair conclusion since `RoBERTa` is not trained on German, and the several multilingual datasets may artificially make the model appear weaker. The next most "interesting" model is `RoBERTa LXA`, which produced $P(C|E) = 0.0905$ and $P(E|C) = 0.0795$. This is a very interesting observation because RoBERTa LXA has been a very well-behaved model so

		P(E|F)	P(N|F)	P(C|F)
Dataset	F			
Taskbase SimpleK	E	0.2539	0.6211	**0.1250**
	N	0.0526	0.7722	0.1751
	C	**0.0284**	0.3337	0.6379
Taskbase Buyer-Seller	E	0.7092	0.2840	**0.0068**
	N	0.3509	0.5614	0.0877
	C	**0.3333**	0.3083	0.3583
Taskbase Evil Regular EN	E	0.2557	0.6097	**0.1346**
	N	0.1041	0.7369	0.1590
	C	**0.0705**	0.4094	0.5200
Taskbase Evil Regular DE	E	0.3155	0.5303	**0.1542**
	N	0.1430	0.6760	0.1811
	C	**0.0888**	0.3450	0.5662
Taskbase Evil Hard	E	0.4482	0.5317	**0.0201**
	N	0.1076	0.7978	0.0946
	C	**0.0427**	0.2027	0.7546
MNLI validation combined	E	0.3681	0.6094	**0.0225**
	N	0.2018	0.7213	0.0769
	C	**0.0209**	0.2525	0.7266
SNLI	E	0.1035	0.8744	**0.0221**
	N	0.1375	0.7608	0.1018
	C	**0.0157**	0.2118	0.7726
XNLI	E	0.2609	0.7003	**0.0388**
	N	0.1497	0.7526	0.0977
	C	**0.0263**	0.3322	0.6415
XNLI shuffled	E	0.2910	0.6668	**0.0422**
	N	0.1482	0.7362	0.1156
	C	**0.0315**	0.3628	0.6056

Table 13: Summary of probabilities of backward entailment predictions, grouped by dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F. Bold values are "interesting" cases.

far. The datasets which give it the most trouble are the two `Taskbase Evil Regular` datasets. These two datasets are challenging for all models, producing interesting cases over 10 percent of the time, but `RoBERTa LXA` is the only one that is much more likely to predict ENTAILMENT given CONTRADICTION.

A striking observation is that all models predict $E|C$ with quite high likelihood on the `Taskbase Buyer-Seller` dataset, between 13 and 64 percent. The aggregate $P(E|C)$ over all models is $33.\bar{3}$ percent for this dataset. There are few contradicting examples in this dataset in the forward direction, so even a single pathogenic example dramatically raises the fraction of interesting examples for this group. However, these cases can be explained by non-symmetric word associations, which shall be described later.

The vast majority of interesting cases are from the `Taskbase Evil Regular` datasets, with a few examples from `MNLI`, `XNLI`, and `SNLI` sprinkled in between.

Figure 5 summarizes the fraction of interesting cases in the complete corpus, grouped by model and source dataset. The most likely models to generate interesting cases are `RoBERTa` and `RoBERTa LXA`. The most challenging datasets are `Taskbase Evil Regular DE`/`EN` and `Taskbase Buyer-Seller`.

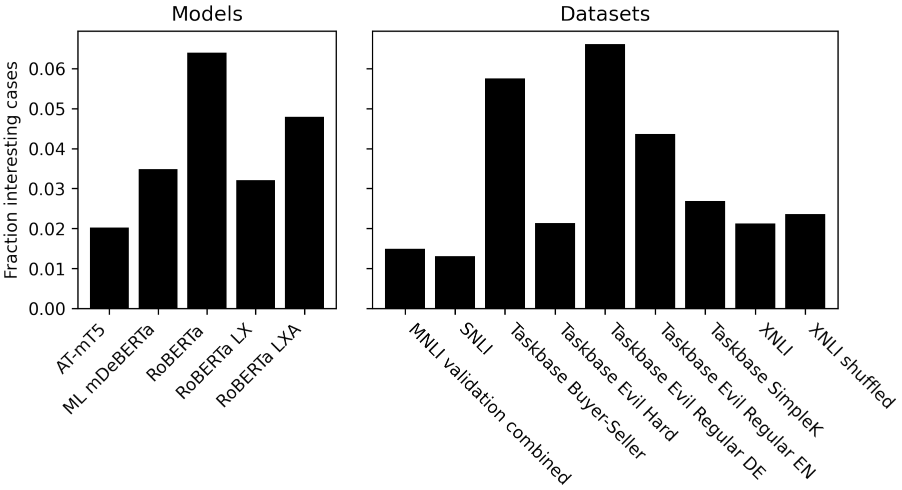


Figure 5: **Left:** Fraction of interesting cases produced by each of the used NLI models. **Right:** Fraction of interesting cases produced by all models for a given dataset.

It is unclear whether the dirty nature of `Taskbase Evil Regular` contributes to the prevalence of interesting cases.

After examining more of these interesting cases, which number 33785 and make up 3.98% over all datasets, several failure cases were identified which suggest that bidirectional entailment is not always suitable as a predictor of equality.

Word choice NLI models often fixate on certain words [9, 48, 54] that disproportionately affect the prediction. In the `Taskbase Buyer-Seller` dataset, this is evident in these examples:

- *Seller* bidirectionally entails *sale*.

- *Buyer* bidirectionally entails *purchase.*
- *Sale* bidirectionally contradicts *purchase.*

This has unwanted and often surprising effects when these words are combined. Looking at the `Taskbase Buyer-Seller` dataset, there are predictions which are ENTAILMENT in one direction and CONTRADICTION in the other direction; however, of these examples, CONTRADICTION in the forward direction is much more common. The following examples falsely produce CONTRADICTION forwards and correctly ENTAILMENT backwards:

p: The seller is obliged to hand over the object of sale to the buyer.
h: The seller must hand over the object of purchase.
(AT-mT5) (T 5.1)

p: The buyer has the obligation to pay the purchase price.
h: The buyer must pay the seller.
(RoBERTa) (T 5.2)

p: The buyer must pay for the goods purchased.
h: The buyer must pay the seller.
(ML mDeBERTa) (T 5.3)

p: The buyer has the obligation to pay the agreed price.
h: The buyer must pay the seller.
(RoBERTa LXA) (T 5.4)

These examples have one thing in common: the premise and hypothesis each contain seemingly-contradicting words such as “buyer”/“seller” or “purchase/sale” that actually entail in context. Of course, whether these words entail or contradict each other depends greatly on the model (Table 14).

Surprisingly, `RoBERTa LX` is the only model that does not show this behaviour on the Buyer-Seller dataset, although it does make a mistake on one example, correctly predicting CONTRADICTION forwards but falsely predicting ENTAILMENT backwards on this pair:

p: The seller can pay, the buyer can provide the goods.
h: The seller must hand over the object of purchase.
(RoBERTa LX) (T 5.5)

If the texts contain contradicting words, why is the entailment relation between them non-symmetric? This is difficult to explain without diving deep into the guts of the models and directly observing their attention, which is beyond the scope of this work. However, some hypotheses can be drawn:

Incomplete/incorrect associations NLI models may have learned incomplete, incorrect, or asymmetric word associations. The "purchase/sale" association is one example of an incomplete association — the association is correct when the word stands on its own, but incorrect in certain contexts. Even the simple pair

p: I purchased a car
h: Someone sold me a car. (T 5.6)

bidirectionally contradicts. If an NLI model never learned the knowledge that for every purchase there must be a sale (or for every buyer there must be a seller, etc.), other features in the pair will muddle the prediction. `AT-mT5` never learns this context, but `RoBERTa LXA` partially learns it, in that Text T 5.6 is NEUTRAL forwards but ENTAILMENT backwards.

Table 14 on page 56 lists some bidirectional examples on words like *purchase* and *sale*.

Incorrect associations are ones where the true relationship between words is not fully captured. For example, `AT-mT5` and `RoBERTa LXA` incorrectly capture the relationship between *never* and *rarely* — these texts entail forwards (incorrect) and contradict backwards (correct). So, many examples containing these two words, such as

p: You never call
h: You rarely call (T 5.7)

will be an interesting case.

So, NLI can fail on words which contradict on their own, suggesting that NLI models are biased by the meaning of individual words in a text as well as the context of the sentence, and that sentence contexts are not always learned. This example was illustrated in the Buyer-Seller example in the form of a transaction (i.e. "The receiving party *gets* something and the giving party *provides* something"), but this could apply to other examples containing contradictory words as well (e.g. "A person *leaves* one places and *enters* another).

Short texts lack features Very often an interesting case will be one with keywords or sentence fragments that are unrelated. Since there is not a lot of information or extractable features in short texts, an NLI model might not be able to reach a solid conclusion but will spit out some prediction anyway. This is acutely evident in the `Taskbase Evil Regular EN` and `DE` datasets, where the majority of responses and hypotheses are sentence fragments. Here are 3:

p: political frictions
h: Cold War (T 5.8)
(Forward: CONTRADICTION*, backward:* ENTAILMENT*)*

p: no freedom of choice
h: Capital letters (title / beginning of line) (T 5.9)
(Forward: ENTAILMENT*, backward:* CONTRADICTION*)*

p: hexe
h: Lehrhaft, belehrend (T 5.10)
(Forward: ENTAILMENT*, backward:* CONTRADICTION*)*

Superlatives, subsets, and senses Does “excellent” entail “good”? What is excellent must at least be good, but what is good isn’t necessarily excellent. In fact, one might make the argument that something which is only good can never be excellent, otherwise the thing would be called excellent in the first place.

To exaggerate even more: does “the best” entail “good“? If so, we get into uncomfortable territory — if a superlative entails a similar but lesser adjective, would “never” justifiably entail “rarely” as above? Would “You always call” entail “You sometimes call”?

If we ask `AT-mT5` or `RoBERTa LXA`, we find that the answer is “it depends”. When presented with this text pair:

p: His grades are brilliant.
h: His grades are good. (T 5.11)

both models predict ENTAILMENT in the forward direction but CONTRADICTION in the backward one. But with this one:

Model	Premise	Hypothesis	Forward	Backward
AT-mT5	Buyer	Seller	C	C
	Buyer	Purchase	E	E
	Buyer	Sale	C	C
	Seller	Purchase	C	C
	Seller	Sale	E	E
RoBERTa	Buyer	Seller	E	C
	Buyer	Purchase	E	E
	Buyer	Sale	E	E
	Seller	Purchase	E	E
	Seller	Sale	E	E
ML mDeBERTa	Buyer	Seller	C	C
	Buyer	Purchase	E	E
	Buyer	Sale	C	C
	Seller	Purchase	C	C
	Seller	Sale	E	E
RoBERTa LXA	Buyer	Seller	C	C
	Buyer	Purchase	E	E
	Buyer	Sale	E	C
	Seller	Purchase	E	E
	Seller	Sale	E	E
RoBERTa LX	Buyer	Seller	C	C
	Buyer	Purchase	E	E
	Buyer	Sale	E	E
	Seller	Purchase	E	E
	Seller	Sale	E	E

Table 14: Bidirectional entailment on several models for the words “Buyer”, “Seller”, “Purchase”, “Sale”. *Forward* is the prediction when entailing the premise against the hypothesis. *Backward* is the prediction when entailing the hypothesis against the premise. *E* represents entailment. *C* represents contradiction.

p: You always call.
h: You sometimes call. (T 5.12)

both directions are CONTRADICTION. Things get messier with this pair:

p: You always call me on Mondays.
h: You often call me on Mondays. (T 5.13)

This one is ENTAILMENT in the forward direction but NEUTRAL backwards, even though it could be argued that “always” contradicts “often” because “often” implies that there are mondays when you don’t call me.

Another example:

p: Kinder
h: Nachkommen (T 5.14)

p: Children
h: Descendants (T 5.15)

`AT-mT5` seems convinced that the forward direction is ENTAILMENT but the backward direction is CONTRADICTION (although it seems to have no issue with “grandchildren”). It is possible that the model is confusing the word sense of “children”, since the word may mean in its context either a descendant, or what we would call a young human who loves to play. Indeed, a quick search through MultiNLI reveals that it overwhelmingly refer to “children” as the latter definition, and seldom in the context of family or as descendants.

It is also possible that the model does not understand the concept of subsets. A child is certainly someone’s descendant, but a descendant is not necessarily a youngster. The strong association of “children” to “youngster” may overwhelm the weaker association between “descendant” and “child”, hence the asymmetric ENTAILMENT-CONTRADICTION relation.

Too Much Information Bidirectional entailment is useful to address what we shall call the **Too Much Information (TMI) problem**, where a student can receive a passing grade despite there being false or confounding knowledge in the response. This tends to happen when hypotheses are too vague, but can occur even with well-constructed ones. Extra knowledge in the premise may (a) confuse the NLI model and cause it to output a wrong prediction, or (b) be incorrect entirely. Let us see how we can confuse an NLI model. Consider the following task:

Task: *How would you describe Ronaldo’s back yard?*
h: *Grassy* (T 5.16)

This hypothesis will tend to follow from many responses that contain the word "grassy". For example, the premise

p: *It is quite grassy, so he can practice football* (T 5.17)

entails *h*, but

p: *It should be grassy although it's quite dry where he lives so probably not* (T 5.18)

also entails, even though it may be considered contradictory!

The risk of writing vague, or keyword-length hypotheses (as opposed to full sentences) to open-ended tasks is that many students will write sentence-format responses which contain confounding or false knowledge. This phenomenon can be observed in Taskbase's production data.

Even with well-constructed hypotheses in a full-sentence environment, a student who mistakenly adds incorrect information to a response may still receive a passing grade. Consider this task and hypothesis:

Task: *What event defined Napoleon's life in 1812?*
h: *Napoleon invades Russia.* (T 5.19)

A student may well write:

p: *Napoleon invades Russia, also known as the Soviet Union.* (T 5.20)

This premise entails in the forward direction, because it completely encompasses the knowledge in *h*. However, there is supplementary information that is incorrect.

Bidirectional entailment again comes to the rescue. By requiring that the texts entail in both directions, it rejects student responses containing Too Much Information. If a student does include Too Much Information, appropriate feedback could be assigned:

It looks like you wrote more than was expected. Try including only the information that you learned in class!

TMI and noise Too Much Information confounds the model further by increasing variability of predictions. Going back to the example with Ronaldo's lawn, the pair

p: It should be grassy although it's quite dry where he lives so probably not
h: Grassy (T 5.21)

entails in the forward direction, but

p: It should be grassy but it's quite dry where he lives so probably not

h: Grassy (T 5.22)

does not (the premises differ only in one word: *although/but*). Interestingly, these two premises entail each other in both directions, suggesting that they are equivalent.

This finding reveals two things:

First, two texts which are semantically equivalent (as judged by bidirectional entailment) are not interchangeable in other contexts. It is likely that NLI models fixate on specific words which disproportionately contribute to the prediction. It is possible that the word "but" is more highly associated with contradictions in the training dataset than with entailments. Indeed, similar behaviour has been seen on open datasets [9, 48, 54].

Second, premises containing A Lot Of Information can introduce noise into the predictor, increasing the likelihood that a long response will accidentally produce false predictions. Responses should therefore be limited in length. Long premises or premises with differing sentence structure from the hypothesis also contribute to noise [85]. This phenomenon is seen in the MultiNLI, XNLI, and ANLI datasets, which tend to have long premises and short hypotheses.

Responses that fool the model into making an incorrect prediction are adversarial, and can be used to fine-tune models. Indeed, the ANLI dataset is made up of similar premises: tricky and long texts expressly designed to confuse NLI models.

Note that, normally, if a premise entails a mistake hypothesis, the response should surely be flagged as incorrect, and appropriate feedback given to the student. In addition to falsely predicting ENTAILMENT when premises contain confounding information, another concern with TMI is that a response may falsely entail a mistake hypothesis, which would return feedback to the student that is not relevant.

5.5.6 Wrapping up

Bidirectional entailment can be a tool to determine semantic equality between two texts, but it is often fragile. Using bidirectional entailment doubles the chance of a misclassification.

It was discovered that entailment is not a transitive relation. $A \vDash B \land B \vDash C$ does not necessarily indicate that $A \vDash C$. In fact, even $A \Leftrightarrow B$ does not imply that both A and B entail with C.

Depending on the training data, a model may learn a subset of senses for a given word, causing a misclassification if the NLI pair refers to a weakly-learned sense. For instance “child” might strongly associate with “youngster” but weakly with “descendant”.

Bidirectional entailment can identify premises with Too Much Information, which is useful for filtering out student responses that contain more information that is present in the hypothesis. This approach, however, is fragile, and fails when the texts contains certain words or when the texts are complex.

Overall, it would help to construct an NLI dataset with labels in both directions so that the backward entailment direction can be better characterized.

5.6 The Homer Simpson Paradox

As alluded to in a few of the above experiments, NLI models may sometimes ignore certain words or base their predictions disproportionately based on certain features in either of $\langle p, h \rangle$. An interesting NLI example discovered at Taskbase deals with NLI models entirely ignoring parts of sentences:

p: Lisa works at a nuclear power plant and eats a sandwich with Homer Simpson. (T 5.23)

h: Homer Simpson works at a nuclear power plant.

This pair, unfortunately produces an ENTAILMENT prediction on `AT-mT5`.

Let us call this phenomenon, where a model falsely predicts ENTAILMENT due to ignoring parts of sentences or the entire sentence structure, the **Homer Simpson Paradox**.

We have seen in the bidirectional entailment experiment that the presence of certain words influence the outcome of the prediction, but are there fragments of texts that *don't*? The goal of this experiment is to determine whether NLI models prefer to focus on specific classes of words (i.e. nouns vs. verbs vs. adjectives, etc.) and ignore others.

5.6.1 Hypothesis

This is a qualitative and explorative experiment; there is no hypothesis.

5.6.2 Data collection

The dataset used was `Taskbase Homer`. This dataset was constructed throughout this experiment and modified with adversarial examples designed to test a model’s affinity to certain parts of speech.

NLI models used were `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, `RoBERTa LX`.

5.6.3 Method

Every example in the Taskbase Homer dataset was run on all of the NLI models. There was no quantitative analysis done. Patterns in the predictions were examined and further tested by hand. When an interesting pattern arose, more examples were constructed and appended to the Taskbase Homer dataset.

5.6.4 Discussion

Several variations on the premise in Text T 5.23 also produce ENTAILMENT:

p_1: *Homer Simpson parachutes at a nuclear power plant.*[a]
p_2: *Homer Simpson eats a sandwich at a nuclear power plant.*
p_3: *Homer Simpson is a parachuter at a nuclear power plant.* (T 5.24)
p_4: *Homer Simpson fdgfwnqehfisf at a nuclear power plant.*

[a] Perhaps he is aiming into the cooling towers?

A pattern emerges here: `AT-mT5` apparently completely disregards the verb attached to the subject, Homer Simpson. Despite writing gibberish or contradicting information into the premise, the model always returns ENTAILMENT. It presumably focuses primarily on the nouns: “Homer Simpson” and “nuclear power plant” are always present, which biases the prediction towards ENTAILMENT.

The remaining NLI models exhibit similar behaviour, but not as severely. Table 15 on the following page shows the predictions of each model. `AT-mT5` misclassified 5 times of 5, `RoBERTa` 3 times, `ML mDeBERTa` 3 times, `RoBERTa LXA` 2 times, and `RoBERTa LX` 3 times. `RoBERTa LXA` is the best performer except for the two examples containing “Homer Simpson is a ____”, which, interestingly, all models got wrong.

There are apparently parts of sentences that are not considered by NLI models, in this case, verbs and objects of “is”. AT-mT5 is particularly bad at detecting not-entailments with substituted words, as it misclassified every example. Overall, it appears that the major deciding factor is matching “Homer Simpson” and “nuclear power plant”. Predictably, these premises correctly produce CONTRADICTION using all five models:

p: *Lisa Simpson works at a nuclear power plant.*
p: *Homer Simpson works at a solar power plant.*
p: *Lisa Simpson works at a solar power plant.* (T 5.25)
p: *Homer Simpson works at a house plant.*
p: *Lisa Simpson works at a house plant.*

Premise	Prediction
Lisa works at a nuclear power plant and eats a sandwich with Homer Simpson.	E C C N E
Homer Simpson parachutes at a nuclear power plant.	E E E N N
Homer Simpson eats a sandwich at a nuclear power plant	E N N N N
Homer Simpson is a parachuter at a nuclear power plant.	E E E E E
Homer Simpson is a fdgfwnqehfisf at a nuclear power plant.	E E E E E

Table 15: Results of entailing various premises against the hypothesis *Homer Simpson works a a nuclear power plant.*. Each of the five predictions was generated by a different NLI model; in order: `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, `RoBERTa LX`. E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION.

These premises show that the NLI models do recognize verb subjects and prepositional objects (albeit, with strings attached, as is seen in the next example).

Giraffes The next trial was run with

$$h\text{: } \textit{Giraffes eat leaves that grow on trees.} \qquad \text{(T 5.26)}$$

Premises were constructed as follows: for each noun and verb in the hypothesis, the word was replaced with (a) a word that preserves semantic correctness, (b) a nonsensical word in the same part of speech, and (c) gibberish. The results of the inference on all models is shown in Table 16 on the next page. The total number of misclassifications using these giraffe-based examples were: 5 for `AT-mT5`, 7 for `RoBERTa`, 5 for `ML mDeBERTa`, 3 for `RoBERTa LXA`, and 8 for `RoBERTa LX`.

The models continue to do somewhat well in recognizing verb subjects (except for `RoBERTa` and `RoBERTa LX`, which often are the worst performers throughout this work's experiments), and verb objects. A reversal over the Homer Simpson examples is seen for verbs — when it comes to giraffes, verb substitutions are caught. However, when all elements except the verb subject are substituted with gibberish, misclassifications arise. The overall best performer, `RoBERTa LXA` is also confused by the verb in the adjective phrase, along with nearly every other model except `AT-mT5` and `ML mDeBERTa`, which correctly recognize "leaves that fall on trees" as a contradiction.

Again, `RoBERTa LXA` is the best performing model, being robust to gibberish and substitutions in all but the verb in the adjective phrase (i.e. "that ____ on trees.).

Premise	Predictions
Rhinos eat leaves that grow on trees.	C C C C E
Specimens eat leaves that grow on trees.	N E C N N
foobarbaz eat leaves that grow on trees.	C C C C E
Giraffes admire leaves that grow on trees.	C C N C N
Giraffes write leaves that grow on trees.	C C C C C
Giraffes iurehe leaves that grow on trees.	E E E C E
Giraffes eat twigs that grow on trees.	C C C C C
Giraffes eat empathies that grow on trees.	C C C C C
Giraffes eat qwfhkmko that grow on trees.	E E C C N
Giraffes eat leaves that fall on trees.	C E C E E
Giraffes eat leaves that think on trees.	E E E E E
Giraffes eat leaves that rwmxkjfhu on trees.	E E E E E
Giraffes eat leaves that grow on shrubs.	C C E C E
Giraffes eat leaves that grow on ideas.	C C C C N
Giraffes eat leaves that grow on nvjoiej.	E E E C E

Table 16: Results of entailing various premises against the hypothesis *Giraffes eat leaves that grow on trees.*. Each of the five predictions was generated by a different NLI model; in order: `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, `RoBERTa LX`. E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION.

Usain Bolt Usain Bolt is the fastest sprinter in the world at the 100 and 200 metre sprints [148]. Some language models, however, appear unaware of this marvelous achievement, as shown in Table 17.

Surprisingly, the only model that correctly predicted the first four examples as not-entailment was `RoBERTa`, often the weakest model of the five! All others are woefully bad at recognizing that sloths, tortoises, and snails are slow animals. The trend persists if the premises are re-formatted to "*Usain Bolt runs like a ⟨animal⟩*", with even `RoBERTa` starting to fail on not-entailment examples.

These examples could be considered figurative or metaphoric language, which have been studied in the context of NLI [1, 2, 130]. The conclusions of these papers are that there is a lack of metaphoric language in existing NLI datasets, and that NLI models are not good at identifying not-entailments in the presence of figurative language, which is what is reflected here.

Concluding that a misbehaving model or $\langle p, h \rangle$ exhibits the Homer Simpson Paradox must be done with restraint. Even though these examples relating to Usain Bolt have adjective phrases as in the Giraffe example, and it appears that they are ignored, asking each model whether a snail, sloth, or tortoise are slow animals yields NEUTRAL. The exception is RoBERTa, which is still very particular about *how* it is asked:

Premise	Hypothesis	Predictions
Usain Bolt runs at the speed of a sloth.	Usain Bolt runs quickly	E C E E E
Usain Bolt runs at the speed of a tortoise.	Usain Bolt runs quickly	E C E E E
Usain Bolt runs at the speed of a snail.	Usain Bolt runs quickly	E C E E E
Usain Bolt runs at the speed of a cheetah.	Usain Bolt runs slowly	C C C C C
Usain Bolt runs at the speed of a cheetah.	Usain Bolt runs quickly	E E E E E
Usain Bolt runs like a sloth.	Usain Bolt runs quickly	E C E E E
Usain Bolt runs like a tortoise.	Usain Bolt runs quickly	E E E E E
Usain Bolt runs like a snail.	Usain Bolt runs quickly	E E E E E
Usain Bolt runs like a cheetah.	Usain Bolt runs slowly	C C N C C
Usain Bolt runs like a cheetah.	Usain Bolt runs quickly	E E E E E

Table 17: Results of entailing various Usain Bolt-related premises and hypothesis. Each of the five predictions was generated by a different NLI model; in order: `AT-mT5`, `RoBERTa`, `ML mDeBERTa`, `RoBERTa LXA`, `RoBERTa LX`. E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION.

h: *A slow animal.*
p: *A Snail.*
p: *A Tortoise.*
p: *A Sloth.* (T 5.27)

All of these examples produce ENTAILMENT on `RoBERTa`. Yes, the capitalization and full stops are important, otherwise the model will sometimes be undecided and predict NEUTRAL (as seen previously with full stops and keywords). So, even if models encode the knowledge that snails, tortoises, and sloths are slow, they might not call upon that knowledge when necessary.

From Table 17, the fact that the first three examples are mostly misclassified but the last example is correctly classified in all models may suggest that there is a strong entailing, association between "quickly" and "speed", and a corresponding contradicting association between "slowly" and "speed". This could be an explanation of the Homer Simpson Paradox: some parts of the texts weigh more on the final prediction than others, suggesting that the latter parts are ignored.

It is possible that the Homer Simpson Paradox goes hand in hand with imperfect word associations in some cases, but this cannot be known for sure unless the attention values are examined directly during the invocation of a model on a particular $\langle p, h \rangle$ pair. This may be a future direction to quantitatively identify patterns in attention between tokens of the premise and hypothesis.

Why models ignore words There are several plausible reasons why an NLI model may choose to disregard certain words:

- NLI datasets generally contain very short hypotheses, and long premises. The model might learn that certain parts of the sentence are irrelevant or not covered by most hypotheses, and ignore them entirely. This may be the case with non-essential qualifiers or phrases like "that grow on trees" which hypotheses are unlikely to contain. Hypotheses, however, will surely vary in subjects, objects, and verbs, but more research is necessary to confirm this.

- Human annotators might disproportionately create not-entailing hypotheses by substituting a certain class of words (e.g. the verb subject), therefore the model learns to be more sensitive to substitutions on those words.

- Artefacts in either the premise or hypothesis may bias the prediction in one or the other direction [9, 48].

- Models disproportionately place greater attention on certain correlated word-pairs, such as "speed" and "quickly", driving the prediction in one or the other direction.

- Models are missing certain word associations, or those associations are not strong enough.

5.6.5 Wrapping up

This experiment explored the Homer Simpson Paradox very shallowly. It did not provide good evidence about which words the model ignores or why, but it suggests a direction for future exploration.

Manually creating a dataset to satisfy the burden of proving the Homer Simpson Paradox would be challenging. Instead, is possible to mutate existing datasets such as MultiNLI by parsing entailing premises and substituting the leaves or even branches of the parse tree with words/texts of the same type. The overwhelmingly likely consequence of such a substitution is that it would invalidate the entailment. By looking at which substitutions produced the most ENTAILMENT predictions, it would be possible to see which parts of sentences are most likely to be ignored. Hypernym/hyponym and antonym substitution has been somewhat studied by Carmona, Mitchell, and Riedel [23].

5.7 Entailing keywords

Much of the Taskbase NLI corpus is made up of keywords or sentence fragments. The `Taskbase SimpleK` dataset is composed of keywords in the premise and hypothesis. Instructors may want to write tasks with keyword-like hypotheses to match as many correct responses as possible, for example:

$$\begin{array}{c}\textbf{Task: } \textit{What have you learned about lightning?} \\ h\text{: } \textit{electricity} \\ p_1\text{: } \textit{It is an electric discharge.} \\ p_2\text{: } \textit{Lightning is basically electricity.}\end{array} \tag{T 5.28}$$

How does entailment behave on keyword hypotheses? To discover this, an automated process was devised that extracts noun chunks from premises and tries to infer entailment between the sentence and noun chunks.

5.7.1 Hypothesis

H_0: Sentences will entail noun chunks only by 50-50 chance.
H_1: Sentences will entail noun chunks taken from the sentence and not entail noun chunks from other sentences.

5.7.2 Data collection

The premises used were collected from Taskbase data sets and SNLI:

Giraffes eat mostly twigs, and sometimes shrubs, grass, and fruit.
The merchant must hand over the object of sale.
The buyer is obligated to pay the merchant for the object of sale.
Whenever I return home, I give a treat to my dog.
Homer Simpson works at a nuclear power plant and eats ham sandwiches with his daughter, Lisa.
The sailor from South Africa lives happily with his wife in the house Jack built.
All living organisms are composed of cells, and are called unicellular when they are composed of a single cell or multicellular, when they have more than one cell.
A man inspects the uniform of a figure in some East Asian country.
A black race car starts up in front of a crowd of people.
A smiling costumed woman is holding an umbrella.

5.7.3 Method

The `AT-mT5` model was chosen for this experiment since it gives all-around good results.

Noun chunks were extracted from each sentence. First, stop words were removed from the sentence to remove short articles and pronouns, then noun chunks were extracted using the spaCy library.

In the first trial, pairs were constructed taking a sentence as the premise, and a noun chunk from that sentence as a hypothesis. All combinations of sentences and in-sentence noun chunks were considered. In the second trial, the hypotheses were noun chunks from different sentences. All pairs from the first trial were expected to entail; all pairs from the second trial were expected to contradict.

p-values were calculated using the Z-test.

5.7.4 Results

First trial In the first trial, extracted noun chunks were used verbatim in hypotheses. Some examples from this corpus:

p: Whenever I return home, I give a treat to my dog.
h: dog (T 5.29)

p: A black race car starts up in front of a crowd of people.
h: black race car (T 5.30)

Sample size was 36.

Pairs in the first trial should all entail. Inferring entailment on `AT-mT5` yielded an accuracy and ENTAILMENT recall of $0.\overline{540}$. That is, predictions were only insignificantly better than chance ($p = 0.3133$). However, when each hypothesis was given a full-stop, accuracy and ENTAILMENT recall were $0.\overline{945}$, which is significantly above chance ($p \approx 4 \times 10^{-8}$).

The only two failures in this corpus were:

p: Giraffes eat mostly twigs, and sometimes shrubs, grass, and fruit.
h: grass.
NEUTRAL (T 5.31)

p: All living organisms are composed of cells, and are called unicellular when they are composed of a single cell or multicellular, when they have more than one cell.
h: single cell.
CONTRADICTION (T 5.32)

The second failure is excusable; The first is an error.

Second trial In the second trial, noun chunks from different sentences were used as hypotheses. All combinations of sentences and out-of-sentence noun chunks were considered. Hypotheses were tested with full stops and without.

Sample size was 303.

Without full stops, accuracy and NOT ENTAILMENT recall were 0.9461, which is significantly over chance ($p \approx 1.08 \times 10^{-54}$).

With full stops, accuracy and NOT ENTAILMENT recall were $0.\overline{9339}$, which is significant over chance ($p \approx 7.43 \times 10^{-52}$).

5.7.5 Discussion

`AT-mT5` clearly recognize noun phrases contained within premises. The flexible nature of NLI means that noun phrases may be worded slightly differently, e.g. using different adjectives, and still be entailed. However, this requires that the hypothesis containing the noun phrase have a full stop.

Punctuation The fact that full stops on the hypotheses improves ENTAILMENT recall is interesting. Perhaps the model considers the full stop as a feature that indicates "this is a sentence". When the premise is also a sentence these features must match in order to predict ENTAILMENT.

Accuracy is slightly lower when full stops are added, which makes sense with `AT-mT5`. As seen in the full stops experiment (Section 10 on page 45, this model tends towards ENTAILMENT when full stops are added to the hypothesis, therefore decreasing NOT ENTAILMENT recall.

Second trial A pattern is seen in the second trial. All of the noun chunks which were entailed by unrelated sentences were very generic, such as *object*, *man*, *people*, *figure*. There were also others that matched in a narrower context, for example, *costume* entailed *uniform*. Interestingly, the word *treat* as entailed by three unrelated sentences, raising the question of how the model (mis-)learned the meaning of that word.

There are small difference between the with-full-stops and without-full-stops corpus. The hypothesis *treat* is only inferred with full stops, but the hypothesis *sale* is entailed by the sailor sentence (a coincidence, or perhaps a pronunciation/spelling error in the training datasets?). However, the difference is not statistically significant ($p = 0.1764$).

Wrapping up The `AT-mT5` model appears to not entail keyword hypotheses very well unless the hypotheses have full stops. Trying to capitalize the hypotheses did not produce any meaningful change. Interestingly, Table 22 shows that

`AT-mT5` on the `Taskbase SimpleK` dataset has no significant change in ENTAILMENT recall and a decline in accuracy when the hypothesis terminates with a full stop, so perhaps this phenomenon only happens with full-sentence premises and keyword hypotheses.

In the future, this experiment should be repeated on a wider range of models and texts, to make sure that robustness to nouns is not specific to `AT-mT5`; as well as with a wider selection and character of data. A dataset could also be produced, either manually from production data, or synthetically using word extraction, containing full-sentence premises and keyword hypotheses. It would also be worthwhile to extract more than just nouns, to determine whether other parts of speech, or even sentence fragments, follow this trend.

5.8 *is-a* relations

What facts can language models learn? General language models can learn information and encode it within the model's parameters [111]. The model can retrieve this information later and use it to answer natural language queries without depending on additional inputs, for example in *closed-book question answering*, where a model must answer questions using only knowledge it has encountered during training.

Consider the following task that might be given to a student in primary school: *Give an example of an invertebrate.* It is clearly impractical to expect the instructor to enumerate all living invertebrates as hypotheses. Instead, the language model should be clever enough to recognize whether a particular input is an example of an invertebrate or not. For example, the query

p: Lobster.
h: This is an example of an invertebrate. (T 5.33)

should indicate ENTAILMENT. These kinds of relations are called *hypernymy/ hyponymy*, or *is-a* relations. *Invertebrate* is a hypernym of `lobster` because lobsters are a kind of invertebrate.

Other kinds of relations exist. For example, an *is-a-part-of* relation is called meronymy or holonymy (a *meronym* is part of a *holonym.* For example, the query

p: A hoof.
h: This is a part of a horse. (T 5.34)

should return ENTAILMENT.

Databases such as WordNet [88] exist that manually enumerate these relations between words.

The goal of this experiment is to recognize whether NLI models encode hypernymy relations as entailment. Hypernymy is closely related to lexical entailment, which is the recognition of entailment between single words.

5.8.1 Hypothesis

On recognizing hypernyms:

H_0 NLI models do not recognize hypernyms and give chance predictions.

H_+ NLI models recognize hypernyms in the hypothesis and predict ENTAILMENT.

On recognizing not-hypernyms:

H_0 NLI models do not recognize not-hypernyms and give chance predictions.

H_+ NLI models recognize not-hypernyms in the hypothesis and predict NOT ENTAILMENT.

5.8.2 Data collection

A list of the 2500 most common nouns in English film subtitles, except for the first 50, was obtained from the SUBTLEXus dataset [138]. Hypernyms were obtained from WordNet.

5.8.3 Method

Only the `AT-mT5` model was used.

For each of the 2500 nouns, hypernyms were obtained from the WordNet database. Only hypernyms that also occurred in the list of 2500 nouns and were not part of WordNet's most common hypernyms (e.g. "unit", "object", "entity", etc.) were retained.

Let MostCommonNouns be the set of the 2500 most common nouns, and WordNetHypernyms(p) be the set of all hypernyms of p which also appear in MostCommonNouns.

Own hypernyms The *own hypernyms* experiment is meant to determine whether `AT-mT5` correctly identifies a hypernym of a word. Premise-hypothesis pairs were constructed like so:

```
pairs = []
for p in MostCommonNouns:
    for h in WordNetHypernyms(p):
        pairs.append((p, h))
```

That is, each of the nouns was paired with all of its hypernyms also appearing in the 2500 noun set. There were 7560 pairs in total. All entailment predictions for these pairs are expected to be ENTAILMENT.

Other hypernyms The *other hypernyms* experiment is meant to determine whether AT-mT5 correctly identifies non-hypernyms. Premise-hypothesis pairs were constructed like so:

```
pairs = []
AllHypernyms = [h for p in MostCommonNouns
                  for h in WordNetHypernyms(p)]
for p in MostCommonNouns:
    not_hypernyms = AllHypernyms - WordNetHypernyms(p)
    for i in range(3):
        pairs.append((p, sample(not_hypernyms, 1)))
```

That is, each of the nouns was paired with 3 words that are hypernyms of other nouns in the 2500 noun set There were 7500 pairs in total. All entailment predictions for these pairs are expected to be NOT ENTAILMENT.

Transformations Various transformations were tried on both premise and hypothesis:

- Punctuation: The premise or hypothesis was punctuated with a full stop at the end.
- Articlification: A non-capitalized indefinite article ("a" or "an") was prepended to the premise or hypothesis.
- Both: Both articlification and punctuation were applied.
- Template The hypothesis was embedded in a template: *This is an example of a/an* ___ (a/an is chosen appropriately).

5.8.4 Results

Table 18 on the next page shows the accuracy for both the *own hypernyms* (top) and *other hypernyms* (bottom) experiments. Maximum values are bold. Some combinations of transformations were not evaluated.

5.8.5 Discussion

The results show that leaving the premise as-is and punctuating+articlifying the hypothesis keyword causes AT-mT5 to detect hypernyms with the greatest

		Own hypernyms				
		Hypothesis transformation				
		None	Punc.	Art.	Both	Template
Premise transfm.	None	0.4767	0.4959	0.6001	**0.6466**	0.4205
	Punc.	0.5205	0.4499	-	-	0.4691
	Art.	0.5417	-	0.5880	-	0.5099
	Both	0.5608	-	-	0.5489	0.5708

		Other hypernyms				
		Hypothesis transformation				
		None	Punc.	Art.	Both	Template
Premise transfm.	None	0.9270	0.8880	0.8462	0.8637	**0.9564**
	Punc.	0.8895	0.9092	-	-	0.9391
	Art.	0.8907	-	0.8590	-	0.9408
	Both	0.8770	-	-	0.8717	0.9108

Table 18: Accuracy of predictions for both *own hypernym* and *other hypernym* experiments. Transformations of the hypothesis text are shown across columns. Transformations of the premise text are shown across rows.

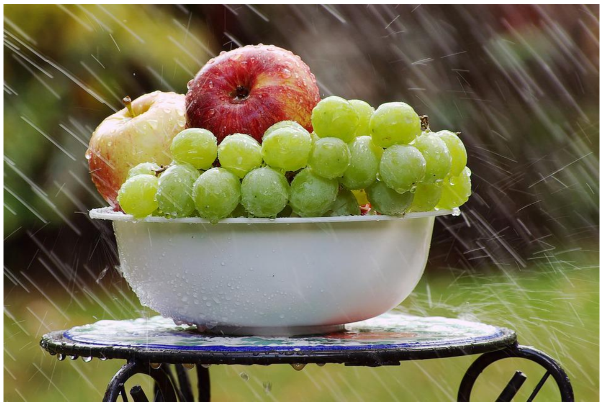
likelihood, at 64.66% ($p \approx 0$ over chance). Some transformations, such as templating the hypothesis and leaving the premise untouched, actually have the opposite of the intended effect. Overall, it seems that articlifying the premise and/or hypothesis has a positive effect on recalling entailment of hypernyms.

The *own hypernyms* experiment might be misleading because hypernyms returned by WordNet are low-quality with respect to colloquial use of English. This will be explored more below. In short, many hypernyms to some word w from WordNet have a very shaky semantic relationship to w. For example, WordNet indicates that two hypernyms to the word “heart” are “impression” and “belief”, which clearly hold no direct relation to “heart” but occur due to artefacts in WordNet’s very detailed and tree-structured database. It is suspected that NLI models actually know colloquial hypernym relations better than this experiment indicates — this can be tested by developing a database of words and their hypernyms as used in common English and repeating this experiment.

In the *other hypernyms* experiment, the transformations with the highest likelihood of yielding the correct entailment prediction were none on the premise and templating on the hypothesis. Interestingly, both experiments yielded the highest accuracies when the premise was untouched. Due to WordNet's too-detailed treatment of hypernyms, it is possible that templating the hypothesis in the *own hypernyms* experiment is actually the most realistic result given everyday English and that the *Both* transformation leads to false entailments.

Of course, this experiment only revealed how `AT-mT5` encodes hypernym relations. Other models may have different behaviour. It would be interesting to see whether they are more or less sensitive to some transformations and whether they perform better on this WordNet corpus than `AT-mT5`.

Sometimes, hypernymy is not desired. Take the following task[36]:



Task: What does this picture show?

A reasonable hypothesis would be *A fruit bowl in the rain* or *A bowl of fruit in the rain*. However, since an orange is a fruit, the premise *A bowl of oranges in the rain* would entail. Here, the instructor doesn't want to accept oranges since there are no oranges. One way to solve this issue would be to introduce a superhypothesis (see Section 6.2 on page 84) which has to entail the response, to place an upper bound on the hypernymy.

WordNet is not a good tool to create common-speech hypernymy datasets for three reasons:

- WordNet is *too* detailed. The word *giraffe* has as a hypernym *artiodactyl*, even though such a comparison would probably never occur in colloquial language, but possibly in professional language. It is possible to eliminate such esoteric words from a dataset by filtering only such hypernyms that occur in the top N most frequently used nouns in English. However, common but non-sensical hypernyms would still appear, as described below:

- WordNet hypernymy is strictly tree-structured, which can introduce unwanted transitive hypernyms that don't make sense. *Giraffe* has a a tran-

[36] Image from `https://pixabay.com/photos/grapes-apples-fruits-food-fresh-4125348`

sitive hypernym *living thing* (which is fair), but also *unit*, *object*, and *entity*. Is a giraffe a unit? Such a question probably doesn't make sense. Is a giraffe an object? One would say "no", since a giraffe is an animate being, not some inanimate "object" as colloquial usage would imply. Unfortunately, filtering these transitive dependencies is difficult because it is unclear how deep in the hypernymy tree to go. *Giraffe* has a fairly deep hypernymy tree because it includes all the biological classes, clades, and geni that giraffes are a part of. *Circus* (in the sense of a "circus troupe" has a relatively shallow hypernymy tree. Thus, heuristics like "take the first N levels" or "take the first $M\%$ of the hypernymy tree fail because of the size variation between hypernymy trees of different words and different senses.

- WordNet lists many different senses for nouns, which sometimes differ very slightly. Apart from each sense having different hypernyms, it is difficult to extract from a short text exactly which sense is meant and select the proper sense in WordNet. This leads to the inclusion of hypernyms that aren't related to the actual meaning of the word in context.

In the future, this experiment can be extended by considering characteristics or properties of things, instead of simple hypernymy. *is-a* relations are not limited to single nouns. For example, hypernymy would not be helpful with the following pairs but a perfect NLI model would recognize their entailment:

p: A rocking chair.
h: This is living room furniture. (T 5.35)

p: Frustration.
h: An upsetting emotion. (T 5.36)

p: A giraffe.
h: An animal living in the savannah. (T 5.37)

p: Rivella
h: This is a sugary drink. (T 5.38)

p: The sky
h: Blue (T 5.39)

p: Running
h: Something an athlete might do. (T 5.40)

p: Safari
h: This is an example of a place with animals. (T 5.41)

p: The Battle of Britain.
h: A historical battle. (T 5.42)

p: In Scotland.
h: In Europe. (T 5.43)

The sky *is-a* something blue (as colloquially understood), although since *blue* is not a noun, it would not appear in a hypernymy database (though it may appear in a general knowledge graph). The direct WordNet hypernym for *sky* is *atmosphere*, which is probably not what is meant by the average person when talking about the sky. A rocking chair is furniture, but it's also *living room furniture*. A bed would not be living room furniture, and this distinction, the qualified *furniture*, is unlikely to appear in hypernymy databases (it does not, for example, in WordNet). The final example is an *is-part-of* relation — Scotland is a part of Europe, so while Scotland might not necessarily imply the whole of Europe, something *in* Scotland does imply that the thing is also *in* Europe.

A future direction might be to investigate knowledge embedding using knowledge graphs instead of the WordNet database.

6 Towards building a framework for entailment-friendly tasks

This section discusses some general challenges with using NLI for digital learning as well as general language processing. Some solutions to these challenges are proposed, which might be directions of future research in how to adapt the NLI task to better fit digital learning, or how to adapt digital learning tasks to be more compatible with current NLI models.

6.1 Ambiguity in NLI

NLI is not a clear-cut task. Natural language is ambiguous[37], therefore NLI must struggle with ambiguities as well. These ambiguities cause issues when determining the gold label for NLI pairs, choosing one of several meanings a sentence may have, and establishing real-world context which affects the inference process.

One of the keys to writing NLI-friendly tasks is to remove possibilities of ambiguity right at the creation of the task. The task author must consider not just how students might interpret ambiguous words or phrases, but how the NLI model might interpret them.

This section will explore a few sources of ambiguity and how they may affect the performance of NLI as a feedback assignment system.

6.1.1 Ambiguity of categories

Consider this simple but cheerfully absurd pair:

p: Humans are exploring the solar system in the rockets that they built. (T 6.1)
h: Animals have discovered rocketry.

Is this an example of entailment or not entailment? This depends on whether a human is considered an animal, which in turn depends on who you ask. For a evolutionary biologist, humans are clearly animals — they are in the kingdom *Animalia* and share common genes and close ancestors with other animals. But ask a sociologist, a clergyman, or an ethologist (a scientist who studies animal behaviour), and their answer might be different: No, humans are not animals because their intelligence and behaviours greatly differs from those of animals. Even the average reader would probably classify this example as NOT

[37] The sentence "I ate a fish with a fork." has three different meanings.

ENTAILMENT, since raccoons zooming around on rockets is possible only in film [47].

Even disregarding the scientific definition of a human, the question of whether a text pair entails or does not entail depends on the context of the task. A task in biology should be interpreted with different presuppositions than one in arts or humanities — to say that a human is an animal in the context of art history is very different compared to the context of biology. Each subject has its own presuppositions.

6.1.2 Reasonableness of prior knowledge

Prior knowledge also affects ambiguity in NLI. Consider this example:

p: Genghis Khan conquered a lot of Asia.
h: The warlord conquered a lot of Asia. (T 6.2)

The gold label for this pair should be ENTAILMENT, although it would be unclear for an uneducated reader, and perhaps a machine, whether Genghis Khan is a warlord or not. Perhaps this text refers to Genghis Khan the stableboy, one of the Great Khan's descendants?

What information is the NLI model allowed to know? Kalouli *et al.* [64] argue that, when judging the gold label for a premise-hypothesis pair, the premise contains all the information against which the hypothesis should be judged, i.e. the premise contains the entire worldview. With this limitation, Text T 6.2 should be annotated as NEUTRAL. Nowhere does it say that *this* Genghis Khan is a warlord, even though it is very likely that *this* Genghis Khan is *the* Genghis Khan. The premise is also technically not contradictory, since *a* Genghis Khan, who was a warlord, did in fact conquer much of Asia. But how is the NLI model supposed to know that under Kalouli's restriction?

A certain amount of prior knowledge is healthy, otherwise the NLI model would suffer greatly in utility and wouldn't be able to understand many texts, like this one:

p: His house was full of flies.
h: A bunch of bugs got into his home. (T 6.3)

or this one:

p: The killer got a life sentence.
h: The killer will spend the rest of his days in jail. (T 6.4)

These are very simple pairs for a human to evaluate. But if we accept Kalouli's argument, where premises contain the entire world-view, the space of premises and hypotheses that entail would be *much* narrower. How does the model know that a fly is a bug? Where does it learn that a life sentence means a life in prison? Simple premises that contain little knowledge would only entail a very small set of hypotheses. Without *some* prior knowledge, NLI models would perform about as well as an average human reading a very scientific article — the grammar makes sense, but a lot of the words don't.

Then again, how would a model behave it it knows *all* available background information? This question can't be answered today. Though there exist language models trained on the whole of Wikipedia (let us assume that Wikipedia is a close as one can reasonably get to the "sum of human knowledge"), the difficult part is convincing the language model to understand which knowledge to retrieve, actually retrieve it, and present it to the user. Knowledge is useless if it is not accessible.

How do NLI models memorize facts in the first place? Do they memorize every prominent historical conquerer? This is not likely, as the figures *Foobar Khan* and *B.B. King* are also inferred to be warlords according to `RoBERTa LXA` and `AT-mT5`. The model is likely remembering word associations between *warlord/conquer* and *Khan/King*. Omitting *Khan* or *King* from the premise produces NOT ENTAILMENT (although `AT-mT5` will also accept the famous warlord, *Brad Pitt*).

6.1.3 Ambiguity of word meaning

The `SNLI` dataset includes this pair:

p: Two people embrace on the end of a dock.
h: Two people are facing opposite directions. (T 6.5)

Labels suggested by human annotators are all over the map: 2 ENTAILMENT, 2 CONTRADICTION, and 1 NEUTRAL. It is possible that some annotators assumed that an embrace is like a hug; others could have supposed it is like two people putting their hands around the other's shoulders, side by side[38] . This example has no gold label (although it *should* be NEUTRAL, since the hypothesis cannot be proven or disproven).

This example is one of the rare ones (alongside Text T 6.1) where $p \vDash h \wedge p \vDash \neg h$. The logical fallacy is resolved depending on the individual reader's interpretation of "embrace".

[38] There is a dataset, `e-SNLI` [20], which provides explanations from human annotators about why they suggested a certain label for most examples. Unfortunately, example this is not one of them.

6.1.4 Ambiguity of senses and homonyms

Section 5.5.5 on page 50 alluded to ambiguity of word senses. If a model was exposed only to one sense of a word during its training, it may have trouble discriminating between them in different contexts. This is further evidence that NLI is profoundly influenced by word-matching and not completely by semantics.

Take the examples listed in Table 19. From these, it is clear that `AT-mT5` has not completely learned the semantics of *bark* in one direction, which causes ambiguity and false predictions when used in different contexts. Notice how even the qualifier *tree bark* is insufficient to narrow down the sense.

Premise	**hypothesis**	**Prediction**
Giraffes eat bark.	Giraffes bark.	E
Giraffes eat bark.	The giraffe barks.	E
Giraffes eat tree bark.	Giraffes bark.	E
Giraffes eat tree bark.	The giraffe barks.	E
Giraffes bark	Giraffes eat bark	N

Table 19: Entailment predictions of `AT-mT5` about barking giraffes.

6.1.5 Ambiguity of subjects

What about this example:

p: At a childrens' football game, a young boy cheers for his team.
h: A child plays football. (T 6.6)

At first glance, this pair seems like it should be a CONTRADICTION. A child cheering is probably in the audience and not one of the players. However, this conclusion is only valid if the children are the same. At a childrens' football game, there is undoubtedly a child playing soccer. Thus, an argument can be made that this pair demonstrates ENTAILMENT if the children are different. Depending on one's interpretation, the gold label may be opposite. Kalouli's theory also does not help to disambiguate the situation, since all the information used to make either prediction follows directly from the premise.

We can also return to the *buyer-seller* example from bidirectional entailment (Section 5.5.5 on page 50):

p: Someone is buying a thing.
h: Someone is selling a thing. (T 6.7)

If this *someone* is the same person, then the pair should contradict (or maybe be neutral, since it is possible that someone is buying and selling at once). If the someone is different, then the pair must entail, since for every buyer there must be a seller.

6.1.6 False implication

This example is adapted from MacCartney [78]:

p: Some of the airlines saw costs growing more than expected.
h: Some of the airlines reported cost increases. (T 6.8)

Most reasonable people might consider this an entailment, yet, the hypothesis is not strictly implied by the premise. *Seeing* a cost increase does not imply *reporting* the cost increase, and so the label should be NEUTRAL. Yey, many NLI models do, like humans, predict ENTAILMENT on this pair. To quote MacCartney: "That the inference is nevertheless considered valid in the NLI setting is a reflection of the informality of the [NLI] definition." [78, p. 2]

6.1.7 Ambiguity of grammar

Ambiguous grammar, which is an inherent feature of natural language, makes it difficult to resolve ambiguities in premises. This premise:

p: *I ate a fish with a fork.* (T 6.9)

entails all of the following hypotheses on both `AT-mT5`, `ML mDeBERTa`, and `RoBERTa LXA`:

h_1: *A fork ate a fish with me.*
h_2: *The fish had a fork.* (T 6.10)
h_3: *I used a fork to eat a fish.*

Even though the grammar is ambiguous, a human would easily pick h_3 as the correct one.

Kalouli's approach fails here too — with no prior knowledge about fishes and forks, this ambiguity is completely unresolvable. Try replacing "ate", "fish" and "fork" with foreign words and ask a human which hypothesis is the entailing one.

What about:

p: *Tom reminded Jerry that he is a cat.* (T 6.11)

Who is the cat? `AT-mT5`, `ML mDeBERTa`, and `RoBERTa LXA` predict entailment on both hypotheses *Tom is a cat.* and *Jerry is a cat.*. Even a human, who had never heard of Tom & Jerry before, would have trouble resolving this example.

6.1.8 No way to enhance an NLI model's knowledge.

Clearly, the outcome of the NLI task is not always set in stone despite the rules of strict logical inference, and depends greatly on the context and the presuppositions of the reader.

Ambiguitity of categories, reasonableness of prior knowledge, and ambiguity of word meaning, can all be solved by enhancing an NLI model's prior knowledge of the world. If it were possible to tell the model "humans are indeed animals", "Genghis Khan was a warlord", or "an embrace is a hug where people face opposite directions", these three ambiguity problems could be addressed. There are 2 ways to do this short of fine-tuning, but none apply for current NLI models.

Context prompting Some NLP deep learning models accept *context* before the actual task is presented. For example, some question-answering models accept a context as well as a question, and attempt to answer the question from the context given. GPT-2 [105] and GPT-3 [17], being text generation engines, also accept context in the form of a paragraph or several before the text generation phase begins.

To date, NLI models do not support this approach. NLI models are fine-tuned to exclusively receive a premise and hypothesis, with no additional context. The context that the models use is limited to the knowledge they picked up during training.

In this work, the possibility of inserting context into the premise was briefly explored, but abandoned quickly. Any context given in the premise which affirms or denies knowledge present in the hypothesis is actually used in the entailment. Let's return to the human-animal debate:

p: A human.
h: This is an example of an animal. (T 6.12)

`AT-mT5`, `RoBERTa`, `RoBERTa LX`, `RoBERTa LXA`, and `ML mDeBERTa` all predict CONTRADICTION. So they take the colloquial opinion that humans aren't animals. But when context is given into the premise,

p: Humans are animals. A human.
h: This is an example of an animal. (T 6.13)

all 5 models then predict ENTAILMENT. So far so good. Unfortunately,

p: Humans are animals. A submarine.
h: This is an example of an animal. (T 6.14)

also becomes ENTAILMENT on all models, suggesting that the context text is actually entailing the hypothesis, not the response *A human*.

Many other texts were tested in this way as well, with the same conclusion being drawn.

Few-shot learning Few-shot learning is a technique wherein a model can learn some context to a task by providing it with a few examples of inputs and outputs before processing the actual prompt. The context is applicable only to one inference — once the inference is done, the model "forgets" all it learned from the context and has a clean slate, ready for the next inference.

Few-shot learning is useful because the few-shot examples can be specialized to the specific task at hand. In addition, typically, only a few examples are needed to see an improvement in performance.

Working off of the GPT models, Schtick and Schültze [120]; and Gao *et al.* [43] explored few-shot capabilities of text generation models for question-answering. Want *et al.* [141] later proposed a few-shot method for classification and regression based on re-formulating texts into entailment tasks.

In digital learning, few-shot learning sounds ideal. When a student response comes in, the model can sample from its collection of manually labeled responses as a reference. The model can draw not only on its own corpus of knowledge learned during training, but also the few-shot examples, which are specific to the task and are a good indicator of which responses entail or not entail which hypotheses.

Sadly, today's NLI models do not allow for few-shot examples to be provided. As with context prompting, it is impossible to provide more information outside the text and hypothesis. A possible workaround is to fine-tune an NLI model on some few-shot examples using a high learning rate and reset the weights at the end of the inference, but this approach may be too computationally expensive to be practical for every single inference.

Short of further fine-tuning, the knowledge that current NLI models possess is fixed. Context may be given to text generation models, but to be effective for NLI, a new dataset would have to be created with examples that require context to correctly infer, at a minimum.

6.1.9 Ambiguity in digital learning

How does ambiguity in NLI affect its performance in digital learning?

First, a language model fine-tuned on a specific dataset may not be appropriate for different learning subjects at once. For example, a dataset comprised of colloquial language may not perform well in a scientific subject, due do differences in vocabulary, ambiguity of categories, prior knowledge, and learned word associations which are not easy to change without further training[39].

Second, prior knowledge is hard to quantify, making it difficult to create tasks and hypotheses that won't be affected by the model's prior knowledge. Certain words might have some undesirable associations that mask desired word associations in the task. The learned association *Human* ⊭ *Animal* is one of these masking associations, and is hard to change without re-training

Third, responses containing elements that are similar to a hypothesis but dissimilar in context may generate false entailments. MacCartney [78] gives an example:

p: The main race track in Qatar is located in Shahaniya, on the Dukhan Road. (T 6.15)
h: Qatar is located in Shahaniya.

`AT-MT5`, `RoBERTa`, `ML mDeBERTa`, and `RoBERTa LX` all predict ENTAILMENT. `RoBERTa` LXA is the only one that correctly predicts CONTRADICTION, presumably because it has seen such tricky examples in the ANLI dataset. This example is not *really* an ambiguity but a failure to recognize word meaning, since the word *Qatar* has different noun functions in the premise and hypothesis; and possibly a failure to resolve the TMI problem described in Section 5.5.5 on page 50.

Fourth, a clever student might deliberately write an ambiguous response to try and confuse the model and cause it to produce incorrect predictions. When asked to provide an example of an animal, a student could write *A human might be an animal*, which entails on `RoBERTa` and `RoBERTa LX`[40] There is also the possibility that the word *animal* in the premise entails *This is an example of an animal*, even though *animal* should be excluded from the space of acceptable answers.

A side note: A clever instructor might create a mistake hypothesis *animal* to curtail such cases. Unfortunately, completely valid responses such as *A giraffe is an example of an animal* would also match this mistake hypothesis and falsely

[39] Section 6.4 explains why further fine-tuning may be a bad idea.

[40] Interestingly, when the premise contains *might not*, all models judge it as CONTRADICTION, further supporting the theory that artefacts such as "not" influence the prediction without considering the surrounding context. [48, 54].

be reported as incorrect. Templating sentences (Section 6.3 on page 87 may provide a solution to this problem.

Finally, because it is impossible to provide external context to today's NLI models without further fine-tuning them, and natural languages contains inherent ambiguities, some ambiguities in NLI are unresolvable and the task should be re-formulated to mitigate the ambiguous language.

Some ambiguities may be resolved by applying classical NLP techniques in addition to NLI, for example, ensuring that subjects and verbs are similar by examining the parse trees, but these methods are outside the scope of this work.

6.2 Superhypothesis-hypothesis model

A major challenge with open-ended questions is that student responses have a very wide range of quality and structure. Consider the following question:

What do giraffes mainly eat?

There are a variety of possible responses, from full-sentence to single words:

Giraffes eat plants.
plants
plants, grass, shrubs, fruit
giraffes eat mainly shrubs
(T 6.16)

The "correct" answer is that giraffes eat plants. How should this be expressed in a hypothesis? If the hypothesis were a full sentence (i.e. *Giraffes eat plants.*), then point-form responses like *plants* would not entail, since the knowledge *Giraffes* is not present in the response. On the other hand, if the hypothesis were point form, then a clever or misguided student might fool the system with a response like *Crabs eat plants.*, which would falsely entail.

A similar problem arises when the student mistakenly (or deliberately) includes extra, non-entailing, or incorrect information in their response (see the TMI problem, Section 5.5.5 on page 50). For example:

task: Which major historical event occurred on June 15, 1215?
premise: Napoleon signed the Magna Carta.
hypothesis: The Magna Carta was signed.

Clearly this premise should not be accepted (Napoleon had nothing to do with it), yet NLI models will classify it as entailing.

A human corrector naturally sets a lower and upper bound on the information that should be present in the response. The lower bound is the information necessary to answer the task, and must be present in the response. The upper bound is any additional *correct* knowledge that the student might enter and that is relevant to the task at hand. Knowledge that is outside the scope of the task, or knowledge that is incorrect, is outside of these bounds and should not appear in a correct response.

Unidirectional entailment sets a lower bound for the knowledge that is present in the NLI premise. As long as the knowledge in the premise is a superset of the knowledge in the hypothesis, the premise will entail. This is one of the fundamental differences between NLI and response-verifying — NLI is only interested in validating the lower bound, while a hypothetical response-verifying system will check the upper bound as well. Let $K(X)$ be the knowledge contained within X.

$$(NLI\langle p, h\rangle = \textsc{entailment}) \Rightarrow K(p) \supseteq K(h)$$

However, a mechanism to set an upper bound is necessary as well, to prevent students from entering extra erroneous information in their responses. The easiest way to do this is by constructing a third text, which we will call a "superhypothesis", which must entail the student's response. The superhypothesis should contain all the acceptable information, i.e. the most detailed answer that could be given.

$$\text{Superhypothesis} \xrightarrow{entails} \text{Response} \xrightarrow{entails} \text{Hypothesis}$$

$$(NLI\langle s, p, h\rangle = \textsc{entailment}) \Rightarrow K(s) \supseteq K(p) \supseteq K(h)$$

Let us return to the previous examples. We can reconstruct the giraffe task like so:

Task: What do giraffes mainly eat?
Superhypothesis: Giraffes mainly eat plants, including grass, shrubs, and fruit. (T 6.17)
Hypothesis: plants

Now, the responses *Giraffes eat plants.*, *plants*, and *grass, shrubs, fruit* all theoretically entail the hypothesis and are all entailed by the superhypothesis. If a student includes some knowledge not entailed by the superhypothesis (e.g. *Giraffes eat grass, just like crabs*), the superhypothesis does not entail the response.

The superhypothesis-hypothesis model provides an easy way to bound the acceptable knowledge space for a response class. The hypothesis contains the bare minimum of information necessary for a response to pass, while the superhypothesis contains the full set of allowable knowledge - any additional knowledge will not entail. Because we are still working with NLI, all the usual caveats apply, including the need for careful phrasing of the (super)hypotheses and random errors that occur along the way. For example, the response *shrubs* does not entail *plants* on some models for some reason. Another example: the response *giraffes mainly eat fruit* is not entailed by the superhypothesis because of the particular word order of *mainly* and *fruit*.

A drawback of the superhypothesis approach is that it requires additional work by the task author, and that it gets messy when there are multiple correct answers to a task. Remember the climate change example from Text T 2.1:

Task: *How will climate change affect our planet?*
Sample response: *Climate change will disrupt weather patterns.* (T 6.18)
Sample response: *Climate change will make temperatures more extreme.*

What would be the hypotheses here? There would have to be two correct hypotheses, at minimum, to be correct. Each correct hypothesis would need to be associated with a superhypothesis to check that a correct answer is within the acceptable knowledge bound.

Model-specific idiosyncrasies are still a factor, and should be evaluated on a case-by-case basis. For example, the text `King John signed the Magna Carta at Runnymede.` entails `Napoleon signed the Magna Carta.` using `RoBERTa`. Clearly, this RoBERTa has trouble distinguishing between historical figures!

The superhypothesis-hypothesis model is relevant when constructing the correct-class hypothesis. For mistake classes, if the response entails the mistake in any way, regardless of any other knowledge present in the response, the response should be marked as incorrect and feedback given to the student. Unfortunately, this approach presents an additional cognitive load for the instructor, and requires them to understand the reasoning behind it, which would make Taskbase's product less user-friendly and less accessible. Further, when constructing hypotheses and superhypotheses, it is helpful to understand some of the nuances of how NLI models interpret text, which users of the NLI system cannot be expected to know. An organization might employ "didactical experts" who are familiar with NLI models and perform premise engineering on texts to maximize entailment accuracy, but this is additional overhead. Until an NLI model is invented which does not require expert intervention to maximize accuracy on arbitrary, human-written texts, the amount of NLI methods for automatic feedback assignment should be minimized. Exploring more deterministic approaches such as premise engineering is also suggested, rather than relying on NLI to magically understand every kind of task.

6.3 Templating sentences

NLI models produce the best accuracy when they are fed complete sentences. Point-form or keyword inputs are more frequently misclassified than complete sentences or long-form texts, and there is greater variability and greater noise. This phenomenon can be expected, since most NLI models are fine-tuned on the same few datasets: SNLI, MultiNLI, and in the case of multilingual models, XNLI. These datasets contain primarily full sentences or long-form texts, so during training, models see natural language as a human would speak or write it in a conversation. Additionally, pre-training datasets are composed of sentences or paragraphs. The models are able to pick up sentence-specific features, for example word order and presence of parts-of-speech, and use those as factors in classification.

Unfortunately, when faced with an open task and a free-form text field, students write responses that are all over the map in terms of sentence structure. Consider the following task and 10 randomly-sampled student responses:

Task: *In his speech, Barack Obama explicitly addresses various groups of people. Explain which groups he is addressing and what intention could be associated with it.*

1. The younger group of people is considered, to them he owes thanks.
2. Spectators watching on TV and cannot be there.
3. it is addressed to fellow citizens and spectators.
4. Athletes.
5. to attract voters
6. USA
7. Specifically the residents of Washington D.C.
8. He addresses people close to him who listen to him.
9. The older people who have served their country in their lifetimes.
10. He makes a large group of people feel addressed.

(T 6.19)

One can see why these examples might be difficult for a machine to deal with. There is a mix of full sentences, keywords, sentence fragments, varying capitalization, and missing punctuation. The task also asks two distinct questions and there is a mix of responses to these. What should the instructor's hypothesis be to maximize accuracy? This is not clear. If at least all students were forced to write in full sentences, the hypothesis could also be a full sentence, and we would expect better performance in this case by eliminating some of the structural variability.

The purpose of templating responses is to coax responses into a specific sentence format where students fill in a cloze[41] . Unlike fill-in-the-blank-type closed-ended tasks, clozes in open-ended tasks would involve supplying a sentence fragment to suggest the response's sentence structure, then the student fills in the rest of the sentence in free-form text. The template should contain no information that enables the student to guess an answer, only what is already presented in the prompt. For example:

Task: *Explain which groups Barack Obama is addressing.*
Template: *He is addressing* ___. (T 6.20)

Task: *Why were aircraft carriers decisive in the Asia-Pacific War?*
Template: *They were decisive because* ___. (T 6.21)

Task: What is one of the functions of the Golgi apparatus?
Template: *One if its tasks is* ___. (T 6.22)

This approach has several benefits. First, it forces students to think and write in terms of full sentences, which can potentially improve communication outcomes. Second, it produces texts which are all full sentences of similar structure, which are easier for machines to process. Third, full-sentence structures enable extracting more knowledge from the NLI model, as seen in Section 5.8.

Many examples in Taskbase's datasets are dirty and not suitable for NLI for several reasons:

- Many task prompts were missing or incomplete. This was a flaw of the data collection process at the time.
- Hypotheses are very often point-form or sentence fragments. It is difficult to infer what the hypothesis means.
- Many hypotheses contain multiple facts in an either/or situation, for example:

  *h***:** *youth/students* (T 6.23)

  Unfortunately this is not how NLI works — an NLI model will try to find both *youth* and *students* in the premise. Premises that do not contain both become NEUTRAL or CONTRADICTION.

[41] The cloze method is a method where words or fragments are removed from a sentence and the reader is asked to fill in the blanks.

For these reasons, constructing a dataset of sufficient size using templated sentences was judged to be too time-consuming. However, many times throughout the work when analyzing stubborn texts, a templated-sentence approach provided a quick fix.

Templated sentence can also work to address the variable structure problem described in the section on the superhypothesis-hypothesis model (Section 6.2 on page 84). Recall the responses from Text T 6.16 on page 84. If these responses were templated (the template is underlined):

Giraffes eat plants.
Giraffes eat plants, grass, shrubs, fruit (T 6.24)
Giraffes eat mainly shrubs

then a single hypothesis format would be sufficient for all of these to entail.

Another example with a more open-ended task:

Task: *How will climate change affect our planet?*
Hypothesis: *Climate change will ___.*
Sample response: *Climate change will disrupt weather patterns.* (T 6.25)
Sample response: *Climate change will make temperatures more extreme.*

Not every task is suitable for templating. If a task is *too* open-ended, there exists no template that can be applied to every possible response:

Task: *What do you know about the Battle of Midway?*
Hypothesis: *???*
Sample response: *Aircraft carriers were used on both sides.* (T 6.26)
Sample response: *It was at the time the biggest naval battle in history.*
Sample response: *The Japanese navy was defeated.*

All three responses are true, but cannot be templated into the same sentence.

Templating sentences seems to provide many solutions to misclassifications arising from variability of responses with little additional cognitive load on the task author. It is not a universal fix, but nevertheless widely applicable. There is also the option of altering existing tasks or creating new tasks to be templating-friendly — just because the task about the Battle of Midway couldn't be efficiently templated, doesn't mean that the task cannot be put into a better, alternative wording.

6.4 Tailored datasets and fine-tuning

The standard process for developing NLP models using machine learning follows the *pre-train, fine-tune, predict* paradigm [74]. A generalized language model (GLM) is first trained on a massive text corpus so that it begins to "understand" the structure and form of natural language. This step is computationally very expensive, so most state-of-the-art GLMs have been trained largely by wealthy organizations with large-scale computing resources, a process which takes several days (e.g. Google's T5, [107], Facebook's BERT [35] and BART [73], Microsoft's DeBERTa [50], and OpenAI's GPT family [104, 105, 17] are or were all state-of-the-art models). Next, the GLM is fine-tuned on some downstream task. This step takes a GLM and "teaches" it to apply its knowledge to different objectives, e.g. text classification, text generation, translation. Most language models today which can be immediately applied to real-world tasks are fine-tuned versions of GLMs.

It is this two-phase training process that enables effective transfer learning using GLMs [58]. Fine-tuning is the most common approach of specializing a GLM to a specific downstream task or a specific context (e.g. the legal world, as in [63, 66]), and yields very good performance across the board [17].

Fine-tuning a generalized model offers many advantages over training a new model from scratch:

- Fewer examples in the fine-tuning dataset are needed. The size of fine-tuning datasets ranges from hundreds of examples [58] to hundreds of thousands [16, 17, 94, 147], compared to the millions and more required for pre-training.

- Fine-tuning can focus on a specific downstream task. For example, GLMs are fine-tuned on NLI datasets that specialize the models for the NLI task.

- Fine-tuning can teach the model new concepts, and reinforce certain areas of knowledge within a restricted context. For instance, a model understanding legal language can be fine-tuned from a GLM using focused datasets like COLIEE [63] and ContractNLI [66] in the legal world.

- Adversarial examples can be introduced into the fine-tuning dataset to teach the GLM misconceptions and eliminate failure cases [94].

With these strengths, fine-tuning would appear a perfect approach to solving many of the pain points of general NLI models in the setting of digital learning. Models can be fine-tuned on existing student responses and hypotheses, new fine-tuned models can be created for each learning subject, and relatively few examples are needed in the fine-tuning set to show a noticable increase in performance.

Unfortunately, fine-tuning a GLM or even a pre-fine-tuned NLI model is an undertaking not perfectly suited for digital learning platforms. There are important considerations in both the development and deployment of fine-tuned models.

First, tasks are evolving entities which must, annoyingly, remain compatible for a long period of time. That is, a task's NLI model must produce the same predictions every time it is invoked with the same inputs. Introducing an updated "version" of an NLI model might introduce regressions in existing tasks. This requirement of consistency also precludes the possibility of continuously fine-tuning the NLI model during a course as new tasks and responses come in. Fine-tuning on new data during a course may similarily cause regressions.

Second, it is not completely clear what set of data to fine-tune from. If a new version of an NLI model is fine-tuned based on data is has already seen, there is a risk of (a) the data becoming out-of-date by the time it is collected, as educational trends shift, and (b) the model overfitting on training data and not generalizing well to new tasks or responses. The risk of overfitting is greater when much of the data is similar, which is the case when the fine-tuning dataset is constructed from few tasks and all of the tasks' responses and hypotheses.

Unlike GLMs, models fine-tuned for digital learning see the world through an extremely narrow lens. For example, if a model M is fine-tuned on a dataset consisting strongly of science-related tasks and some instructor creates a task about art history for the very first time, there is a good chance that M would not perform very well since it has not seen relevant examples about this new field. A possible workaround would be to curate several models for different learning domains, e.g. a model for art history, one for chemistry, one for French, etc., and a "general" one as new subjects are introduced

Third, if the fine-tuning dataset comes from existing tasks, it is very likely that the learning domains for these tasks will be imbalanced. This imbalance would unfairly penalize tasks whose subjects are under-represented in the fine-tuning set.

Fourth, in the case of Taskbase, there are very few data which are clean enough to construct a fine-tuning dataset. See Section 6.5.

The fifth and final drawback is a technical one. Language models are heavyweight, requiring several gigabytes of memory if they are to be loaded and available. Initializing a language model takes tens of seconds to minutes, depending on the hardware and size of the model. Unfortunately, the evolution of language models appears to follow the mantra "bigger is better", with GLMs having more and more parameters than previous state-of-the-art. The number of parameters in language models is increasing very rapidly (see Figure 6). Increasing model size provides a greater boost in performance than other hyperparameters [17, 107, 116]. Serving many of these, potentially subject-specific, models at one time would require several GPUs in order to have enough memory. GPUs are an expensive commodity.

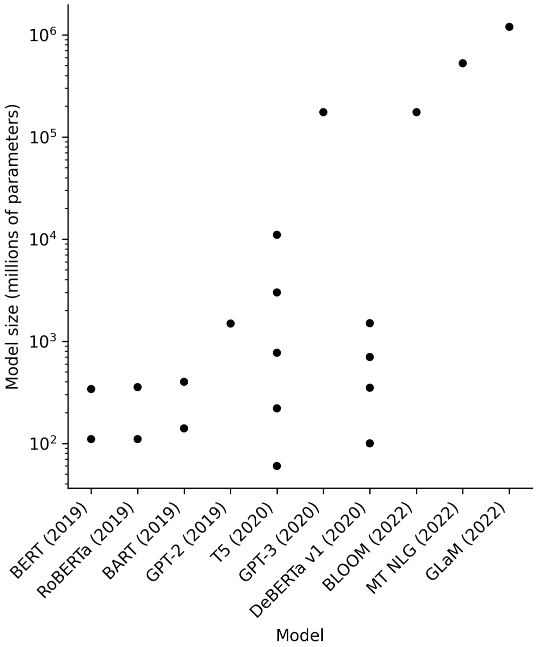


Figure 6: Trend of GLM model sizes over time. Models are often released in various sizes to fit different use cases. For each model, every point represents one of these sizes. The maximum model sizes are 175 billion parameters (GPT-3) and 176 billion parameters (BLOOM). The exception is DeBERTa v1, which is relatively small with respect to the trend but outperforms T5 in certain benchmarks [50].

Given these give drawbacks, developing new NLI models and evolving existing ones must be done with great care when working with online learning platforms.

There are methods other than fine-tuning that can increase NLI performance. These methods don't require massive amounts of fine-tuning data, are fast to apply, and can be toggled on and off if need be. Suppose that an instructor wishes to repeat a course from the previous year. Re-using the course's tasks for new students is useful because the instructor would already have developed task hypotheses and would understand the edge cases where NLI might fail. There also exists a corpus of student responses that can augment the predictive strength of the model, which can be done in several ways (Figure 7 on the following page):

1. Using clustering techniques to assign new responses to mistake (or correct) classes based on the assignment of previous responses to classes.

2. Using few-shot classification approaches. This involves using a few-shot model (one that does not need to be pre-trained, but receives context directly in the text) to infer entailment from previous examples [17]

3. Augmenting the NLI model with extra layers, pre-training their weights using open datasets, then fine-tuning only those layers on existing labeled examples from previous years. This approach allows for fine-tuning modular “heads” that can be swapped out depending on the task without touching the base model.

4. Selecting a different NLI model based on its performance on past, labeled responses.

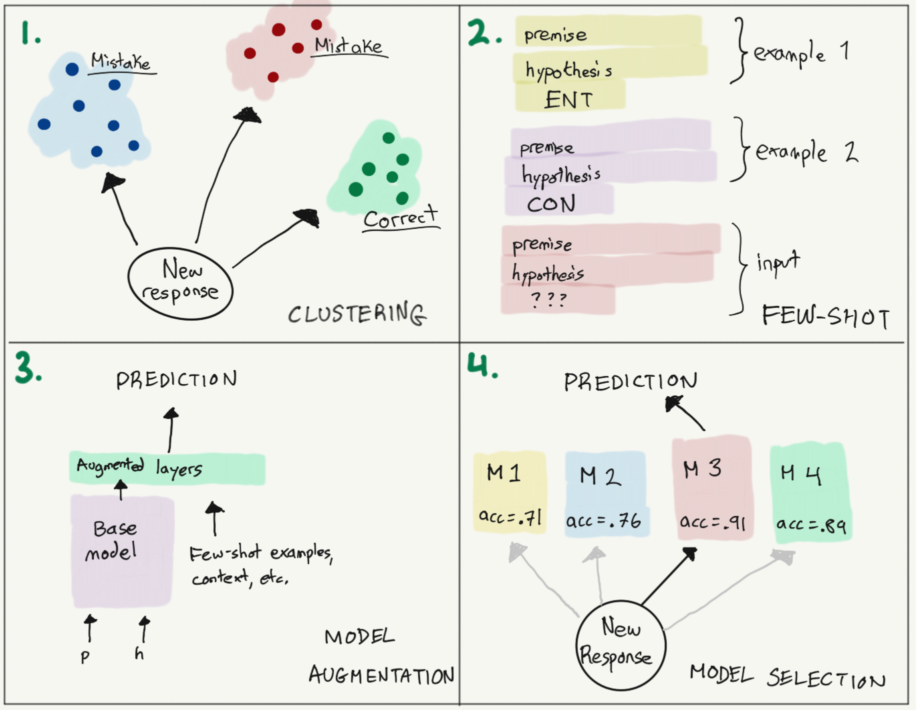


Figure 7: Illustrations of how to exploit existing responses when re-using tasks alongside NLI. **1.** A response is assigned into clusters of correct or mistake classes based on some distance function. **2.** The input is fed into a text generation model alongside several existing examples in a few-shot fashion. **3.** A smaller model can be fine-tuned on existing data as a re-usable head for the larger base model. **4.** Several models are evaluated on existing responses and the best-performing one is selected.

6.5 Tasks from the Taskbase corpus

Previous sections looked at analyzing tasks from Taskbase datasets *en masse*. This section will focus on tasks individually and attempt to deduce the various "types" of tasks and whether they are suitable for being used with NLI, whether they should be somehow modified, or if they are unsuitable at all.

After sampling many tasks from the Taskbase corpus and learning from this work's findings, it was determined that tasks which are good candidates for NLI...

- have responses which cover a small structural space;
- can be templated into full sentences or accept only full-sentence responses.

Conversely, bad candidates...

- have responses covering a large structural space;
- accept keywords or sentence fragments as responses;
- accept combinations of responses at once (i.e. questions “list N examples of X“;
- are unclear in how hypotheses and responses should be formulated.

A “bad candidate” has trouble accepting correct responses and frequently produces false predictions under normal use. “Good candidates” perform well under normal use, but these still succumb to failure under adversarial or challenging conditions, as will be demonstrated. Creating a task which is a “good candidate” does not preclude including additional safeguards such as bidirectional entailment, superhypotheses, or non-NLI techniques to minimize adversarial cases.

Each type of task in Table 20 on the next page was evaluated on several models to confirm whether they exhibit common failures with NLI, or whether NLI is robust for that type of task. To do this, adversarial premise-hypothesis pairs for each type were constructed. These pairs were designed to look like a legitimate pair, yet fool the NLI model into giving an incorrect prediction. Some inspiration was drawn from the ANLI dataset. In some examples, premises directly from student responses from the Taskbase Platform were used. For reasons of user privacy, these texts will not be printed in this section.

The following sections explore each type of task and present 10 important principles to which tasks should adhere in order to be NLI-friendly.

Type	NLI-friendly?	Example
Translation	Yes	Translate the following into French: *Paríž je hlavné mesto Francúzka.*
Negation	Yes	Negate the following sentence: *Jane studied in London.*
Captioning	Yes	Describe the interaction between the girl and the dog in this picture.
Clear answer	Yes	In a full sentence, why did Jane move to Frankfurt?
Verb tense	No	Write the following in past tense: Jane is running a marathon today.
List N examples	No	What does the main character like about the country? Name two things.
Multiple questions	No	Martin Luther King addresses different people in his "I have a dream" speech. Which groups is he addressing and what is his intention?
Keywords or categories	No	What is the word for these? Peach, orange, pineapple, grape.
No clear hypothesis	Partially	Write a sentence about dogs in past tense.
Large structural space	Partially	In The Little Red Riding Hood, what is the wolf's intention by dressing up as the girl's grandmother?

Table 20: A summary of the tasks in the Taskbase corpus and whether they are good or bad candidates for NLI.

6.5.1 Translation

Translation tasks ask the student to translate a text from one language to the other:

Task: *Translate this sentence into French:*
Emily was outside buying eggs at the market. (T 6.27)

Multilingual NLI models can handle premises and hypotheses in different languages. This trait gives task authors a lot of flexibility in creating hypotheses:

- Hypotheses can be written in the source language. Since multilingual models handle cross-language tasks well, the target-language response will probably entail the hypothesis.

- Hypotheses can be written in the target language. This boils down to single-language entailment.

Since multilingual models can predict entailment cross-language, there must be an additional language detection mechanism to ensure that the student response is written in the target language, otherwise a student might write the source-language prompt, which will most likely entail itself.

A similar approach to verifying translation tasks would be to employ a machine translation mechanism, either translating the prompt to the target language or the response back to the source language, and comparing the two same-language texts. This approach has pitfalls. First, a translation machine gives a specifically-formed sentence, which may not structurally match what the student wrote. Second, NLI is an easier problem than machine translation, and a translation machine is likely to give errors or misunderstand a word sense. In the end, verifying translation tasks using machine translation would be equivalent to NLI anyway, but with more steps that can go wrong. In fact, multilingual NLI is used to benchmark translation machines [33].

Since a translated text must be semantically equivalent to its source text, bidirectional entailment may be used to ensure equivalence.

Another benefit of multilingual models is that the instructor does need to know the target language and does not need to translate the texts themself. Since multilingual NLI will happily match e.g. an English sentence to a French one, specifying the source sentence is all that is required. In fact, this process can be completely automated for a language course: a sentence can be sampled from a large corpus (say, Wikipedia, or works of fiction), given to the student to translate, and a multilingual NLI model will compare the source sentence to the translated sentence and give appropriate correct/incorrect feedback.

Principle #1

Translation tasks are self-correcting, but require additional support beyond NLI.

6.5.2 Negation

Some tasks on language learning ask students to negate a sentence. Negation means a dramatic shift in the semantics of a sentence, and so transcends other grammatical features such as verb tenses and active/passive voice. One of the benefits of NLI models is their ability to detect negations even when the sentences are otherwise similar.

Care must be taken when a sentence has several possible negations:

Task: *Write this sentence using "not":*
Jean was relieved that he could come to Sofi's party. (T 6.28)

The hypotheses *Jean was relieved that he could **not** come to Sofi's party.* and *Jean was **not** relieved that he could come to Sofi's party.* falsely bidirectionally entail, and are interchangeable in this case since they both entail the prompt sentence.

Additional care must be exercised — if models display prediction biases due to hypothesis artefacts, placing a "not" in a hypothesis might encourage the model to predict CONTRADICTION [9, 48, 54]. Hossain *et al.* [53, 55] also demonstrated that negations are not well-represented in NLI corpora, which may also create artefacts.

Furthermore, this task has a very well-defined answer with little room for the student to deviate. It is possible that simple negation tasks could be performed cheaper and with better accuracy using simple pattern-matching and classical NLP pipelines. Although NLI models generally handle single negations well, negation tasks might be better suited for other methods.

Principle #2

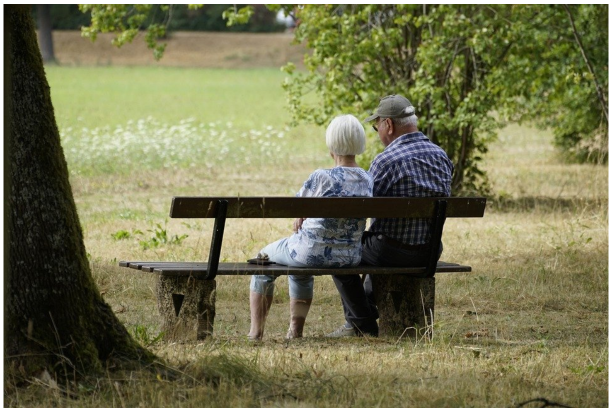
Reconsider using NLI when simpler NLP methods do the trick.

6.5.3 Captioning

The SNLI dataset is made up of image captions from Flickr. It would be a reasonable assumption that fine-tuning NLI models on SNLI would cause them to better understand captioning and captioning-like tasks, but, as always, caveats and failure cases occur.

Captioning is a didactically useful task for language learning, especially when combined with learning other facets such as grammatical structures or a second language. The following task demonstrates all of these[42]:

[42] Image from `https://pixabay.com/photos/retiree-pensioners-elderly-couple-7390179/`



Task: Describe what you see in this image in Spanish. Use past tense.

NLI is sometimes quite tolerant to structural variance of text and hypernyms. If a student provides a response that is more detailed than the hypothesis, the model may still predict ENTAILMENT. In this example:

p: A golden retriever is being embraced by a little girl.
h: A child is hugging a dog. (T 6.29)

the student writes a response with two hypernyms and passive voice, but it is still judged as ENTAILMENT by `AT-mT5`.

Unfortunately, even models trained on SNLI still exhibit problems with word-matching, TMI, and producing false entailments. The following nonsensical premises still entail on `AT-mT5` (which was not trained on SNLI), *and* `RoBERTa YNIE` (which was). The latter premise actually entails bidirectionally.

p_1: *A golden retriever is being embraced by a little girl about the morality of condensation.* (T 6.30)
p_2: *Hugging children on historical dogs*

Other tasks may resemble captioning:

Task: *What does Hermione do when Ron calls her a know-it-all?* (T 6.31)
h: She points her wand at Ron threateningly.

This example has commonalities with image captioning: the hypothesis is written in present tense, and could be a caption to an image (except the pronoun *She* and the name *Ron*, which are context-dependent).

A possible way to transform misbehaving tasks to an NLI-friendly format could be to phrase the prompt in a way to encourage captioning-like responses, but this strategy was not thoroughly tested.

Principle #3

Captioning-like tasks are good under normal use, but fine-tuning does not eliminate even simple adversarial examples such as those which arise from matched words.

6.5.4 Clear answer

Tasks can be open-ended but still have one or at most a few clear, well-structured responses. For example:

p: In a full sentence, why did Anne Frank spend all her time indoors? (T 6.32)
h: She was hiding from the Nazis.

This is a rather clear answer that presumably is taught verbatim in all studies about Anne Frank.

Another example:

p: How will climate change affect the world's oceans? (T 6.33)
h: Ocean levels will rise.

Other common and clear-cut hypotheses might be *Sea life will die off* or *The oceans will become warmer.* The point is, there is a finite set of acceptable hypotheses that are well-entailed by clear, well though-out responses.

Note that these two examples *may* be asked in the format of multiple choice since they expect a fact as an answer.

NLI works well here because the correct answers and common misconceptions are enumerable, have relatively little knowledge per sentence/hypothesis so as to not confuse the model, and have relatively few ways in which an answer can be phrased. In fact, this type of task is ideal for NLI because it very closely matches NLI models' training data.

Principle #4

Tasks should have few hypotheses, and few ways to phrase them

6.5.5 Verb tense (and possibly other grammar-sensitive tasks

In tasks which ask the student to provide a specific verb tense, `AT-mT5` is easily fooled with a different verb tense. In English, the sentence *Jane ran a marathon.* entails *Jane is running a marathon.*. In other languages with richer verb tenses, the problem is more critical. French for example has two future tenses: a *futur simple* and *futur composé* between which the `AT-MT5` model is unable to differentiate:

p: Jean va fermer la porte
h: Jean fermera la porte (T 6.34)

produces ENTAILMENT.

Worse, `AT-mT5` often cannot differentiate between less related verb tenses. The phrase *Jean est allé à France.* does entail *Jean ira à France* forwards and backwards, despite the former being past tense and the latter being future tense. This phenomenon also makes it difficult to construct feedback for misconceptions, as a misconception using an incorrect verb tense is likely to be entailed even by a correct response.

It is possible that NLI models apply some stemming logic to sentences. In this case, *est allé* and *ira* might both be stemmed to the infinitive *aller* (*to go*). To verify responses to verb-tense tasks, a grammar engine or string matching approach should be used.

It is not clear to which grammatical constructs this problem extends — constructing a comprehensive dataset of semantically-identical but grammatically-different sentences should be the first step to investigate this. For example, `AT-mT5` has trouble differentiating active versus passive voice:

p: The package was delivered by Jean.
h: Jean delivered the package. (T 6.35)

entail. Other grammatical constructs were not tested.

Principle #5

NLI is not a replacement for classical NLP.

6.5.6 List N examples

When a task asks to list N examples of something, it is impossible to construct a single hypothesis which is entailed by all correct responses when the total number of valid examples is greater than N. Suppose that a response p lists 3 valid examples, but the instructor allows 5 valid examples in the hypothesis h. Then p cannot entail h since there is some knowledge in h not present in p.

Further, as demonstrated by the `Taskbase SimpleK` dataset, if the N examples are keywords, there may be a lot of variability introduced.

Constructing misconception hypotheses is also challenging, since an NLI model may "miss" some items in a list, as shown here:

Task: name 3 things that giraffes eat.
p: Giraffes eat twigs, shrubs, and fruit. (T 6.36)
h: Giraffes eat shrubs

AT-mT5 predicts NEUTRAL on this example[43] . If the word *shrubs* were changed to *twigs* or *fruit*, then AT-mT5 would correctly predict ENTAILMENT. Since shrubs aren't entailed, the student would falsely be given feedback that they entered something that giraffes do not eat.

A workaround for this problem exists. When a task calls for N examples of something, the student can be asked to provide N distinct responses, which are matched against a set $H : |H| \geq N$ of hypothesis. If more than N of the hypotheses are entailed (and no mistake hypotheses are matched), the response is correct. This approach boils each of the N responses to entailment on a single fact, which is easy for NLI models to reason about. Templating responses to improve entailment accuracy can also be used here.

Principle #6

Tasks should not ask to provide more than one answer. If necessary, reduce these tasks to a multiple-answer ensemble.

6.5.7 Multiple questions

Sometimes, tasks ask multiple questions in the prompt:

Task: *Martin Luther King addresses different people in his "I have a dream" speech. Which groups is he addressing and what is his intention?* (T 6.37)

Multiple questions in a task are very NLI-unfriendly. First, task hypotheses would have to address both questions. If there are multiple possible hypotheses for each question, then the number of hypotheses that need to be written grows superlinearly, increasing time needed for task creation and reponse evaluation. Second, students are overwhelmingly likely (based on Taskbase's data) to only respond to one of the questions, leading to a result of "incorrect" since the hypotheses address more knowledge. Third, placing more knowledge on the shoulders of an NLI model is more likely to introduce noise.

In prompts that ask for more than one atomic thing, the task should be split up into sub-tasks. Subtasks are faster, more modular, easier to interpret, easier to correct, and do not lose any ability to evaluate students' learning. Moons *et al.* [92] describe some of the difficulties in maintaining non-atomic tasks and feedback items.

Principle #7

Tasks should ask an atomic question. Non-atomic questions should be split into several tasks.

[43] This is actually one of those examples where adding a full stop to the hypothesis makes everything work correctly.

6.5.8 Fragments, keywords and hypernyms

Task: What is the word for these? *Peach, orange, pineapple, grape.* (T 6.38)

The correct answer is *fruit*. However, as seen in Section 5.8 on page 69, any fruit can be used as a response and it would generally entail *fruit*. An adversarial premise might be *banana*, but a mistaken student may simple name one or all of *peach, orange, pineapple, grape* and the response would still entail *fruit*.

Section 5.8 on page 69 alluded to the idea of *categories* and *characteristics* and not necessarily *hypernyms*. A giraffe is an animal living in the Savannah, but this is not strictly a hypernymy relation. If an instructor wants to have students name animals living in the Savannah, because NLI models have little knowledge of relations between words and their *characteristics*, the instructor would have to enumerate all Savannah animals.

As seen throughout Section 5, the `Taskbase SimpleK` dataset of keywords and sentence fragments displayed significantly different behaviour and more variability than datasets composed of full-sentence examples. It is for this reason that keyword-based or sentence fragment-based tasks should be avoided: they are ambiguous, differ in behaviour, and require additional training for NLI models which understand complete sentences.

Tasks dealing with keyword hypotheses or responses are a good candidate for templating.

Principle #8

Prefer tasks whose responses are full sentences. Beware of variability and unwanted relations in keywords.

6.5.9 No clear hypothesis

Sometimes, a task can be so open-ended that there cannot be a reference answer:

Task: *Write a sentence in German using both a feminine and masculine noun.* (T 6.39)

Task: *What might Harry Potter think about a new course at Hogwarts called "Beginning Conjuration"?* (T 6.40)

Task: $\log_{10} 10 < N < \sqrt{25}$. How might N occur in real life? (T 6.41)

How is an NLI model supposed to evaluate responses to these without a reference? The space of hypothesis for these types of tasks is unbounded or so large that it is impractical to cover.

Tasks like this are common in language learning, where students must construct their own sentences or paragraphs with no strict guideline except the usage of a vague theme, as in Text T 6.39 on the preceding page.

NLI tasks characteristically require specific correct and mistake hypotheses. Tasks which allow an unbounded space of possible responses are, by definition, unsuitable for NLI.

Principle #9

Tasks must have a well-defined hypothesis.

6.5.10 Large structural space

Task: *In The Little Red Riding Hood, what is the wolf's intention by dressing up as the girl's grandmother?* (T 6.42)

This task is interesting. At first, it might appear to be a good candidate for NLI. However, when the task is posed to students, several "types" of responses appear:

The wolf is lying to her so she can eat her later.
The wolf is trying to assuage her.
The wolf invites her into the house.
The wolf wants to make her feel calm.
The wolf tries to calm her.
The wolf wants to show Little Red Riding Hood that she would be safe with him. (T 6.43)
The wolf lures her into the house reassuringly.
The wolf tries to win her trust.
The wolf appears nice.
The wolf is fooling Little Red Riding Hood.

Important structural differences are underlined.

These are all correct responses and should all entail the correct class. Many of these responses are semantically equivalent, and yet, do not entail the same hypothesis! For example, when a premise is phrased like *The wolf is calming her* and *The wolf wants to calm her*, `AT-mT5` will consider these as semantically different texts. There are several things that are happening with this particular task:

1. There are many different verbs.
2. There are many ways verbs can be qualified: as present tense (*lures her*, as a participle (*is lying)*, or as an infinitive (as in *wants to calm* or *tries to...* or *is trying to...*).
3. There is supplemental knowledge in about half of these prompts.
4. There are ambiguous pronouns (e.g. is the wolf a she or a he? If the wolf is female, the *she* pronoun becomes ambiguous).
5. There is passive and active phrasing, e.g. *calm her* and *make her feel calm*.

So many variations! The combinations in which they can be written are enormous, probably far greater than an instructor cares to list in individually-crafted hypotheses. And, students will no doubt invent even more ways to phrase a correct answer which may not be picked up by the NLI model. Even though some variations *technically* have different semantics, as is the case with *The wolf wants to calm her* and *the wolf is calming her*, this particular difference does not matter in the context of this question.

The problem with this task is that the structural space of premises is too large — there are too many ways in which a student can structure a response to mean the same thing. To be a good task for NLI, this space needs to be reduced. One way to do this is by templating, i.e. providing a prompt like *The wolf is trying to* ___ and having students fill in the blank. There are still many verbs to consider, but at least many of the variations are eliminated by the template.

There probably does not exist an algorithm for determining whether a task has a large structural space of premises. This of course highly depends on the task and the creativity of students. The major drawback of these kinds of tasks is that there is a large risk of differently-structured responses to not entail a common hypothesis. This risk is greater if there are slight semantic variances between responses that *should* entail a common hypothesis, but do not matter in context.

Principle #10

There should be few ways to phrase a response. Beware tasks which actually accept several subtle semantic variations of a response.

6.5.11 Wrapping up

Some tasks are NLI-friendly in that they comfortably entail responses to well-crafted hypotheses under normal use. Some tasks need to be adapted, others have too many failure cases. However, all tasks on several models demonstrate failures on adversarial texts, which may allow a clever student to “bypass” the task, or worse, entail a lazy or poorly-constructed response to an incorrect hypothesis. Additional methods of verifying responses are recommended, such as bidirectional entailment, superhypotheses, or classical NLP techniques.

To summarize the 10 principles:

1. Translation tasks are self-correcting, but require additional support beyond NLI.

2. Reconsider using NLI when simpler NLP methods do the trick.

3. NLI is well-behaved only under normal use. Fine-tuning does not eliminate even simple adversarial examples such as those which arise from matched words.

4. Tasks should have few hypotheses, and few ways to phrase them.

5. NLI is not a replacement for classical NLP.

6. Tasks should not ask to provide more than one answer. If necessary, reduce these tasks to a multiple-answer ensemble.

7. Tasks should ask an atomic question. Non-atomic questions should be split into several tasks.

8. Prefer tasks whose responses are full sentences. Beware of variability and unwanted relations in keywords.

9. Tasks must have a well-defined hypothesis.

10. There should be few ways to phrase a response. Beware tasks which actually accept several subtle semantic variations of a response.

7 Future Directions

Much work has gone into this thesis to arrive at an unsatisfying conclusion: "it depends." It depends on which model is used, which dataset is tested, which datasets the model was trained on, the specific wording of texts, whether a text uses the word "although" instead of "but", the speed of Usain Bolt, and the retrograde of Jupiter. Clearly, more questions have been raised than have been answered, but these questions are ripe for study.

Testing more models Is there an untested model that might have better performance? To answer this, a standard test suite of general and domain-specific NLI examples must be created. One interesting direction is investigating DeBERTa in more detail, as it offers fantastic performance for its size — DeBERTa v3 XSMALL outperforms RoBERTa in a quarter of the size, and the 1.5B parameter model outperforms Google's T5 with 11B parameters [50]. Small and performant models may be great candidates to fine-tune on common learning subjects and serve multiple subject-specific models at once.

Leveraging classical NLP Can classical NLP techniques be used to augment NLI? Could techniques such as parse tree inspection, word relation extraction [88], or Rhetorial Structure Theory (RST) [56, 81, 134] give a hint to resolve pairs which are ambiguous to a deep NLI model?

Fine-tuning datasets What is the effect of fine-tuning on a particular dataset? An ablation study where one model differs from another by only an absence of a fine-tuning dataset could reveal the contributions of individual datasets on NLI performance. An example has been alluded to about ANLI earlier in this work.

Exploiting existing responses When a task already has a sizeable corpus of manually- or automatically-labeled responses, how can these existing responses be used to increase prediction accuracy for new responses? Section 6.4 on page 90 proposes some possibilities.

Creating new datasets It would be tremendously useful to create new datasets to quantitatively address some of the findings from Section 5:

- A dataset of NLI examples where parts of one text's parse tree are replaced, to see whether NLI models are more sensitive to certain structures or parts of speech than others.

- A dataset consisting exclusively of premise-hypotheses pairs which resemble student responses and hypotheses to as task, perhaps constructed from a Question Answering (QA) dataset. The purpose of this dataset would be to assess whether it is a suitable surrogate for real-world data. SciTail [65] could be used as a starting point.
- An NLI dataset where the backward direction is also labeled, to assess the accuracy and value of bidirectional entailment.

Using adversarial examples Could fine-tuning on pathogenic cases improve prediction quality? Certain $\langle p, h \rangle$ pairs are pathogenic, meaning that they consistently fail and interpreting why they fail is hard. These pairs could be collected into a new dataset, which we shall call the Digital Education Adversarial NLI (DEANLI)[44], for use in future fine-tuning. Examples of adversarial pairs can be collected directly from a learning platform if task authors report misclassified responses through a user interface.

Exploring knowledge graphs Do knowledge graphs encode the same information as an NLI model? Some research has gone into using knowledge graphs alongside NLI [38, 121, 126, 127, 143]. Knowledge graphs can be thought of as similar to language parse trees, except that they encode semantic relations instead of syntactic ones. A knowledge graph could be used in the context of the TMI problem 5.5.5 on page 50 to assert that extra knowledge in the premise does not contradict a known fact.

Investigating dataset imbalance Most modern NLI datasets are constructed for 3-way entailment. However, as this work was concerned with 2-way entailment, datasets suddenly became imbalanced in favour of NOT ENTAILMENT outcomes. This may affect a model's apparent bias since models appear more performant if they bias predictions towards NOT ENTAILMENT and makes statistical analysis a little hazy (i.e. is a "chance" outcome 50/50% or 33/67% ?). Re-fine-tuning a base GLM on NLI datasets balanced for 2-way entailment would indicate if the imbalance has an effect on performance.

Investigating domain-specific datasets When models are fine-tuned on non-general datasets, does that knowledge transfer to evaluating entailment on Taskbase's datasets? Datasets such as COLIEE [63], ContractNLI [66], MedNLI [115], and SciTail [65] have not been studied in this work but may illuminate the path forward.

[44] it's funny because a "dean" is a person in charge of a school

8 Conclusion

Using NLI in digital learning is not a new concept. Methods of automatically verifying and grading student responses by comparing them to reference answers have existed since the 1990s. However, the use of deep learning to perform NLI is relatively new.

NLI appears to be a promising tool in digital learning, but sometimes requires human assistance to engineer tasks, hypotheses, and pre-process premises in order to perform well. No universal approach to engineering the NLI pipeline was found that worked on all texts. NLI can be a great tool to check simple student responses for semantic correctness of the presence of certain ideas, but should be augmented with other techniques to verify the technical correctness of the response. Many applications (e.g. knowledge extraction, document retrieval, document matching) of NLI assume that premise and hypothesis texts already exist and are fixed, or that at least premises are fixed, and that a mis-entailment has little practical consequence. But when a user can input arbitrary premises and hunt for adversarial cases, trouble ensues.

Performing NLI for feedback assignment is not easy. Many factors which can confound the NLI model and may inhibit prediction accuracy need to be taken into account, such as the Too Much Information problem (Section 5.5.5), several types of ambiguities (Section 6.1), variance between different NLI models, selection of training datasets, and domain-specific requirements that restrict the ability to fine-tune models.

NLI models often display random and inexplicable behaviour on some texts. Many times, these noisy predictions can be solved by prompt engineering on the premise and hypothesis, which may include enforcement of correct capitalization, enforcement of proper punctuation, entailment in the opposite direction, or bounding the acceptable knowledge space of student answers.

It is relatively easy to make NLI work in digital learning for common cases, but incredibly difficult to remove adversarial cases and ways for students to abuse the system and cause false entailment predictions to be made. Oftentimes, the inclusion of certain keywords is enough to trigger an ENTAILMENT prediction and fool a model into classifying a response as entailing. Indeed, NLI models appear to consider word similarity between the premise and hypothesis as well as semantic similarity, a property that has been revealed by engineering nonsensical premises using the same or similar words as a hypothesis. Models appear more sensitive to the presence of certain types of words while appearing to ignore other words entirely.

A challenge to deep-learning NLI is how to provide context knowledge to the NLI model, that is, how to augment or override the model's existing knowledge base in order to teach it subject-specific words or concepts. Current NLI models have no mechanism to insert context, nor are there any NLI datasets that provide tricky, context-dependent texts. It may be possible to exploit a general text-generation model such as GPT-3 to provide context, but such alternative directions were not studied in this work. Indeed, many failures in the tested NLI models were due to incorrect word associations, so a method to instruct an NLI model to interpret a word or concept a certain way would be tremendously helpful.

NLI models trained on the challenging ANLI dataset appeared to catch more pathogenic cases than other models. This suggests that adversarial data might be an important tool for continuously improving NLI models for digital learning. Indeed, results show that training and fine-tuning datasets are more important than model architecture. For example, three RoBERTa-based models, each fine-tuned on different datasets, all displayed different behaviour. However, fine-tuning on many datasets has diminishing returns — the `RoBERTa Ynie` model was trained on four datasets, including ANLI, but does not show such a dramatic increase in performance over others, and is actually outperformed by simpler models in some areas.

This thesis made the following contributions:

- A benchmark of common NLI model architectures and fine-tuning schemes on many of the most common NLI datasets with a greater range of collected statistics than many current surveys;
- A benchmark of the same models on datasets collected from the Taskbase Learning Platform;
- A characterization of the effect of full stops and capitalization on model performance;
- A characterization of NLI model behaviour on full sentences versus keywords or sentence fragments;
- Identification of pain points and failure cases of NLI models for a variety of phenomena, including word association;
- A presentation and characterization of bidirectional entailment as a tool for evaluating student responses to tasks;
- A short study of how well NLI models encode hypernymy relations;
- A summary of different types of ambiguity and how ambiguity might affect the NLI task in general as well as within digital learning;

- The **superhypothesis-hypothesis** model, in which the information of a student response may be bound;
- A text templating approach to NLI, in which student responses can be coaxed into a similar textual structure to eliminate variance and improve performance;
- An exploration of the drawbacks of model fine-tuning in digital learning and how these models may be augmented in other ways to improve performance over time; and
- An analysis of common task types from the Taskbase Learning Platform and the applicability of NLI to each task type.

A Tables and Figures

This section provides full data tables and figures from Section 5 which are too space-consuming to insert into the text body.

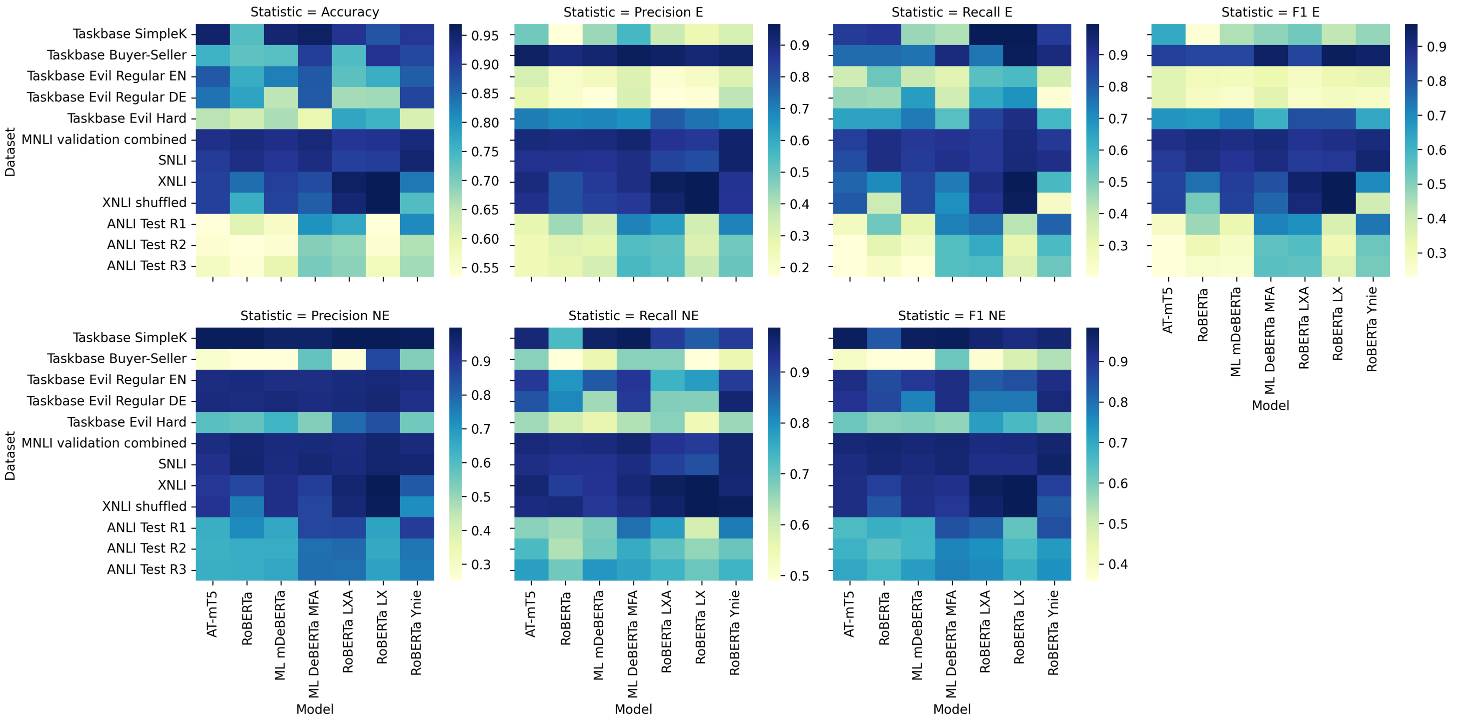


Figure 8: Heatmap of aggregated statistics for each model and dataset pair. Table 21 on page 116 displays these data in table form.

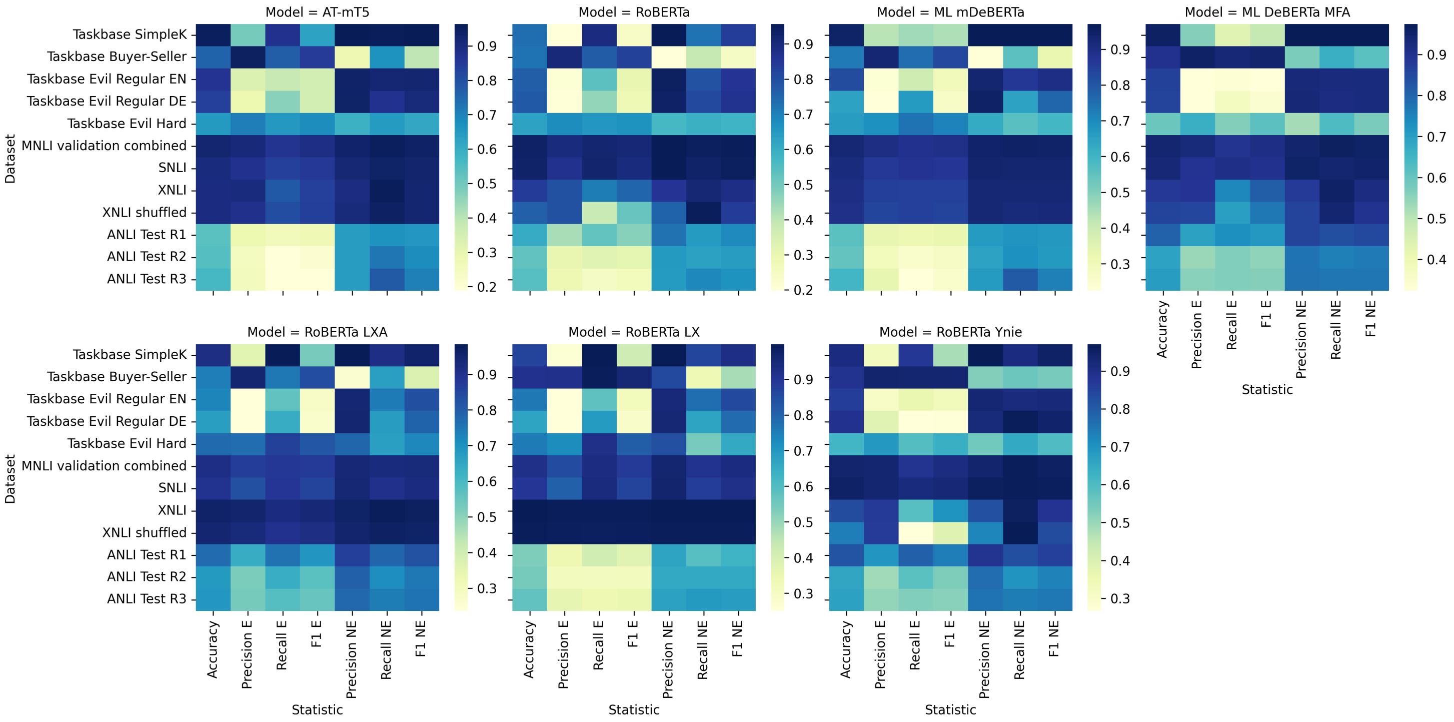


Figure 9: Heatmap of model performance for each dataset and statistic pair. Table 21 on page 116 displays these data in table form.

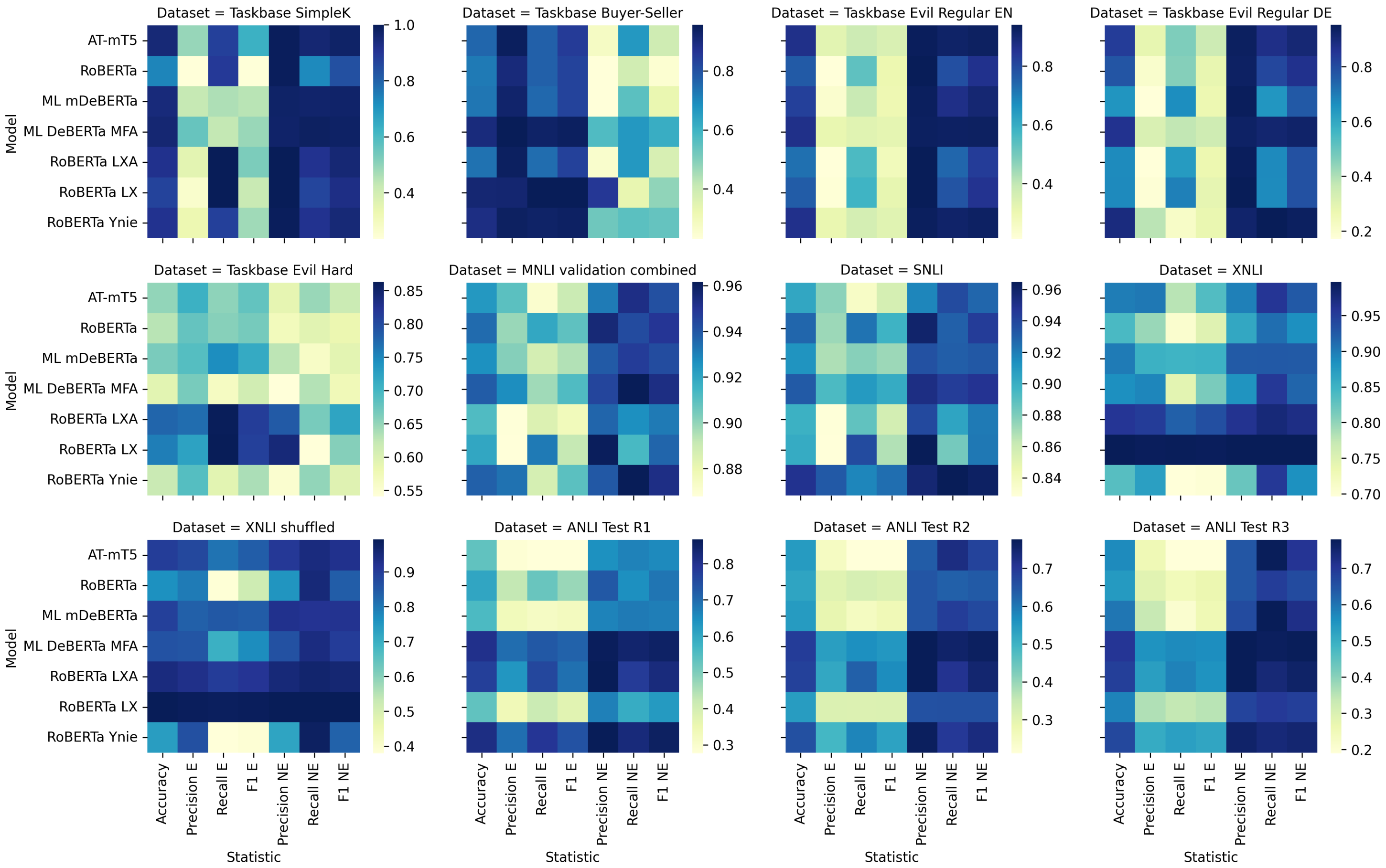


Figure 10: Heatmap of dataset performance for each model and statistic pair. Table 21 on the following page displays these data in table form.

Table 21: Summary of aggregated statistics for each model and dataset.

Model	Dataset	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
AT-mT5	Taskbase SimpleK	0.9518	0.4914	0.8769	0.6298	0.9937	0.9555	0.9742
	Taskbase Buyer-Seller	0.7582	0.9455	0.7704	0.8490	0.2791	0.6667	0.3934
	Taskbase Evil Regular EN	0.8668	0.3307	0.3815	0.3543	0.9334	0.9182	0.9257
	Taskbase Evil Regular DE	0.8371	0.2855	0.4658	0.3540	0.9393	0.8765	0.9068
	Taskbase Evil Hard	0.6513	0.7075	0.6541	0.6798	0.5897	0.6477	0.6173
	MNLI validation combined	0.9240	0.9096	0.8714	0.8901	0.9313	0.9527	0.9419
	SNLI	0.9036	0.8765	0.8367	0.8561	0.9168	0.9385	0.9275
	XNLI	0.8986	0.9031	0.7795	0.8367	0.8968	0.9582	0.9265
	XNLI shuffled	0.8988	0.8788	0.8076	0.8417	0.9076	0.9443	0.9256
	ANLI Test R1	0.5340	0.2871	0.2665	0.2764	0.6449	0.6682	0.6563
	ANLI Test R2	0.5450	0.2510	0.1826	0.2114	0.6394	0.7267	0.6803
	ANLI Test R3	0.5675	0.2552	0.1517	0.1903	0.6452	0.7769	0.7049
RoBERTa	Taskbase SimpleK	0.7345	0.1381	0.8923	0.2392	0.9928	0.7268	0.8392
	Taskbase Buyer-Seller	0.7255	0.9043	0.7704	0.8320	0.1842	0.3889	0.2500
	Taskbase Evil Regular EN	0.7665	0.2129	0.5327	0.3042	0.9411	0.7913	0.8597
	Taskbase Evil Regular DE	0.7776	0.2040	0.4554	0.2818	0.9336	0.8117	0.8684
	Taskbase Evil Hard	0.6299	0.6787	0.6570	0.6677	0.5709	0.5947	0.5826
	MNLI validation combined	0.9350	0.8989	0.9195	0.9091	0.9554	0.9435	0.9494
	SNLI	0.9277	0.8729	0.9237	0.8976	0.9589	0.9298	0.9442
	XNLI	0.8422	0.7961	0.7078	0.7493	0.8616	0.9094	0.8848
	XNLI shuffled	0.7619	0.7962	0.3841	0.5182	0.7554	0.9508	0.8419
	ANLI Test R1	0.6060	0.4268	0.5240	0.4704	0.7305	0.6471	0.6863
	ANLI Test R2	0.5290	0.3059	0.3234	0.3144	0.6507	0.6321	0.6413
	ANLI Test R3	0.5417	0.2874	0.2488	0.2667	0.6455	0.6892	0.6667
ML mDeBERTa	Taskbase SimpleK	0.9460	0.4286	0.4615	0.4444	0.9735	0.9698	0.9716
	Taskbase Buyer-Seller	0.7320	0.9273	0.7556	0.8327	0.2326	0.5556	0.3279
	Taskbase Evil Regular EN	0.8232	0.2404	0.3911	0.2977	0.9309	0.8690	0.8989
	Taskbase Evil Regular DE	0.6494	0.1666	0.6645	0.2665	0.9480	0.6478	0.7697

Continued on next page

Table 21: Summary of benchmarked aggregated statistics for each model and dataset.

Model	Dataset	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase Evil Hard	0.6645	0.6882	0.7442	0.7151	0.6271	0.5606	0.5920
	MNLI validation combined	0.9261	0.9020	0.8874	0.8946	0.9390	0.9473	0.9431
	SNLI	0.9125	0.8684	0.8777	0.8730	0.9358	0.9306	0.9332
	XNLI	0.9005	0.8524	0.8487	0.8505	0.9245	0.9265	0.9255
	XNLI shuffled	0.8947	0.8368	0.8497	0.8432	0.9243	0.9171	0.9207
	ANLI Test R1	0.5590	0.3323	0.3174	0.3247	0.6652	0.6802	0.6726
	ANLI Test R2	0.5450	0.2921	0.2545	0.2720	0.6488	0.6907	0.6691
	ANLI Test R3	0.5958	0.3347	0.2090	0.2573	0.6649	0.7907	0.7224
ML DeBERTa MFA	Taskbase SimpleK	0.9576	0.5600	0.4308	0.4870	0.9724	0.9834	0.9779
	Taskbase Buyer-Seller	0.9020	0.9545	0.9333	0.9438	0.5714	0.6667	0.6154
	Taskbase Evil Regular EN	0.8659	0.3133	0.3355	0.3240	0.9290	0.9221	0.9256
	Taskbase Evil Regular DE	0.8648	0.3225	0.3736	0.3462	0.9325	0.9169	0.9246
	Taskbase Evil Hard	0.5938	0.6667	0.5640	0.6110	0.5268	0.6326	0.5749
	MNLI validation combined	0.9389	0.9271	0.8976	0.9121	0.9450	0.9614	0.9531
	SNLI	0.9319	0.8936	0.9097	0.9016	0.9525	0.9435	0.9479
	XNLI	0.8849	0.8923	0.7445	0.8117	0.8820	0.9551	0.9171
	XNLI shuffled	0.8604	0.8563	0.6985	0.7694	0.8620	0.9414	0.8999
	ANLI Test R1	0.8040	0.6971	0.7305	0.7135	0.8615	0.8408	0.8511
	ANLI Test R2	0.6930	0.5385	0.5659	0.5518	0.7766	0.7568	0.7665
	ANLI Test R3	0.7033	0.5561	0.5672	0.5616	0.7797	0.7719	0.7758
RoBERTa LXA	Taskbase SimpleK	0.9158	0.3571	1.0000	0.5263	1.0000	0.9117	0.9538
	Taskbase Buyer-Seller	0.7386	0.9439	0.7481	0.8347	0.2609	0.6667	0.3750
	Taskbase Evil Regular EN	0.7260	0.1882	0.5610	0.2818	0.9411	0.7435	0.8307
	Taskbase Evil Regular DE	0.6698	0.1709	0.6349	0.2692	0.9457	0.6735	0.7867
	Taskbase Evil Hard	0.7763	0.7708	0.8605	0.8132	0.7857	0.6667	0.7213
	MNLI validation combined	0.9120	0.8684	0.8850	0.8766	0.9365	0.9267	0.9316
	SNLI	0.8987	0.8291	0.8875	0.8573	0.9391	0.9046	0.9215

Continued on next page

Table 21: Summary of benchmarked aggregated statistics for each model and dataset.

Model	Dataset	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	XNLI	0.9581	0.9519	0.9208	0.9361	0.9610	0.9767	0.9688
	XNLI shuffled	0.9440	0.9295	0.9003	0.9147	0.9509	0.9659	0.9583
	ANLI Test R1	0.7740	0.6350	0.7605	0.6921	0.8667	0.7808	0.8215
	ANLI Test R2	0.6830	0.5210	0.6317	0.5710	0.7933	0.7087	0.7486
	ANLI Test R3	0.6875	0.5308	0.5796	0.5541	0.7779	0.7419	0.7595
RoBERTa LX	Taskbase SimpleK	0.8712	0.2664	1.0000	0.4207	1.0000	0.8649	0.9276
	Taskbase Buyer-Seller	0.9150	0.9178	0.9926	0.9537	0.8571	0.3333	0.4800
	Taskbase Evil Regular EN	0.7620	0.2194	0.5800	0.3183	0.9461	0.7813	0.8558
	Taskbase Evil Regular DE	0.6730	0.1824	0.6929	0.2888	0.9537	0.6709	0.7877
	Taskbase Evil Hard	0.7566	0.7237	0.9215	0.8107	0.8412	0.5417	0.6590
	MNLI validation combined	0.9198	0.8546	0.9316	0.8914	0.9607	0.9134	0.9365
	SNLI	0.9015	0.8061	0.9382	0.8672	0.9648	0.8823	0.9217
	XNLI	0.9963	0.9939	0.9951	0.9945	0.9975	0.9969	0.9972
	XNLI shuffled	0.9912	0.9878	0.9858	0.9868	0.9929	0.9939	0.9934
	ANLI Test R1	0.5330	0.3382	0.4162	0.3732	0.6689	0.5916	0.6279
	ANLI Test R2	0.5440	0.3163	0.3144	0.3153	0.6572	0.6592	0.6582
	ANLI Test R3	0.5775	0.3615	0.3408	0.3508	0.6772	0.6967	0.6868
RoBERTa Ynie	Taskbase SimpleK	0.9108	0.3295	0.8769	0.4790	0.9934	0.9125	0.9512
	Taskbase Buyer-Seller	0.8889	0.9403	0.9333	0.9368	0.5263	0.5556	0.5405
	Taskbase Evil Regular EN	0.8616	0.3105	0.3646	0.3354	0.9314	0.9142	0.9227
	Taskbase Evil Regular DE	0.8917	0.3833	0.2145	0.2751	0.9205	0.9634	0.9415
	Taskbase Evil Hard	0.6184	0.6892	0.5930	0.6375	0.5513	0.6515	0.5972
	MNLI validation combined	0.9378	0.9335	0.8871	0.9097	0.9399	0.9654	0.9525
	SNLI	0.9489	0.9327	0.9172	0.9249	0.9572	0.9655	0.9613
	XNLI	0.8331	0.8697	0.5873	0.7011	0.8225	0.9560	0.8842
	XNLI shuffled	0.7374	0.8661	0.2509	0.3891	0.7236	0.9806	0.8327
	ANLI Test R1	0.8140	0.6947	0.7904	0.7395	0.8871	0.8258	0.8554

Continued on next page

Table 21: Summary of benchmarked aggregated statistics for each model and dataset.

		Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
Model	Dataset							
	ANLI Test R2	0.6590	0.4911	0.5808	0.5322	0.7686	0.6982	0.7317
	ANLI Test R3	0.6700	0.5071	0.5299	0.5182	0.7577	0.7406	0.7490

Table 22: Calculated statistics for the *full stop* experiment. Shown are statistics for every combination of model, dataset, and fullstop variation. Bolded values represent the maximum value across all variations for the same model and dataset. Green values show statistically significant increases over the null variation. Red values show statistically significant decreases.

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
AT-mT5	Taskbase	none	**0.9504**	0.4831	0.8769	0.6230	0.9937	**0.9540**	**0.9734**
	SimpleK	hypothesis	0.9324	0.3986	0.8769	0.5481	0.9936	0.9351	0.9635
		premise	**0.9504**	**0.4833**	**0.8923**	**0.6270**	**0.9945**	0.9532	0.9734
		both	0.9266	0.3791	**0.8923**	0.5321	0.9943	0.9283	0.9602
	Taskbase	none	0.7516	**0.9533**	0.7556	0.8430	**0.2826**	**0.7222**	**0.4062**
	Buyer-Seller	hypothesis	**0.7582**	0.9455	**0.7704**	**0.8490**	0.2791	0.6667	0.3934
		premise	0.7451	0.9528	0.7481	0.8382	0.2766	**0.7222**	0.4000
		both	**0.7582**	0.9455	**0.7704**	**0.8490**	0.2791	0.6667	0.3934
	Taskbase	none	**0.8673**	**0.3320**	0.3800	0.3544	0.9333	**0.9190**	**0.9261**
	Evil Regular	hypothesis	0.8607	0.3265	0.4268	**0.3700**	0.9372	0.9067	0.9217
	EN	premise	0.8639	0.3237	0.3859	0.3521	0.9336	0.9145	0.9240
		both	0.8552	0.3149	**0.4345**	0.3652	**0.9376**	0.8998	0.9183
	Taskbase	none	**0.8373**	**0.2862**	0.4669	0.3548	0.9395	**0.8766**	**0.9069**
	Evil Regular	hypothesis	0.8276	0.2814	0.5146	**0.3638**	0.9436	0.8607	0.9003
	DE	premise	0.8336	0.2787	0.4639	0.3482	0.9389	0.8728	0.9046
		both	0.8229	0.2755	**0.5208**	0.3604	**0.9439**	0.8549	0.8972
	Taskbase	none	0.6480	0.7097	0.6395	0.6728	0.5839	0.6591	0.6192
	Evil Hard	hypothesis	0.6530	0.7085	0.6570	0.6817	0.5917	0.6477	0.6184
		premise	**0.6612**	**0.7197**	0.6570	**0.6869**	**0.5986**	**0.6667**	**0.6308**
		both	0.6513	0.7050	**0.6599**	0.6817	0.5909	0.6402	0.6145

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	MNLI validation combined	none	0.9224	0.9184	0.8565	0.8864	0.9244	0.9584	0.9411
		hypothesis	**0.9242**	0.9098	0.8718	**0.8904**	0.9315	0.9528	**0.9420**
		premise	0.9225	**0.9188**	0.8562	0.8864	0.9243	**0.9587**	0.9412
		both	0.9240	0.9085	**0.8727**	0.8902	**0.9319**	0.9520	0.9418
	SNLI	none	**0.9033**	**0.8882**	0.8213	0.8534	0.9103	**0.9461**	**0.9278**
		hypothesis	0.9028	0.8725	0.8391	0.8555	0.9177	0.9360	0.9268
		premise	0.9028	0.8868	0.8213	0.8528	0.9102	0.9453	0.9274
		both	0.9031	0.8717	**0.8412**	**0.8561**	**0.9186**	0.9354	0.9269
	XNLI	none	0.8903	0.9095	0.7449	0.8190	0.8830	0.9629	0.9213
		hypothesis	0.8977	0.9009	0.7786	0.8353	0.8963	0.9572	0.9258
		premise	0.8904	**0.9102**	0.7446	0.8191	0.8830	**0.9632**	0.9214
		both	**0.8985**	0.9023	**0.7799**	**0.8367**	**0.8970**	0.9578	**0.9264**
	XNLI shuffled	none	0.8938	0.8884	0.7792	0.8302	0.8960	0.9510	0.9227
		hypothesis	**0.8991**	0.8781	**0.8096**	**0.8424**	**0.9084**	0.9438	**0.9257**
		premise	0.8938	**0.8913**	0.7760	0.8297	0.8948	**0.9527**	0.9229
		both	0.8986	0.8781	0.8079	0.8416	0.9077	0.9439	0.9254
RoBERTa	Taskbase SimpleK	none	0.7338	0.1378	**0.8923**	0.2387	0.9928	0.7260	0.8387
		hypothesis	**0.7518**	**0.1465**	**0.8923**	**0.2516**	**0.9930**	**0.7449**	**0.8512**
		premise	0.7381	0.1398	**0.8923**	0.2417	0.9928	0.7306	0.8417
		both	0.7331	0.1374	**0.8923**	0.2382	0.9928	0.7253	0.8382
	Taskbase Buyer-Seller	none	**0.7778**	**0.9106**	**0.8296**	**0.8682**	**0.2333**	**0.3889**	**0.2917**
		hypothesis	0.7320	0.9052	0.7778	0.8367	0.1892	**0.3889**	0.2545
		premise	0.7451	0.9068	0.7926	0.8458	0.2000	**0.3889**	0.2642
		both	0.7255	0.9043	0.7704	0.8320	0.1842	**0.3889**	0.2500
	Taskbase Evil Regular EN	none	0.7624	0.2120	0.5448	0.3053	0.9421	0.7854	0.8567
		hypothesis	0.7464	0.2071	0.5825	0.3056	0.9452	0.7637	0.8449
		premise	**0.7792**	**0.2199**	0.5118	**0.3076**	0.9398	**0.8076**	**0.8687**
		both	0.7344	0.2010	**0.5960**	0.3007	**0.9459**	0.7490	0.8360

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase Evil Regular DE	none	0.7626	0.2022	**0.5016**	0.2882	**0.9374**	0.7903	0.8575
		hypothesis	0.7873	0.2152	0.4609	**0.2934**	0.9350	0.8219	0.8748
		premise	**0.8141**	**0.2289**	0.3966	0.2902	0.9307	**0.8584**	**0.8931**
		both	0.7873	0.2149	0.4596	0.2928	0.9349	0.8220	0.8748
	Taskbase Evil Hard	none	0.6086	0.6587	0.6395	0.6490	0.5474	0.5682	0.5576
		hypothesis	**0.6250**	**0.6758**	0.6483	0.6617	**0.5647**	0.5947	**0.5793**
		premise	0.5987	0.6613	0.5959	0.6269	0.5336	**0.6023**	0.5658
		both	0.6217	0.6696	**0.6541**	**0.6618**	0.5625	0.5795	0.5709
	MNLI validation combined	none	0.9338	0.8979	0.9169	0.9073	0.9541	0.9430	0.9485
		hypothesis	**0.9356**	0.9007	0.9190	**0.9098**	0.9553	0.9447	**0.9499**
		premise	0.9341	**0.9023**	0.9123	0.9072	0.9518	**0.9460**	0.9489
		both	0.9349	0.8977	**0.9205**	0.9090	**0.9559**	0.9427	0.9493
	SNLI	none	0.9280	0.8749	0.9219	0.8978	0.9581	0.9312	0.9445
		hypothesis	0.9284	0.8750	0.9231	0.8984	0.9587	0.9312	0.9448
		premise	**0.9289**	**0.8799**	0.9181	**0.8986**	0.9563	**0.9346**	**0.9453**
		both	0.9278	0.8715	**0.9261**	0.8979	**0.9601**	0.9287	0.9442
	XNLI	none	0.8365	0.7793	**0.7108**	0.7434	0.8615	0.8993	0.8800
		hypothesis	0.8410	0.7952	0.7043	0.7470	0.8602	0.9093	0.8840
		premise	0.8395	**0.8046**	0.6847	0.7398	0.8533	**0.9168**	0.8839
		both	**0.8424**	0.7961	0.7085	**0.7498**	**0.8619**	0.9093	**0.8849**
	XNLI shuffled	none	0.7562	0.7957	0.3615	0.4972	0.7492	0.9536	0.8391
		hypothesis	0.7604	0.7994	0.3753	0.5108	0.7531	0.9529	0.8413
		premise	0.7518	**0.8144**	0.3310	0.4707	0.7420	**0.9623**	0.8379
		both	**0.7629**	0.7968	**0.3874**	**0.5214**	**0.7563**	0.9506	**0.8424**
ML mDeBERTa	Taskbase SimpleK	none	0.9604	0.6000	0.4615	0.5217	0.9739	0.9849	0.9794
		hypothesis	**0.9748**	**0.7419**	0.7077	**0.7244**	0.9857	**0.9879**	**0.9868**
		premise	0.9295	0.3613	0.6615	0.4674	0.9827	0.9426	0.9622
		both	0.8705	0.2444	**0.8462**	0.3793	**0.9914**	0.8717	0.9277

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase	none	0.7124	0.9333	0.7259	0.8167	0.2292	**0.6111**	0.3333
	Buyer-Seller	hypothesis	0.7190	**0.9340**	0.7333	0.8216	**0.2340**	**0.6111**	**0.3385**
		premise	0.7124	0.9333	0.7259	0.8167	0.2292	**0.6111**	0.3333
		both	**0.7320**	0.9273	**0.7556**	**0.8327**	0.2326	0.5556	0.3279
	Taskbase	none	0.8259	0.2379	0.3708	0.2899	0.9291	**0.8741**	**0.9008**
	Evil Regular	hypothesis	**0.8265**	**0.2507**	0.4077	0.3104	0.9328	0.8709	0.9007
	EN	premise	0.8154	0.2349	0.4109	0.2989	0.9322	0.8582	0.8937
		both	0.7729	0.2198	**0.5372**	**0.3119**	**0.9421**	0.7979	0.8640
	Taskbase	none	0.6569	0.1678	0.6518	0.2669	0.9469	0.6575	0.7761
	Evil Regular	hypothesis	**0.6580**	**0.1687**	0.6541	**0.2682**	0.9473	**0.6584**	**0.7769**
	DE	premise	0.6400	0.1645	0.6763	0.2647	0.9488	0.6361	0.7616
		both	0.6077	0.1605	**0.7316**	0.2633	**0.9544**	0.5946	0.7327
	Taskbase	none	0.6266	0.6789	0.6453	0.6617	0.5658	0.6023	0.5835
	Evil Hard	hypothesis	0.6530	0.6812	0.7267	0.7032	0.6100	0.5568	0.5822
		premise	0.6743	**0.7074**	0.7238	0.7155	0.6289	**0.6098**	**0.6192**
		both	**0.7023**	0.7022	**0.8227**	**0.7577**	**0.7024**	0.5455	0.6141
	MNLI	none	0.9239	0.9161	0.8637	0.8892	0.9278	0.9568	0.9421
	validation	hypothesis	0.9249	0.9051	0.8796	0.8922	0.9352	0.9496	0.9424
	combined	premise	0.9245	**0.9169**	0.8646	0.8900	0.9282	**0.9572**	0.9425
		both	**0.9264**	0.8998	**0.8908**	**0.8953**	**0.9407**	0.9458	**0.9432**
	SNLI	none	0.9088	0.8827	0.8465	0.8642	0.9216	0.9413	0.9313
		hypothesis	**0.9123**	0.8720	0.8720	0.8720	0.9332	0.9332	0.9332
		premise	0.9117	**0.8870**	0.8510	0.8686	0.9239	**0.9435**	**0.9336**
		both	0.9116	0.8619	**0.8839**	**0.8728**	**0.9386**	0.9261	0.9323
	XNLI	none	0.8966	0.8633	0.8196	0.8409	0.9120	0.9351	0.9234
		hypothesis	0.8980	0.8515	0.8406	0.8460	0.9208	0.9267	0.9237
		premise	0.8981	**0.8700**	0.8162	0.8422	0.9108	**0.9390**	0.9247
		both	**0.9007**	0.8519	**0.8499**	**0.8509**	**0.9250**	0.9261	**0.9256**

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	XNLI shuffled	none	0.8915	0.8458	0.8250	0.8353	0.9136	0.9248	0.9191
		hypothesis	0.8924	0.8345	0.8448	0.8396	0.9219	0.9162	0.9191
		premise	0.8923	**0.8543**	0.8160	0.8347	0.9100	**0.9304**	0.9201
		both	**0.8945**	0.8360	**0.8501**	**0.8430**	**0.9244**	0.9166	**0.9205**
RoBERTa LXA	Taskbase SimpleK	none	0.9151	0.3552	**1.0000**	0.5242	**1.0000**	0.9109	0.9534
		hypothesis	0.9115	0.3457	**1.0000**	0.5138	**1.0000**	0.9072	0.9513
		premise	**0.9187**	**0.3652**	**1.0000**	**0.5350**	**1.0000**	**0.9147**	**0.9555**
		both	0.9065	0.3333	**1.0000**	0.5000	**1.0000**	0.9019	0.9484
	Taskbase Buyer-Seller	none	0.7582	0.9455	0.7704	0.8490	0.2791	**0.6667**	0.3934
		hypothesis	0.7386	0.9439	0.7481	0.8347	0.2609	**0.6667**	0.3750
		premise	**0.7712**	**0.9464**	**0.7852**	**0.8583**	**0.2927**	**0.6667**	**0.4068**
		both	0.7386	0.9439	0.7481	0.8347	0.2609	**0.6667**	0.3750
	Taskbase Evil Regular EN	none	0.7222	0.1860	0.5627	0.2796	0.9410	0.7391	0.8279
		hypothesis	0.7209	0.1858	**0.5653**	0.2796	0.9412	0.7374	0.8269
		premise	**0.7321**	**0.1906**	0.5529	**0.2834**	0.9407	**0.7511**	**0.8353**
		both	0.7246	0.1879	0.5642	0.2819	**0.9414**	0.7415	0.8296
	Taskbase Evil Regular DE	none	0.6668	0.1697	0.6364	0.2680	0.9456	0.6701	0.7843
		hypothesis	0.6743	0.1735	**0.6373**	0.2727	0.9464	0.6782	0.7902
		premise	**0.6814**	**0.1754**	0.6281	0.2742	0.9458	**0.6871**	**0.7959**
		both	0.6775	0.1749	0.6362	**0.2743**	**0.9465**	0.6818	0.7927
	Taskbase Evil Hard	none	0.7681	0.7609	0.8605	0.8076	0.7808	0.6477	0.7081
		hypothesis	**0.7780**	0.7714	**0.8634**	**0.8148**	**0.7892**	0.6667	0.7228
		premise	0.7681	0.7678	0.8459	0.8050	0.7686	0.6667	0.7140
		both	**0.7780**	**0.7728**	0.8605	0.8143	0.7867	**0.6705**	**0.7239**
	MNLI validation combined	none	0.9115	0.8687	0.8832	0.8759	0.9356	0.9270	0.9313
		hypothesis	0.9119	0.8689	**0.8842**	0.8765	**0.9361**	0.9271	0.9316
		premise	**0.9125**	**0.8706**	0.8836	**0.8770**	0.9359	**0.9282**	**0.9320**
		both	0.9114	0.8679	0.8839	0.8758	0.9359	0.9265	0.9312

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	SNLI	none	0.8995	0.8301	0.8890	0.8585	0.9398	0.9050	0.9221
		hypothesis	**0.8997**	0.8303	**0.8893**	**0.8588**	**0.9400**	0.9052	0.9223
		premise	**0.8997**	**0.8305**	0.8890	0.8587	0.9399	**0.9054**	**0.9223**
		both	**0.8997**	0.8303	**0.8893**	**0.8588**	**0.9400**	0.9052	0.9223
	XNLI	none	0.9561	0.9509	0.9156	0.9329	0.9586	0.9763	0.9674
		hypothesis	0.9570	0.9502	0.9193	0.9345	0.9603	0.9759	0.9680
		premise	0.9579	**0.9538**	0.9181	0.9356	0.9598	**0.9778**	0.9687
		both	**0.9582**	0.9519	**0.9211**	**0.9362**	**0.9612**	0.9767	**0.9689**
	XNLI shuffled	none	0.9402	0.9247	0.8933	0.9087	0.9475	0.9636	0.9555
		hypothesis	0.9439	0.9300	0.8993	0.9144	0.9504	0.9662	0.9582
		premise	0.9434	**0.9305**	0.8973	0.9136	0.9496	**0.9665**	0.9579
		both	**0.9440**	0.9295	**0.9001**	**0.9146**	**0.9509**	0.9659	**0.9583**
RoBERTa LX	Taskbase SimpleK	none	0.8698	0.2642	**1.0000**	0.4180	**1.0000**	0.8634	0.9267
		hypothesis	0.8504	0.2381	**1.0000**	0.3846	**1.0000**	0.8430	0.9148
		premise	**0.8791**	**0.2790**	**1.0000**	**0.4362**	**1.0000**	**0.8732**	**0.9323**
		both	0.8446	0.2313	**1.0000**	0.3757	**1.0000**	0.8370	0.9113
	Taskbase Buyer-Seller	none	0.9020	0.9167	0.9778	0.9462	0.6667	0.3333	0.4444
		hypothesis	0.8954	0.9103	0.9778	0.9429	0.6250	0.2778	0.3846
		premise	0.9085	0.9231	0.9778	0.9496	0.7000	**0.3889**	0.5000
		both	**0.9216**	**0.9241**	**0.9926**	**0.9571**	**0.8750**	**0.3889**	**0.5385**
	Taskbase Evil Regular EN	none	0.7627	0.2203	0.5815	0.3195	0.9463	0.7819	0.8563
		hypothesis	0.7299	0.2050	0.6317	0.3095	0.9499	0.7403	0.8321
		premise	**0.7697**	**0.2237**	0.5685	**0.3211**	0.9453	**0.7910**	**0.8613**
		both	0.7321	0.2069	**0.6337**	0.3119	**0.9503**	0.7426	0.8337
	Taskbase Evil Regular DE	none	0.6717	0.1823	0.6963	0.2890	0.9541	0.6691	0.7866
		hypothesis	0.6494	0.1754	**0.7186**	0.2820	0.9556	0.6421	0.7681
		premise	**0.6852**	**0.1881**	0.6892	**0.2956**	0.9541	**0.6848**	**0.7973**
		both	0.6561	0.1782	0.7169	0.2855	**0.9559**	0.6497	0.7736

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase Evil Hard	none	0.7533	0.7256	0.9070	0.8062	0.8202	0.5530	0.6606
		hypothesis	**0.7582**	0.7213	**0.9331**	**0.8137**	**0.8589**	0.5303	0.6557
		premise	0.7516	**0.7260**	0.9012	0.8042	0.8122	**0.5568**	**0.6607**
		both	**0.7582**	0.7244	0.9244	0.8123	0.8462	0.5417	0.6605
	MNLI validation combined	none	**0.9225**	**0.8637**	0.9271	**0.8943**	0.9585	**0.9200**	**0.9389**
		hypothesis	0.9202	0.8553	0.9317	0.8919	0.9608	0.9139	0.9367
		premise	0.9218	0.8632	0.9252	0.8931	0.9575	0.9199	0.9383
		both	0.9192	0.8531	**0.9319**	0.8907	**0.9608**	0.9123	0.9359
	SNLI	none	0.9014	0.8089	0.9326	0.8664	0.9618	0.8851	0.9218
		hypothesis	0.9014	0.8039	**0.9421**	**0.8675**	**0.9668**	0.8801	0.9214
		premise	**0.9015**	**0.8098**	0.9314	0.8663	0.9612	**0.8858**	**0.9220**
		both	0.9007	0.8031	0.9409	0.8666	0.9661	0.8796	0.9209
	XNLI	none	**0.9965**	**0.9946**	0.9949	**0.9948**	0.9975	**0.9973**	**0.9974**
		hypothesis	0.9964	0.9937	**0.9954**	0.9945	**0.9977**	0.9969	0.9973
		premise	0.9962	0.9945	0.9942	0.9943	0.9971	0.9972	0.9972
		both	0.9963	0.9939	0.9951	0.9945	0.9975	0.9969	0.9972
	XNLI shuffled	none	0.9907	0.9892	0.9829	0.9860	0.9915	0.9946	0.9930
		hypothesis	0.9911	0.9880	0.9852	0.9866	0.9926	0.9940	0.9933
		premise	0.9907	**0.9896**	0.9825	0.9860	0.9913	**0.9948**	0.9931
		both	**0.9913**	0.9879	**0.9861**	**0.9870**	**0.9930**	0.9939	**0.9935**

Table 23: p-values for the *full stop* experiment. Shown are p-values for every combination of model, dataset, and fullstop variation except for the null variation. Green values show $p < 0.05$. Red values shows $p > 0.95$. Significance for F_1 scores was not measured.

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
AT-mT5	Taskbase SimpleK	hypothesis	0.9990	0.9746	0.5000	0.5225	0.9995
		premise	0.5000	0.4974	0.3529	0.3637	0.5521
		both	1.0000	0.9919	0.3529	0.3887	1.0000
	Taskbase Buyer-Seller	hypothesis	0.4258	0.6500	0.3444	0.5210	0.7006
		premise	0.5742	0.5087	0.5794	0.5356	0.5000
		both	0.4258	0.6500	0.3444	0.5210	0.7006
	Taskbase Evil Regular EN	hypothesis	1.0000	0.8145	0.0000	0.0005	1.0000
		premise	0.9878	0.9132	0.1999	0.4017	0.9997
		both	1.0000	0.9974	0.0000	0.0002	1.0000
	Taskbase Evil Regular DE	hypothesis	1.0000	0.8311	0.0000	0.0002	1.0000
		premise	0.9871	0.9321	0.6589	0.6842	0.9924
		both	1.0000	0.9836	0.0000	0.0001	1.0000
	Taskbase Evil Hard	hypothesis	0.3995	0.5190	0.2502	0.3934	0.6516
		premise	0.2485	0.3466	0.2502	0.3047	0.3976
		both	0.4326	0.5732	0.2159	0.4039	0.7419
	MNLI validation combined	hypothesis	0.1823	0.9947	0.0001	0.0010	0.9993
		premise	0.4894	0.4528	0.5273	0.5212	0.4470
		both	0.2118	0.9984	0.0001	0.0006	0.9999
	SNLI	hypothesis	0.5678	0.9976	0.0035	0.0173	0.9998
		premise	0.5678	0.6006	0.5000	0.5076	0.6086
		both	0.5272	0.9985	0.0013	0.0087	0.9999

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.0004	0.9877	0.0000	0.0000	0.9998
		premise	0.4820	0.4343	0.5224	0.5133	0.4273
		both	0.0001	0.9701	0.0000	0.0000	0.9992
	XNLI	hypothesis	0.0075	0.9944	0.0000	0.0000	0.9999
	shuffled	premise	0.4909	0.2336	0.7322	0.6740	0.1889
		both	0.0131	0.9941	0.0000	0.0000	0.9999
RoBERTa	Taskbase	hypothesis	0.0646	0.3039	0.5000	0.4733	0.0618
	SimpleK	premise	0.3579	0.4532	0.5000	0.4935	0.3558
		both	0.5242	0.5077	0.5000	0.5011	0.5246
	Taskbase	hypothesis	0.9133	0.5814	0.9455	0.7315	0.5000
	Buyer-Seller	premise	0.8346	0.5573	0.8738	0.6795	0.5000
		both	0.9401	0.5936	0.9665	0.7540	0.5000
	Taskbase	hypothesis	1.0000	0.9088	0.0000	0.0055	1.0000
	Evil Regular	premise	0.0000	0.0155	1.0000	0.9724	0.0000
	EN	both	1.0000	0.9987	0.0000	0.0010	1.0000
	Taskbase	hypothesis	0.0000	0.0006	1.0000	0.9718	0.0000
	Evil Regular	premise	0.0000	0.0000	1.0000	1.0000	0.0000
	DE	both	0.0000	0.0008	1.0000	0.9782	0.0000
	Taskbase	hypothesis	0.2030	0.2573	0.3681	0.2802	0.1922
	Evil Hard	premise	0.6910	0.4604	0.9539	0.6799	0.1317
		both	0.2531	0.3379	0.2872	0.3062	0.3547
	MNLI	hypothesis	0.1508	0.2120	0.2572	0.2575	0.2108
	validation	premise	0.4317	0.1110	0.9179	0.8889	0.0729
	combined	both	0.2734	0.5162	0.1385	0.1559	0.5608
	SNLI	hypothesis	0.4380	0.4899	0.3986	0.4046	0.5000
		premise	0.3627	0.1833	0.7981	0.7650	0.1396
		both	0.5311	0.7311	0.1843	0.2099	0.7843

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.0428	0.0016	0.8771	0.6750	0.0001
		premise	0.1259	0.0000	1.0000	0.9976	0.0000
		both	0.0121	0.0009	0.6572	0.4466	0.0001
	XNLI shuffled	hypothesis	0.0860	0.3062	0.0096	0.1177	0.6444
		premise	0.9262	0.0054	1.0000	0.9842	0.0000
		both	0.0143	0.4413	0.0000	0.0161	0.9500
ML mDeBERTa	Taskbase SimpleK	hypothesis	0.0030	0.0010	0.0000	0.0041	0.1837
		premise	1.0000	1.0000	0.0006	0.0242	1.0000
		both	1.0000	1.0000	0.0000	0.0000	1.0000
	Taskbase Buyer-Seller	hypothesis	0.4291	0.4896	0.4235	0.4685	0.5000
		premise	0.5000	0.5000	0.5000	0.5000	0.5000
		both	0.2960	0.5990	0.2201	0.4781	0.6856
	Taskbase Evil Regular EN	hypothesis	0.3694	0.0027	0.0000	0.0022	0.9811
		premise	1.0000	0.7424	0.0000	0.0084	1.0000
		both	1.0000	1.0000	0.0000	0.0000	1.0000
	Taskbase Evil Regular DE	hypothesis	0.3104	0.3717	0.3681	0.3767	0.3407
		premise	1.0000	0.8876	0.0002	0.0647	1.0000
		both	1.0000	0.9965	0.0000	0.0000	1.0000
	Taskbase Evil Hard	hypothesis	0.0899	0.4627	0.0008	0.0814	0.9344
		premise	0.0075	0.1228	0.0012	0.0230	0.4007
		both	0.0001	0.1708	0.0000	0.0000	0.9704
	MNLI validation combined	hypothesis	0.3046	0.9994	0.0001	0.0006	1.0000
		premise	0.3836	0.4095	0.4169	0.4211	0.4137
		both	0.0983	1.0000	0.0000	0.0000	1.0000
	SNLI	hypothesis	0.1167	0.9715	0.0000	0.0002	0.9970
		premise	0.1547	0.2170	0.2367	0.2486	0.2293
		both	0.1632	0.9999	0.0000	0.0000	1.0000

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.2580	0.9970	0.0000	0.0002	1.0000
		premise	0.2505	0.0595	0.7679	0.6879	0.0339
		both	0.0286	0.9962	0.0000	0.0000	1.0000
	XNLI shuffled	hypothesis	0.3413	0.9943	0.0000	0.0003	0.9999
		premise	0.3667	0.0280	0.9733	0.9277	0.0069
		both	0.0901	0.9860	0.0000	0.0000	0.9998
RoBERTa LXA	Taskbase SimpleK	hypothesis	0.6848	0.6061	nan	nan	0.6852
		premise	0.3152	0.3881	nan	nan	0.3148
		both	0.8759	0.7333	nan	nan	0.8765
	Taskbase Buyer-Seller	hypothesis	0.7144	0.5280	0.7304	0.6061	0.5000
		premise	0.3529	0.4821	0.3412	0.4202	0.5000
		both	0.7144	0.5280	0.7304	0.6061	0.5000
	Taskbase Evil Regular EN	hypothesis	0.7308	0.5324	0.3620	0.4366	0.7852
		premise	0.0000	0.0832	0.9120	0.6042	0.0000
		both	0.1190	0.2860	0.4184	0.3831	0.1170
	Taskbase Evil Regular DE	hypothesis	0.0002	0.0949	0.4517	0.2805	0.0001
		premise	0.0000	0.0237	0.8816	0.4605	0.0000
		both	0.0000	0.0364	0.5121	0.2495	0.0000
	Taskbase Evil Hard	hypothesis	0.2821	0.3147	0.4382	0.3804	0.2597
		premise	0.5000	0.3759	0.7817	0.6714	0.2597
		both	0.2821	0.2920	0.5000	0.4163	0.2197
	MNLI validation combined	hypothesis	0.4204	0.4749	0.3968	0.4051	0.4864
		premise	0.3256	0.3178	0.4554	0.4456	0.3044
		both	0.5200	0.5757	0.4259	0.4390	0.5944
	SNLI	hypothesis	0.4732	0.4823	0.4781	0.4787	0.4831
		premise	0.4732	0.4707	0.5000	0.4974	0.4662
		both	0.4732	0.4823	0.4781	0.4787	0.4831

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.2562	0.6015	0.1356	0.1535	0.6337
		premise	0.1073	0.1374	0.2272	0.2314	0.1397
		both	0.0738	0.3528	0.0517	0.0621	0.3880
	XNLI	hypothesis	0.0138	0.0522	0.0565	0.0633	0.0581
	shuffled	premise	0.0264	0.0395	0.1423	0.1446	0.0395
		both	0.0118	0.0710	0.0342	0.0410	0.0828
RoBERTa LX	Taskbase	hypothesis	0.9843	0.8296	nan	nan	0.9846
	SimpleK	premise	0.1501	0.2955	nan	nan	0.1492
		both	0.9974	0.8848	nan	nan	0.9974
	Taskbase	hypothesis	0.6071	0.6082	0.5000	0.6031	0.6915
	Buyer-Seller	premise	0.3929	0.3903	0.5000	0.4172	0.3085
		both	0.2074	0.3727	0.1215	0.0956	0.3085
	Taskbase	hypothesis	1.0000	1.0000	0.0000	0.0012	1.0000
	Evil Regular	premise	0.0001	0.1657	0.9644	0.7921	0.0000
	EN	both	1.0000	0.9999	0.0000	0.0004	1.0000
	Taskbase	hypothesis	1.0000	0.9919	0.0004	0.1012	1.0000
	Evil Regular	premise	0.0000	0.0207	0.8525	0.4947	0.0000
	DE	both	1.0000	0.9242	0.0010	0.0699	1.0000
	Taskbase	hypothesis	0.3889	0.5784	0.0474	0.0928	0.7712
	Evil Hard	premise	0.5375	0.4923	0.6448	0.6088	0.4507
		both	0.3889	0.5225	0.1327	0.1874	0.6448
	MNLI	hypothesis	0.8902	0.9823	0.0698	0.1049	0.9946
	validation	premise	0.6555	0.5473	0.7258	0.7149	0.5261
	combined	both	0.9586	0.9962	0.0638	0.1034	0.9993
	SNLI	hypothesis	0.5000	0.7872	0.0139	0.0219	0.8941
		premise	0.4865	0.4464	0.6083	0.5973	0.4227
		both	0.5936	0.8227	0.0272	0.0404	0.9140

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.6403	0.8398	0.3030	0.3045	0.8417
		premise	0.7637	0.5678	0.8050	0.8041	0.5663
		both	0.6840	0.7966	0.4317	0.4328	0.7980
	XNLI shuffled	hypothesis	0.3030	0.8202	0.0782	0.0816	0.8278
		premise	0.5000	0.3638	0.6116	0.6092	0.3615
		both	0.1883	0.8471	0.0236	0.0253	0.8562

Table 24: Calculated statistics for the *capitalization* experiment. Shown are statistics for every combination of model, dataset, and capitalization variation. Bold values represent the maximum value across all variations for the same model and dataset. Green values show statistically significant increases over the null variation. Red values show statistically significant decreases.

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
AT-mT5	Taskbase	none	0.9518	0.4915	**0.8923**	0.6339	0.9945	0.9547	0.9742
	SimpleK	hypothesis	**0.9540**	**0.5043**	**0.8923**	**0.6444**	**0.9945**	**0.9570**	**0.9754**
		premise	0.9496	0.4793	**0.8923**	0.6237	0.9945	0.9525	0.9730
		both	0.9482	0.4706	0.8615	0.6087	0.9929	0.9525	0.9723
	Taskbase	none	**0.7647**	**0.9541**	**0.7704**	**0.8525**	**0.2955**	**0.7222**	**0.4194**
	Buyer-Seller	hypothesis	0.7582	0.9455	**0.7704**	0.8490	0.2791	0.6667	0.3934
		premise	0.7582	0.9455	**0.7704**	0.8490	0.2791	0.6667	0.3934
		both	0.7582	0.9455	**0.7704**	0.8490	0.2791	0.6667	0.3934
	Taskbase	none	**0.8703**	**0.3416**	0.3815	**0.3604**	0.9336	0.9221	**0.9278**
	Evil Regular	hypothesis	0.8657	0.3279	0.3823	0.3530	0.9334	0.9170	0.9251
	EN	premise	0.8702	0.3404	0.3783	0.3583	0.9333	**0.9223**	0.9278
		both	0.8652	0.3279	**0.3874**	0.3552	**0.9338**	0.9159	0.9247
	Taskbase	none	0.8352	0.2851	0.4775	0.3570	0.9404	0.8731	0.9055
	Evil Regular	hypothesis	**0.8388**	**0.2908**	0.4741	0.3605	0.9403	**0.8774**	**0.9078**
	DE	premise	0.8342	0.2837	0.4790	0.3564	0.9405	0.8719	0.9049
		both	0.8379	0.2907	**0.4801**	**0.3621**	**0.9408**	0.8759	0.9072
	Taskbase	none	0.6464	0.6972	0.6500	0.6728	0.5911	0.6418	0.6154
	Evil Hard	hypothesis	**0.6513**	**0.7025**	**0.6529**	**0.6768**	**0.5959**	**0.6493**	**0.6214**
		premise	0.6431	0.6952	0.6441	0.6687	0.5870	0.6418	0.6132
		both	0.6414	0.6955	0.6382	0.6656	0.5845	0.6455	0.6135

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	MNLI validation combined	none	0.9238	0.9104	0.8698	0.8896	0.9305	0.9532	0.9418
		hypothesis	0.9240	0.9096	**0.8714**	0.8901	0.9313	0.9527	0.9419
		premise	**0.9241**	**0.9108**	0.8704	0.8901	0.9308	**0.9534**	**0.9420**
		both	**0.9241**	0.9099	**0.8714**	**0.8902**	**0.9313**	0.9529	0.9420
	SNLI	none	0.8992	**0.8601**	0.8373	0.8485	0.9183	**0.9307**	0.9245
		hypothesis	0.8986	0.8596	0.8357	0.8475	0.9176	0.9306	0.9241
		premise	**0.8995**	0.8585	**0.8406**	**0.8494**	**0.9198**	0.9295	**0.9246**
		both	0.8984	0.8593	0.8354	0.8472	0.9175	0.9304	0.9239
	XNLI	none	0.8977	**0.9034**	0.7759	0.8348	0.8953	**0.9585**	0.9259
		hypothesis	0.8981	0.9034	0.7772	0.8356	0.8959	0.9585	0.9261
		premise	0.8972	0.9016	0.7762	0.8342	0.8954	0.9576	0.9255
		both	**0.8987**	0.9033	**0.7793**	**0.8368**	**0.8968**	0.9583	**0.9265**
	XNLI shuffled	none	0.8962	0.8729	0.8058	0.8380	0.9065	0.9413	0.9236
		hypothesis	0.8968	0.8747	0.8057	0.8388	0.9065	0.9423	0.9241
		premise	0.8963	0.8726	**0.8066**	0.8383	0.9068	0.9411	0.9236
		both	**0.8971**	**0.8750**	**0.8066**	**0.8394**	**0.9069**	**0.9424**	**0.9243**
RoBERTa	Taskbase SimpleK	none	**0.7748**	**0.1593**	**0.8923**	**0.2704**	**0.9932**	**0.7691**	**0.8669**
		hypothesis	0.7144	0.1295	**0.8923**	0.2261	0.9926	0.7057	0.8249
		premise	0.7223	0.1327	**0.8923**	0.2311	0.9927	0.7140	0.8306
		both	0.6777	0.1162	**0.8923**	0.2057	0.9921	0.6672	0.7978
	Taskbase Buyer-Seller	none	**0.7386**	**0.9060**	**0.7852**	**0.8413**	**0.1944**	**0.3889**	**0.2593**
		hypothesis	0.7255	0.9043	0.7704	0.8320	0.1842	**0.3889**	0.2500
		premise	0.7320	0.9052	0.7778	0.8367	0.1892	**0.3889**	0.2545
		both	0.7255	0.9043	0.7704	0.8320	0.1842	**0.3889**	0.2500
	Taskbase Evil Regular EN	none	0.7667	0.2173	**0.5519**	**0.3119**	**0.9433**	0.7894	0.8595
		hypothesis	**0.7711**	**0.2185**	0.5391	0.3109	0.9422	0.7956	**0.8627**
		premise	0.7618	0.2119	0.5468	0.3054	0.9423	0.7845	0.8562
		both	0.7706	0.2163	0.5316	0.3075	0.9413	**0.7959**	0.8625

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase Evil Regular DE	none	**0.7804**	0.2048	0.4483	0.2812	0.9331	**0.8156**	**0.8704**
		hypothesis	0.7761	**0.2051**	**0.4647**	**0.2846**	**0.9345**	0.8091	0.8673
		premise	0.7779	0.2018	0.4462	0.2779	0.9327	0.8130	0.8687
		both	0.7750	0.2029	0.4603	0.2816	0.9339	0.8084	0.8666
	Taskbase Evil Hard	none	0.6217	0.6618	0.6618	0.6618	0.5709	0.5709	0.5709
		hypothesis	**0.6316**	**0.6737**	0.6618	0.6677	**0.5803**	**0.5933**	**0.5867**
		premise	0.6283	0.6657	**0.6735**	**0.6696**	0.5795	0.5709	0.5752
		both	0.6201	0.6627	0.6529	0.6578	0.5678	0.5784	0.5730
	MNLI validation combined	none	**0.9363**	**0.9034**	0.9179	**0.9106**	0.9547	**0.9464**	**0.9506**
		hypothesis	0.9359	0.9014	0.9192	0.9102	0.9554	0.9451	0.9502
		premise	0.9360	0.9019	0.9188	0.9102	0.9551	0.9454	0.9502
		both	0.9350	0.8990	**0.9193**	0.9091	**0.9554**	0.9436	0.9494
	SNLI	none	0.9226	**0.8592**	0.9215	0.8893	0.9585	**0.9232**	0.9406
		hypothesis	0.9225	0.8588	0.9218	0.8892	0.9587	0.9229	0.9405
		premise	0.9217	0.8557	**0.9236**	0.8883	**0.9595**	0.9208	0.9397
		both	**0.9230**	0.8592	0.9230	**0.8900**	0.9593	0.9231	**0.9408**
	XNLI	none	0.8426	0.8017	0.7010	0.7480	0.8594	0.9133	0.8855
		hypothesis	0.8416	0.7957	0.7060	0.7482	0.8608	0.9094	0.8844
		premise	**0.8429**	**0.8022**	0.7016	0.7485	0.8596	**0.9135**	**0.8857**
		both	0.8421	0.7958	**0.7081**	**0.7494**	**0.8617**	0.9091	0.8848
	XNLI shuffled	none	**0.7637**	**0.8050**	0.3843	0.5202	0.7559	**0.9534**	**0.8433**
		hypothesis	0.7634	0.8004	0.3867	0.5214	0.7563	0.9518	0.8429
		premise	0.7632	0.8034	0.3835	0.5192	0.7556	0.9531	0.8429
		both	0.7634	0.7993	**0.3874**	**0.5219**	**0.7565**	0.9513	0.8428
ML mDeBERTa	Taskbase SimpleK	none	0.9410	0.3867	0.4462	0.4143	0.9726	0.9653	0.9689
		hypothesis	0.9187	0.2895	0.5077	0.3687	0.9749	0.9389	0.9566
		premise	**0.9489**	**0.4531**	0.4462	**0.4496**	0.9729	**0.9736**	**0.9732**
		both	0.9259	0.3241	**0.5385**	0.4046	**0.9766**	0.9449	0.9605

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase	none	0.7255	0.9266	0.7481	0.8279	0.2273	0.5556	0.3226
	Buyer-Seller	hypothesis	**0.7320**	0.9273	**0.7556**	**0.8327**	0.2326	0.5556	0.3279
		premise	0.7255	**0.9346**	0.7407	0.8264	**0.2391**	**0.6111**	**0.3438**
		both	**0.7320**	0.9273	**0.7556**	**0.8327**	0.2326	0.5556	0.3279
	Taskbase	none	0.8332	0.2541	0.3827	0.3054	0.9309	0.8809	0.9052
	Evil Regular	hypothesis	0.8148	0.2413	**0.4354**	**0.3105**	**0.9346**	0.8550	0.8930
	EN	premise	**0.8383**	**0.2564**	0.3617	0.3001	0.9293	**0.8888**	**0.9086**
		both	0.8161	0.2385	0.4192	0.3040	0.9331	0.8582	0.8941
	Taskbase	none	0.6506	0.1671	0.6643	0.2670	0.9480	0.6491	0.7706
	Evil Regular	hypothesis	0.6369	0.1631	**0.6752**	0.2627	**0.9484**	0.6329	0.7592
	DE	premise	**0.6537**	**0.1677**	0.6594	**0.2673**	0.9476	**0.6531**	**0.7732**
		both	0.6384	0.1625	0.6682	0.2615	0.9475	0.6352	0.7606
	Taskbase	none	0.6546	0.6796	0.7235	0.7009	0.6179	0.5672	0.5914
	Evil Hard	hypothesis	**0.6924**	**0.6987**	**0.7912**	**0.7421**	**0.6816**	0.5672	**0.6191**
		premise	0.6546	0.6806	0.7206	0.7000	0.6169	**0.5709**	0.5930
		both	0.6793	0.6913	0.7706	0.7288	0.6594	0.5634	0.6076
	MNLI	none	0.9255	0.9033	0.8838	0.8934	0.9372	0.9483	0.9427
	validation	hypothesis	**0.9261**	0.9013	**0.8882**	**0.8947**	**0.9394**	0.9469	0.9431
	combined	premise	0.9251	**0.9046**	0.8809	0.8926	0.9358	**0.9492**	0.9425
		both	**0.9261**	0.9022	0.8872	0.8946	0.9389	0.9474	**0.9432**
	SNLI	none	0.9074	0.8528	0.8765	0.8645	0.9363	0.9231	0.9296
		hypothesis	0.9062	0.8488	**0.8783**	0.8633	**0.9370**	0.9205	0.9287
		premise	**0.9084**	**0.8555**	0.8762	**0.8658**	0.9363	**0.9248**	**0.9305**
		both	0.9069	0.8516	0.8765	0.8639	0.9362	0.9223	0.9292
	XNLI	none	0.9000	0.8540	0.8442	0.8491	0.9225	0.9278	0.9252
		hypothesis	**0.9006**	0.8523	**0.8490**	**0.8506**	**0.9246**	0.9264	**0.9255**
		premise	0.8997	**0.8544**	0.8425	0.8484	0.9218	**0.9282**	0.9250
		both	0.9004	0.8524	0.8481	0.8502	0.9242	0.9266	0.9254

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	XNLI shuffled	none	0.8943	0.8383	0.8461	0.8422	0.9227	0.9184	0.9205
		hypothesis	**0.8955**	0.8377	**0.8513**	**0.8445**	**0.9251**	0.9175	**0.9213**
		premise	0.8937	**0.8383**	0.8439	0.8411	0.9217	**0.9186**	0.9202
		both	0.8947	0.8369	0.8497	0.8433	0.9243	0.9172	0.9207
RoBERTa LXA	Taskbase SimpleK	none	0.9158	0.3571	**1.0000**	**0.5263**	**1.0000**	0.9117	0.9538
		hypothesis	**0.9165**	**0.3575**	0.9846	0.5246	0.9992	**0.9132**	**0.9543**
		premise	0.9151	0.3536	0.9846	0.5203	0.9992	0.9117	0.9534
		both	0.9144	0.3516	0.9846	0.5182	0.9992	0.9109	0.9530
	Taskbase Buyer-Seller	none	**0.7516**	**0.9450**	**0.7630**	**0.8443**	**0.2727**	**0.6667**	**0.3871**
		hypothesis	0.7451	0.9444	0.7556	0.8395	0.2667	**0.6667**	0.3810
		premise	**0.7516**	**0.9450**	**0.7630**	**0.8443**	**0.2727**	**0.6667**	**0.3871**
		both	0.7386	0.9439	0.7481	0.8347	0.2609	**0.6667**	0.3750
	Taskbase Evil Regular EN	none	0.7209	0.1849	**0.5614**	0.2782	**0.9407**	0.7378	0.8270
		hypothesis	0.7224	0.1852	0.5580	0.2781	0.9405	0.7398	0.8281
		premise	0.7179	0.1824	0.5580	0.2749	0.9401	0.7349	0.8249
		both	**0.7230**	**0.1859**	0.5597	**0.2791**	0.9407	**0.7403**	**0.8286**
	Taskbase Evil Regular DE	none	0.6711	0.1710	0.6322	**0.2692**	0.9454	0.6753	0.7878
		hypothesis	**0.6742**	**0.1714**	0.6258	0.2691	0.9448	**0.6794**	**0.7904**
		premise	0.6697	0.1705	**0.6334**	0.2687	**0.9455**	0.6735	0.7867
		both	0.6725	0.1708	0.6275	0.2685	0.9449	0.6772	0.7890
	Taskbase Evil Hard	none	0.7664	0.7578	**0.8559**	0.8039	**0.7812**	0.6530	0.7114
		hypothesis	**0.7714**	**0.7666**	0.8500	**0.8061**	0.7792	**0.6716**	**0.7214**
		premise	0.7697	0.7646	0.8500	0.8050	0.7783	0.6679	0.7189
		both	0.7681	0.7639	0.8471	0.8033	0.7749	0.6679	0.7174
	MNLI validation combined	none	**0.9132**	**0.8703**	0.8863	**0.8782**	0.9373	**0.9278**	**0.9325**
		hypothesis	0.9126	0.8697	0.8853	0.8774	0.9367	0.9275	0.9321
		premise	0.9130	0.8696	**0.8866**	0.8780	**0.9374**	0.9274	0.9323
		both	0.9120	0.8685	0.8849	0.8766	0.9365	0.9268	0.9316

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	SNLI	none	**0.8953**	**0.8165**	**0.8892**	**0.8513**	**0.9410**	**0.8983**	**0.9192**
		hypothesis	0.8940	0.8139	0.8889	0.8498	0.9407	0.8967	0.9182
		premise	0.8941	0.8140	**0.8892**	0.8499	0.9409	0.8967	0.9182
		both	0.8933	0.8136	0.8868	0.8486	0.9397	0.8967	0.9176
	XNLI	none	0.9574	0.9518	0.9187	0.9349	0.9601	0.9767	0.9683
		hypothesis	0.9576	0.9517	0.9196	0.9354	0.9605	0.9766	0.9685
		premise	0.9580	0.9519	0.9207	0.9360	0.9610	0.9767	0.9688
		both	**0.9583**	**0.9526**	**0.9208**	**0.9364**	**0.9611**	**0.9771**	**0.9690**
	XNLI shuffled	none	0.9444	0.9271	0.9042	0.9155	0.9527	0.9644	0.9585
		hypothesis	0.9445	0.9263	0.9054	0.9157	0.9532	0.9640	0.9586
		premise	**0.9452**	**0.9277**	**0.9061**	**0.9168**	**0.9536**	**0.9647**	**0.9591**
		both	0.9442	0.9257	0.9051	0.9153	0.9531	0.9637	0.9584
RoBERTa LX	Taskbase SimpleK	none	**0.8691**	**0.2612**	0.9846	**0.4129**	0.9991	0.8634	**0.9263**
		hypothesis	0.8576	0.2471	**1.0000**	0.3963	**1.0000**	0.8506	0.9192
		premise	0.8676	0.2531	0.9385	0.3987	0.9965	**0.8642**	0.9256
		both	0.8568	0.2443	0.9846	0.3914	0.9991	0.8506	0.9189
	Taskbase Buyer-Seller	none	0.9020	0.9110	0.9852	0.9466	0.7143	0.2778	0.4000
		hypothesis	0.9020	0.9110	0.9852	0.9466	0.7143	0.2778	0.4000
		premise	0.9085	0.9172	0.9852	0.9500	0.7500	**0.3333**	0.4615
		both	**0.9150**	**0.9178**	**0.9926**	**0.9537**	**0.8571**	**0.3333**	**0.4800**
	Taskbase Evil Regular EN	none	**0.7723**	**0.2274**	0.5740	**0.3257**	0.9462	**0.7933**	**0.8630**
		hypothesis	0.7588	0.2182	0.5874	0.3182	**0.9467**	0.7769	0.8535
		premise	0.7707	0.2245	0.5678	0.3218	0.9454	0.7922	0.8620
		both	0.7560	0.2161	**0.5885**	0.3161	0.9467	0.7738	0.8515
	Taskbase Evil Regular DE	none	0.6865	**0.1886**	0.6878	**0.2960**	**0.9540**	0.6864	0.7984
		hypothesis	0.6722	0.1821	**0.6933**	0.2884	0.9537	0.6700	0.7871
		premise	**0.6870**	0.1881	0.6833	0.2950	0.9535	**0.6874**	**0.7989**
		both	0.6700	0.1809	0.6931	0.2870	0.9535	0.6676	0.7853

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	F1 E	Prec. NE	Rec. NE	F1 NE
	Taskbase Evil Hard	none	0.7500	0.7136	**0.9235**	0.8051	0.8452	0.5299	0.6514
		hypothesis	0.7451	0.7107	0.9176	0.8010	0.8343	0.5261	0.6453
		premise	0.7516	**0.7172**	0.9176	0.8052	0.8382	**0.5410**	**0.6576**
		both	**0.7533**	0.7169	**0.9235**	**0.8072**	**0.8471**	0.5373	0.6575
	MNLI validation combined	none	0.9209	0.8581	0.9297	0.8925	0.9598	0.9160	0.9374
		hypothesis	0.9191	0.8532	0.9311	0.8905	0.9604	0.9125	0.9358
		premise	**0.9217**	**0.8593**	0.9307	**0.8936**	0.9603	**0.9167**	**0.9380**
		both	0.9198	0.8545	**0.9316**	0.8914	**0.9607**	0.9133	0.9364
	SNLI	none	0.8943	0.7896	0.9360	0.8566	0.9641	**0.8732**	0.9164
		hypothesis	0.8941	0.7877	**0.9390**	0.8567	**0.9656**	0.8713	0.9160
		premise	0.8933	0.7870	0.9372	0.8556	0.9646	0.8710	0.9154
		both	**0.8952**	**0.7900**	0.9384	**0.8579**	0.9654	**0.8732**	**0.9169**
	XNLI	none	0.9962	0.9936	0.9949	0.9942	0.9975	0.9968	0.9971
		hypothesis	**0.9963**	**0.9939**	0.9949	0.9944	0.9975	**0.9969**	**0.9972**
		premise	**0.9963**	0.9937	**0.9951**	**0.9944**	**0.9975**	0.9969	0.9972
		both	**0.9963**	**0.9939**	0.9949	0.9944	0.9975	**0.9969**	**0.9972**
	XNLI shuffled	none	0.9904	0.9862	0.9850	0.9856	0.9925	0.9931	0.9928
		hypothesis	0.9914	**0.9876**	0.9865	0.9870	0.9933	**0.9938**	0.9935
		premise	0.9906	0.9862	0.9856	0.9859	0.9928	0.9931	0.9930
		both	**0.9915**	0.9867	**0.9877**	**0.9872**	**0.9939**	0.9933	**0.9936**

Table 25: p-values for the *capitalization* experiment. Shown are p-values for every combination of model, dataset, and fullstop variation except for the null variation. Green values show $p < 0.05$. Red values shows $p > 0.95$. Significance for F_1 scores was not measured.

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
AT-mT5	Taskbase SimpleK	hypothesis	0.3536	0.3902	0.5000	0.4975	0.3459
		premise	0.6464	0.6045	0.5000	0.5025	0.6541
		both	0.7344	0.6756	0.7882	0.7766	0.6541
	Taskbase Buyer-Seller	hypothesis	0.5756	0.6680	0.5000	0.5934	0.7006
		premise	0.5756	0.6680	0.5000	0.5934	0.7006
		both	0.5756	0.6680	0.5000	0.5934	0.7006
	Taskbase Evil Regular EN	hypothesis	0.9986	0.9828	0.4522	0.5865	1.0000
		premise	0.5268	0.5759	0.6739	0.6009	0.4297
		both	0.9996	0.9827	0.2001	0.4399	1.0000
	Taskbase Evil Regular DE	hypothesis	0.0156	0.1327	0.6799	0.5292	0.0029
		premise	0.7164	0.6012	0.4190	0.4720	0.7796
		both	0.0501	0.1354	0.3629	0.3494	0.0396
	Taskbase Evil Hard	hypothesis	0.3996	0.4178	0.4547	0.4333	0.3994
		premise	0.5674	0.5296	0.5899	0.5559	0.5000
		both	0.6004	0.5254	0.6754	0.5910	0.4493
	MNLI validation combined	hypothesis	0.4572	0.5914	0.3474	0.3682	0.6157
		premise	0.4359	0.4627	0.4433	0.4471	0.4665
		both	0.4359	0.5608	0.3474	0.3665	0.5832
	SNLI	hypothesis	0.5797	0.5316	0.5930	0.5831	0.5195
		premise	0.4600	0.6063	0.3023	0.3351	0.6519
		both	0.6057	0.5518	0.6112	0.6006	0.5389

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.4260	0.5006	0.3959	0.4140	0.5173
		premise	0.5921	0.6827	0.4766	0.4942	0.6987
		both	0.3206	0.5087	0.2499	0.2896	0.5518
	XNLI	hypothesis	0.3906	0.3337	0.5123	0.4964	0.3161
	shuffled	premise	0.4815	0.5302	0.4385	0.4506	0.5440
		both	0.3300	0.3114	0.4385	0.4324	0.3032
RoBERTa	Taskbase	hypothesis	1.0000	0.9560	0.5000	0.5902	1.0000
	SimpleK	premise	1.0000	0.9358	0.5000	0.5777	1.0000
		both	1.0000	0.9931	0.5000	0.6509	1.0000
	Taskbase	hypothesis	0.6436	0.5240	0.6624	0.5627	0.5000
	Buyer-Seller	premise	0.5730	0.5119	0.5830	0.5323	0.5000
		both	0.6436	0.5240	0.6624	0.5627	0.5000
	Taskbase	hypothesis	0.0105	0.3836	0.9609	0.8192	0.0006
	Evil Regular	premise	0.9948	0.9234	0.7594	0.7849	0.9941
	EN	both	0.0196	0.6057	0.9973	0.9485	0.0004
	Taskbase	hypothesis	0.9884	0.4729	0.0119	0.1399	0.9998
	Evil Regular	premise	0.9120	0.7779	0.6154	0.6349	0.9187
	DE	both	0.9980	0.6917	0.0502	0.2637	1.0000
	Taskbase	hypothesis	0.3079	0.3220	0.5000	0.3776	0.2295
	Evil Hard	premise	0.3690	0.4392	0.3233	0.3870	0.5000
		both	0.5333	0.4857	0.6345	0.5414	0.4025
	MNLI	hypothesis	0.5924	0.7200	0.3470	0.3685	0.7485
	validation	premise	0.5810	0.6735	0.3965	0.4128	0.6957
	combined	both	0.7763	0.8973	0.3310	0.3668	0.9219
	SNLI	hypothesis	0.5151	0.5303	0.4742	0.4779	0.5371
		premise	0.6330	0.7299	0.3255	0.3504	0.7718
		both	0.4400	0.5030	0.3733	0.3817	0.5186

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.6510	0.8785	0.1889	0.3061	0.9484
		premise	0.4537	0.4685	0.4574	0.4643	0.4755
		both	0.5693	0.8754	0.1045	0.2150	0.9575
	XNLI shuffled	hypothesis	0.5397	0.7410	0.3437	0.4519	0.8168
		premise	0.5660	0.5887	0.5500	0.5357	0.5813
		both	0.5463	0.7923	0.2987	0.4355	0.8749
ML mDeBERTa	Taskbase SimpleK	hypothesis	0.9998	0.9710	0.1591	0.3058	1.0000
		premise	0.1052	0.0974	0.5000	0.4800	0.0494
		both	0.9916	0.8890	0.0672	0.1899	1.0000
	Taskbase Buyer-Seller	hypothesis	0.4281	0.4893	0.4214	0.4667	0.5000
		premise	0.5000	0.3748	0.5786	0.4256	0.3176
		both	0.4281	0.4893	0.4214	0.4667	0.5000
	Taskbase Evil Regular EN	hypothesis	1.0000	0.9947	0.0000	0.0015	1.0000
		premise	0.0011	0.3215	0.9985	0.9006	0.0000
		both	1.0000	0.9991	0.0000	0.0389	1.0000
	Taskbase Evil Regular DE	hypothesis	1.0000	0.9303	0.0575	0.3863	1.0000
		premise	0.0749	0.4201	0.7614	0.6275	0.0407
		both	1.0000	0.9537	0.2890	0.6527	1.0000
	Taskbase Evil Hard	hypothesis	0.0249	0.2146	0.0026	0.0219	0.5000
		premise	0.5000	0.4836	0.5483	0.5120	0.4509
		both	0.1004	0.3139	0.0262	0.0945	0.5491
	MNLI validation combined	hypothesis	0.3620	0.7057	0.1229	0.1542	0.7646
		premise	0.5860	0.3566	0.7730	0.7429	0.3153
		both	0.3620	0.6222	0.1844	0.2134	0.6703
	SNLI	hypothesis	0.6491	0.7436	0.3757	0.4081	0.7854
		premise	0.3639	0.3269	0.5211	0.5048	0.3045
		both	0.5691	0.5816	0.5000	0.5065	0.5919

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.3798	0.6555	0.1402	0.1820	0.7374
		premise	0.5562	0.4629	0.6447	0.6234	0.4336
		both	0.4161	0.6450	0.1902	0.2317	0.7151
	XNLI	hypothesis	0.2986	0.5576	0.1177	0.1535	0.6478
	shuffled	premise	0.6086	0.4988	0.6945	0.6704	0.4622
		both	0.4271	0.6232	0.2079	0.2481	0.6935
RoBERTa LXA	Taskbase	hypothesis	0.4615	0.4955	1.0000	1.0000	0.4232
	SimpleK	premise	0.5385	0.5397	1.0000	1.0000	0.5000
		both	0.5766	0.5613	1.0000	1.0000	0.5386
	Taskbase	hypothesis	0.5742	0.5093	0.5802	0.5363	0.5000
	Buyer-Seller	premise	0.5000	0.5000	0.5000	0.5000	0.5000
		both	0.6459	0.5187	0.6572	0.5707	0.5000
	Taskbase	hypothesis	0.2342	0.4720	0.6810	0.5882	0.1709
	Evil Regular	premise	0.9265	0.7838	0.6810	0.6971	0.9166
	EN	both	0.1498	0.3804	0.5930	0.5086	0.1153
	Taskbase	hypothesis	0.0733	0.4507	0.8180	0.6754	0.0332
	Evil Regular	premise	0.7526	0.5652	0.4280	0.4854	0.7827
	DE	both	0.2660	0.5271	0.7472	0.6540	0.1887
	Taskbase	hypothesis	0.3868	0.3452	0.6213	0.5296	0.2606
	Evil Hard	premise	0.4240	0.3797	0.6213	0.5436	0.3039
		both	0.4618	0.3906	0.6784	0.5920	0.3039
	MNLI	hypothesis	0.6098	0.5618	0.6044	0.5989	0.5545
	validation	premise	0.5404	0.5697	0.4699	0.4780	0.5815
	combined	both	0.7200	0.6723	0.6473	0.6461	0.6721
	SNLI	hypothesis	0.6537	0.6530	0.5221	0.5341	0.6740
		premise	0.6415	0.6500	0.5000	0.5140	0.6740
		both	0.7343	0.6735	0.6711	0.6700	0.6740

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.4306	0.5153	0.3941	0.4011	0.5229
		premise	0.3247	0.4855	0.2803	0.2922	0.5000
		both	0.2532	0.3762	0.2654	0.2750	0.3871
	XNLI	hypothesis	0.4754	0.5937	0.3697	0.3831	0.6104
	shuffled	premise	0.3110	0.4295	0.2945	0.3059	0.4443
		both	0.5491	0.6628	0.4015	0.4159	0.6798
RoBERTa LX	Taskbase	hypothesis	0.8984	0.6948	0.1568	0.1594	0.9131
	SimpleK	premise	0.5632	0.6155	0.9988	0.9985	0.4681
		both	0.9118	0.7302	0.5000	0.5060	0.9131
	Taskbase	hypothesis	0.5000	0.5000	0.5000	0.5000	0.5000
	Buyer-Seller	premise	0.3929	0.3950	0.5000	0.4157	0.2994
		both	0.2933	0.3858	0.2381	0.1973	0.2994
	Taskbase	hypothesis	1.0000	0.9927	0.0315	0.3164	1.0000
	Evil Regular	premise	0.8026	0.7741	0.8040	0.7546	0.7213
	EN	both	1.0000	0.9986	0.0224	0.3394	1.0000
	Taskbase	hypothesis	1.0000	0.9859	0.2064	0.5928	1.0000
	Evil Regular	premise	0.4038	0.5659	0.7458	0.6827	0.3188
	DE	both	1.0000	0.9951	0.2155	0.6545	1.0000
	Taskbase	hypothesis	0.6106	0.5540	0.6584	0.6531	0.5487
	Evil Hard	premise	0.4627	0.4337	0.6584	0.6008	0.3567
		both	0.4257	0.4400	0.5000	0.4738	0.4033
	MNLI	hypothesis	0.8225	0.8892	0.3194	0.3590	0.9250
	validation	premise	0.3362	0.3864	0.3712	0.3719	0.3867
	combined	both	0.7105	0.8157	0.2708	0.3034	0.8616
	SNLI	hypothesis	0.5262	0.6131	0.2389	0.2592	0.6725
		premise	0.6286	0.6549	0.3882	0.4071	0.6989
		both	0.3964	0.4737	0.2850	0.2947	0.5000

Continued on next page

Model	Dataset	Variation	Acc.	Prec. E	Rec. E	Prec. NE	Rec. NE
	XNLI	hypothesis	0.4097	0.3805	0.5000	0.4997	0.3800
		premise	0.4097	0.4392	0.4317	0.4318	0.4393
		both	0.4097	0.3805	0.5000	0.4997	0.3800
	XNLI shuffled	hypothesis	0.0841	0.1719	0.1568	0.1580	0.1732
		premise	0.3859	0.4977	0.3435	0.3451	0.5000
		both	0.0639	0.3684	0.0349	0.0362	0.3768

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F. Bold values are "interesting" cases.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
AT-mT5	Taskbase SimpleK	E	0.2672	0.6466	**0.0862**
		N	0.0318	0.7685	0.1997
		C	**0.0137**	0.3139	0.6724
	Taskbase Buyer-Seller	E	0.5000	0.5000	**0.0000**
		N	0.1304	0.8696	0.0000
		C	**0.1000**	0.4000	0.5000
	Taskbase Evil Regular EN	E	0.2175	0.6455	**0.1370**
		N	0.0505	0.7728	0.1767
		C	**0.0278**	0.4175	0.5547
	Taskbase Evil Regular DE	E	0.2577	0.6166	**0.1257**
		N	0.0715	0.7353	0.1932
		C	**0.0391**	0.3934	0.5675
	Taskbase Evil Hard	E	0.4528	0.5063	**0.0409**
		N	0.0573	0.9172	0.0255
		C	**0.0075**	0.2256	0.7669
	MNLI validation combined	E	0.3271	0.6547	**0.0182**
		N	0.1658	0.7606	0.0736
		C	**0.0107**	0.2907	0.6986
	SNLI	E	0.0874	0.8949	**0.0177**
		N	0.0978	0.7736	0.1286
		C	**0.0067**	0.2371	0.7562

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = entailment, N = neutral, C = contradiction). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

Model	Dataset	F	P(E|F)	P(N|F)	P(C|F)
	XNLI	E	0.1979	0.7756	**0.0265**
		N	0.0887	0.8295	0.0818
		C	**0.0089**	0.3796	0.6115
	XNLI shuffled	E	0.2171	0.7594	**0.0235**
		N	0.1000	0.8245	0.0754
		C	**0.0106**	0.3795	0.6099
RoBERTa	Taskbase SimpleK	E	0.1619	0.6381	**0.2000**
		N	0.1263	0.6018	0.2719
		C	**0.0700**	0.2525	0.6775
	Taskbase Buyer-Seller	E	0.6957	0.2783	**0.0261**
		N	0.8000	0.0000	0.2000
		C	**0.2727**	0.4545	0.2727
	Taskbase Evil Regular EN	E	0.2591	0.5677	**0.1732**
		N	0.1527	0.6742	0.1731
		C	**0.1437**	0.4059	0.4504
	Taskbase Evil Regular DE	E	0.1875	0.4865	**0.3260**
		N	0.1405	0.5463	0.3132
		C	**0.1064**	0.2923	0.6013
	Taskbase Evil Hard	E	0.5315	0.4324	**0.0360**
		N	0.2716	0.6049	0.1235
		C	**0.1340**	0.1598	0.7062

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
	MNLI validation combined	E	0.3438	0.6336	**0.0227**
		N	0.1948	0.7382	0.0669
		C	**0.0122**	0.1968	0.7910
	SNLI	E	0.0946	0.8864	**0.0191**
		N	0.1252	0.7950	0.0798
		C	**0.0089**	0.1167	0.8744
	XNLI	E	0.2157	0.7122	**0.0721**
		N	0.1177	0.7292	0.1531
		C	**0.0271**	0.3523	0.6207
	XNLI shuffled	E	0.2017	0.6922	**0.1061**
		N	0.0671	0.7120	0.2209
		C	**0.0256**	0.4586	0.5159
ML mDeBERTa	Taskbase SimpleK	E	0.4429	0.4000	**0.1571**
		N	0.0164	0.8501	0.1335
		C	**0.0058**	0.2168	0.7775
	Taskbase Buyer-Seller	E	0.7545	0.2455	**0.0000**
		N	0.4286	0.5000	0.0714
		C	**0.2069**	0.2069	0.5862
	Taskbase Evil Regular EN	E	0.2624	0.5968	**0.1409**
		N	0.0418	0.8304	0.1279
		C	**0.0323**	0.3543	0.6133

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
	Taskbase Evil Regular DE	E	0.3684	0.5046	**0.1270**
		N	0.1239	0.7641	0.1119
		C	**0.1108**	0.2897	0.5996
	Taskbase Evil Hard	E	0.5134	0.4812	**0.0054**
		N	0.0427	0.9231	0.0342
		C	**0.0084**	0.3025	0.6891
	MNLI validation combined	E	0.3350	0.6475	**0.0174**
		N	0.1542	0.7679	0.0778
		C	**0.0105**	0.2931	0.6964
	SNLI	E	0.0823	0.9007	**0.0170**
		N	0.1123	0.7765	0.1113
		C	**0.0093**	0.2422	0.7485
	XNLI	E	0.2333	0.7291	**0.0376**
		N	0.1206	0.7804	0.0990
		C	**0.0187**	0.3937	0.5877
	XNLI shuffled	E	0.2584	0.6975	**0.0441**
		N	0.1287	0.7651	0.1063
		C	**0.0215**	0.3776	0.6009
RoBERTa LXA	Taskbase SimpleK	E	0.2912	0.6319	**0.0769**
		N	0.0429	0.8064	0.1507
		C	**0.0179**	0.4464	0.5357

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
	Taskbase Buyer-Seller	E	0.7944	0.1963	**0.0093**
		N	0.5000	0.2500	0.2500
		C	**0.6471**	0.1765	0.1765
	Taskbase Evil Regular EN	E	0.2507	0.6243	**0.1251**
		N	0.1656	0.6939	0.1405
		C	**0.1127**	0.4310	0.4563
	Taskbase Evil Regular DE	E	0.3263	0.5390	**0.1347**
		N	0.1926	0.6688	0.1386
		C	**0.1142**	0.3931	0.4927
	Taskbase Evil Hard	E	0.3411	0.6458	**0.0130**
		N	0.1491	0.7105	0.1404
		C	**0.0000**	0.1000	0.9000
	MNLI validation combined	E	0.4578	0.5069	**0.0353**
		N	0.2824	0.6303	0.0873
		C	**0.0506**	0.2140	0.7354
	SNLI	E	0.1373	0.8200	**0.0427**
		N	0.2079	0.6699	0.1222
		C	**0.0401**	0.1979	0.7620
	XNLI	E	0.3793	0.5828	**0.0379**
		N	0.2663	0.6477	0.0859
		C	**0.0568**	0.2491	0.6940

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
	XNLI shuffled	E	0.4431	0.5150	**0.0419**
		N	0.2979	0.6189	0.0832
		C	**0.0737**	0.2682	0.6581
RoBERTa LX	Taskbase SimpleK	E	0.3238	0.6352	**0.0410**
		N	0.0714	0.7676	0.1610
		C	**0.0406**	0.4594	0.5000
	Taskbase Buyer-Seller	E	0.7808	0.2192	**0.0000**
		N	0.3333	0.6667	0.0000
		C	**0.2500**	0.5000	0.2500
	Taskbase Evil Regular EN	E	0.2708	0.6253	**0.1038**
		N	0.1192	0.7031	0.1777
		C	**0.0641**	0.4619	0.4740
	Taskbase Evil Regular DE	E	0.3495	0.5374	**0.1131**
		N	0.1904	0.6386	0.1710
		C	**0.0890**	0.4158	0.4953
	Taskbase Evil Hard	E	0.4201	0.5685	**0.0114**
		N	0.0714	0.6857	0.2429
		C	**0.0000**	0.2500	0.7500
	MNLI validation combined	E	0.3728	0.6084	**0.0188**
		N	0.2171	0.7038	0.0791
		C	**0.0205**	0.2677	0.7119

Continued on next page

Table 26: Summary of probabilities of backward entailment predictions, aggregated by model and dataset. F represents the prediction in the forward direction (E = ENTAILMENT, N = NEUTRAL, C = CONTRADICTION). $P(X|F)$ represents the probability of prediction X in the backward direction given the forward prediction F.

			P(E|F)	P(N|F)	P(C|F)
Model	Dataset	F			
	SNLI	E	0.1122	0.8740	**0.0138**
		N	0.1413	0.7896	0.0692
		C	**0.0155**	0.2603	0.7242
	XNLI	E	0.2682	0.7099	**0.0218**
		N	0.1573	0.7731	0.0696
		C	**0.0207**	0.2821	0.6972
	XNLI shuffled	E	0.2877	0.6853	**0.0270**
		N	0.1621	0.7657	0.0722
		C	**0.0289**	0.3007	0.6705

B The Taskbase Homer dataset

This is a listing of the final Taskbase Homer dataset used in the Homer Simpson Paradox experiment in Section 5.6 on page 60. The dataset is given in CSV format. It includes every pair that was collected throughout the experiment.

The purpose of this dataset is not to provide a comprehensive benchmark of the emergent behaviour of NLI models, but rather to demonstrate possible failure cases for further in-depth study.

```
premise,hypothesis,entailment
Homer Simpson works at a nuclear power plant.,Homer Simpson works
    at a nuclear power plant.
Homer Simpson parachutes at a nuclear power plant.,Homer Simpson
    works at a nuclear power plant.
Homer Simpson eats a sandwich at a nuclear power plant.,Homer
    Simpson works at a nuclear power plant.
Lisa works at a nuclear power plant and eats a sandwich with Homer
    Simpson.,Homer Simpson works at a power plant.
Homer Simpson is a worker at a nuclear power plant.,Homer Simpson
    works at a nuclear power plant.
Homer Simpson is a parachuter at a nuclear power plant.,Homer
    Simpson works at a nuclear power plant.
Homer Simpson fdgfwnqehfisf at a nuclear power plant.,Homer Simpson
     works at a nuclear power plant.
Lisa Simpson works at a nuclear power plant.,Homer Simpson works at
     a nuclear power plant.
Homer Simpson works at a solar power plant.,Homer Simpson works at
    a nuclear power plant.
Lisa Simpson works at a solar power plant.,Homer Simpson works at a
     nuclear power plant.
Homer Simpson works at a house plant.,Homer Simpson works at a
    nuclear power plant.
Lisa Simpson works at a house plant.,Homer Simpson works at a
    nuclear power plant.
Rhinos eat leaves that grow on trees.,Giraffes eat leaves that grow
     on trees.
Specimens eat leaves that grow on trees.,Giraffes eat leaves that
    grow on trees.
foobarbaz eat leaves that grow on trees.,Giraffes eat leaves that
    grow on trees.
Giraffes admire leaves that grow on trees.,Giraffes eat leaves that
     grow on trees.
Giraffes write leaves that grow on trees.,Giraffes eat leaves that
    grow on trees.
Giraffes iurehe leaves that grow on trees.,Giraffes eat leaves that
     grow on trees.
Giraffes eat twigs that grow on trees.,Giraffes eat leaves that
    grow on trees.
```

```
Giraffes eat empathies that grow on trees.,Giraffes eat leaves that
     grow on trees.
Giraffes eat qwfhkmko that grow on trees.,Giraffes eat leaves that
    grow on trees.
Giraffes eat leaves that fall on trees.,Giraffes eat leaves that
    grow on trees.
Giraffes eat leaves that think on trees.,Giraffes eat leaves that
    grow on trees.
Giraffes eat leaves that rwmxkjfhu on trees.,Giraffes eat leaves
    that grow on trees.
Giraffes eat leaves that grow on buildings.,Giraffes eat leaves
    that grow on trees.
Giraffes eat leaves that grow on ideas.,Giraffes eat leaves that
    grow on trees.
Giraffes eat leaves that grow on nvjoiej.,Giraffes eat leaves that
    grow on trees.
Usain Bolt runs at the speed of a sloth.,Usain Bolt runs quickly.
Usain Bolt runs at the speed of a tortoise.,Usain Bolt runs quickly
    .
Usain Bolt runs at the speed of a snail.,Usain Bolt runs quickly.
Usain Bolt runs at the speed of a cheetah.,Usain Bolt runs slowly.
Usain Bolt runs at the speed of a cheetah.,Usain Bolt runs quickly.
Usain Bolt runs like a sloth.,Usain Bolt runs quickly.
Usain Bolt runs like a tortoise.,Usain Bolt runs quickly.
Usain Bolt runs like a snail.,Usain Bolt runs quickly.
Usain Bolt runs like a cheetah.,Usain Bolt runs slowly.
Usain Bolt runs like a cheetah.,Usain Bolt runs quickly.
```

C Glossary

Closed-ended task **Tasks** which have a small space of correct answers that can be easily verified by a machine. Examples of closed-ended tasks are multiple-choice, fill-in-the-blank, or mathematics. See **Task**.

Compatibility See **Neutral**.

Contradiction A relationship between two texts A and B where B cannot reasonably be true given A.

Correct class The set of student responses to a task which entail the **correct hypothesis**. Not to be confused with the set of correct responses, as a response may belong to the correct class as well as a **mistake class**.

Digital learning platform An always-available computer, online, or mobile application that allows instructors to issue quizzes, assignments, exercises, or tests to students, and allows students to participate in these activities while possibly evaluating responses and/or giving feedback based on student responses.

Entailment A relationship between two texts A and B where the meaning of B can be inferred from the meaning of A.

Equivalence A relationship between two tets A and B where A entails B and B entails A.

Formative feedback Feedback given during the learning process, meant to change the way a student approaches a subject or to guide the student to a particular though process during learning.

Hypothesis (NLI) One of the two texts provided to an NLI odel, the other being the **premise**.

Hypothesis (task) A reference answer to a task. A **correct hypothesis** is a reference correct answer; a **mistake hypothesis** is a reference incorrect answer representing some mistake or misconception.

Mistake class The set of student responses to a task which entail a particular mistake hypothesis, i.e. answers with the same mistake or misconception as encoded in the mistake hypothesis.

Natural Language Inference (NLI) A subfield of **Natural Language Processing** which deals with recognizing **entailment** relationships between texts, that is, whether the truth of text A implies the truth of Text B.

Natural Language Processing (NLP) A field of computing that deals with handling human language.

Natural Language Understanding(NLU) A sub-field of **Natural Language Processing** which deals with teaching a machine to understand the semantics of human language. **NLI** is a crucial prerequisite to complete Natural Language Understanding.

Neutral A relationship between two texts A and BB where the truth of B cannot be totally inferred from the meaning of A; i.e. B may or may not be true given A.

NLI See **Natural Language Inference**.

NLI model A mechanism that, given a **premise** text p and **hypothesis** text h, determines if p **entails**, **contradicts**, or is **compatible** with h.

NLP See **Natural Language Processing**.

NLU See **Natural Language Understanding**.

Open-ended task **Tasks** to which students can respond in free-form text. Often there is more than correct answer and more than one way to formulate them. Open-ended tasks require human intervention or very sophisticated NLP techniques to correct them.

Premise (NLI) One of the two texts provided to an NLI model, the other one being the **hypothesis**.

Response (task) A student's answer to a task.

Task Within an assignment, exercise, test, or exam on a digital learning platform, a task is a single question to which a student responds with a single answer.

References

[1] Rodrigo Agerri. “Metaphor in Textual Entailment”. In: *COLING 2008, 22nd International Conference on Computational Linguistics, Posters Proceedings, 18-22 August 2008, Manchester, UK*. Ed. by Donia Scott and Hans Uszkoreit. 2008, pp. 3–6. URL: https://aclanthology.org/C08-2001/.

[2] Rodrigo Agerri et al. “Textual Entailment as an Evaluation Framework for Metaphor Resolution: A Proposal”. In: *Semantics in Text Processing. STEP 2008 Conference Proceedings, Venice, Italy, September 22-24, 2008*. Ed. by Johan Bos and Rodolfo Delmonte. Association for Computational Linguistics, 2008. URL: https://aclanthology.org/W08-2228/.

[3] Dimitrios Alikaniotis, Helen Yannakoudakis, and Marek Rei. “Automatic Text Scoring Using Neural Networks”. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016, pp. 715–725. DOI: 10.18653/v1/P16-1068. URL: https://aclanthology.org/P16-1068.

[4] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: http://arxiv.org/abs/1409.0473.

[5] Collin F. Baker, Charles J. Fillmore, and John B. Lowe. “The Berkeley FrameNet Project”. In: *36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics, COLING-ACL ’98, August 10-14, 1998, Université de Montréal, Montréal, Quebec, Canada. Proceedings of the Conference*. Ed. by Christian Boitet and Pete Whitelock. Morgan Kaufmann Publishers / ACL, 1998, pp. 86–90. DOI: 10.3115/980845.980860. URL: https://aclanthology.org/P98-1013/.

[6] Robert L. Bangert-Drowns et al. “The Instructional Effect of Feedback in Test-Like Events”. In: *Review of Educational Research* 61.2 (June 1991), pp. 213–238. ISSN: 0034-6543, 1935-1046. DOI: 10.3102/00346543061002213. URL: http://journals.sagepub.com/doi/10.3102/00346543061002213 (visited on 08/22/2022).

[7] Jerrold E. Barnett and Alisha L. Francis. "Using higher order thinking questions to foster critical thinking: a classroom study". In: *Educational Psychology* 32.2 (2012), pp. 201–211. DOI: 10.1080/01443410.2011.638619. URL: https://doi.org/10.1080/01443410.2011.638619.

[8] Sumit Basu, Chuck Jacobs, and Lucy Vanderwende. "Powergrading: a Clustering Approach to Amplify Human Effort for Short Answer Grading". In: *Transactions of the Association for Computational Linguistics* 1 (Oct. 2013), pp. 391–402. ISSN: 2307-387X. DOI: 10.1162/tacl_a_00236. URL: https://doi.org/10.1162/tacl%5C_a%5C_00236.

[9] Yonatan Belinkov et al. "Don't Take the Premise for Granted: Mitigating Artifacts in Natural Language Inference". In: *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Ed. by Anna Korhonen, David R. Traum, and Lluís Màrquez. Association for Computational Linguistics, 2019, pp. 877–891. DOI: 10.18653/v1/p19-1084. URL: https://doi.org/10.18653/v1/p19-1084.

[10] Jan Philip Bernius, Stephan Krusche, and Bernd Bruegge. "A Machine Learning Approach for Suggesting Feedback in Textual Exercises in Large Courses". In: *Proceedings of the Eighth ACM Conference on Learning @ Scale*. L@S '21. event-place: Virtual Event, Germany. New York, NY, USA: Association for Computing Machinery, 2021, pp. 173–182. ISBN: 978-1-4503-8215-1. DOI: 10.1145/3430895.3460135. URL: https://doi.org/10.1145/3430895.3460135.

[11] BigScience. *bigscience/bloom - Hugging Face*. July 29, 2022. URL: https://huggingface.co/bigscience/bloom/tree/main (visited on 07/31/2022).

[12] Paul Black and Dylan Wiliam. "Assessment and classroom learning". In: *Assessment in Education: principles, policy & practice* 5.1 (1998). Publisher: Taylor & Francis, pp. 7–74.

[13] Benjamin S. Bloom. "The 2 Sigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring". In: *Educational Researcher* 13.6 (1984), pp. 4–16. ISSN: 0013189X, 1935102X. URL: http://www.jstor.org/stable/1175554 (visited on 08/09/2022).

[14] Johan Bos and Katja Markert. "Recognising Textual Entailment with Logical Inference". In: *HLT/EMNLP 2005, Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference, 6-8 October 2005, Vancouver, British Columbia, Canada*. The Association for Computational Linguistics, 2005, pp. 628–635. URL: https://aclanthology.org/H05-1079/.

[15] Johan Bos and Katja Markert. “When logical inference helps determining textual entailment (and when it doesn’t)”. In: *Proceedings of the second PASCAL RTE challenge*. 2006, p. 26.

[16] Samuel R. Bowman et al. “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, 2015.

[17] Tom B. Brown et al. “Language Models are Few-Shot Learners”. In: *arXiv:2005.14165 [cs]* (July 22, 2020). arXiv: `2005.14165`. URL: `http://arxiv.org/abs/2005.14165` (visited on 03/09/2022).

[18] Jill Burstein, Claudia Leacock, and Richard Swartz. *Automated evaluation of essays and short answers*. Publisher: Loughborough University. 2001.

[19] David H Callear, Jenny Jerrams-Smith, and Victor Soh. “CAA of short non-MCQ answers”. In: (2001). Publisher: Loughborough University.

[20] Oana-Maria Camburu et al. “e-SNLI: Natural Language Inference with Natural Language Explanations”. In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Ed. by Samy Bengio et al. 2018, pp. 9560–9572. URL: `https://proceedings.neurips.cc/paper/2018/hash/4c7a167bb329bd92580a99ce422d6fa6-Abstract.html`.

[21] Leon Camus and Anna Filighera. “Investigating Transformers for Automatic Short Answer Grading”. In: *Artificial Intelligence in Education*. Ed. by Ig Ibert Bittencourt et al. Cham: Springer International Publishing, 2020, pp. 43–48. ISBN: 978-3-030-52240-7.

[22] Joaquin Quiñonero Candela et al., eds. *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*. Vol. 3944. Lecture Notes in Computer Science. Southampton, UK: Springer, 2006. ISBN: 3-540-33427-0. DOI: `10.1007/11736790`. URL: `https://doi.org/10.1007/11736790`.

[23] Vicente Iván Sánchez Carmona, Jeff Mitchell, and Sebastian Riedel. “Behavior Analysis of NLI Models: Uncovering the Influence of Three Factors on Robustness”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans,*

Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers). Ed. by Marilyn A. Walker, Heng Ji, and Amanda Stent. Association for Computational Linguistics, 2018, pp. 1975–1985. DOI: `10.18653/v1/n18-1179`. URL: `https://doi.org/10.18653/v1/n18-1179`.

[24] Noam Chomsky. *Syntactic Structures*. OCLC: 1146261827. Netherlands: De Gruyter Mouton, 1957. ISBN: 978-3-11-230484-6.

[25] Noam Chomsky. "Systems of Syntactic Analysis". In: *J. Symb. Log.* 18.3 (1953), pp. 242–256. DOI: `10.2307/2267409`. URL: `https://doi.org/10.2307/2267409`.

[26] European Commission et al. *Teachers in Europe : careers, development and well-being*. Ed. by P Birch. Publications Office of the European Union, 2021. DOI: `doi/10.2797/997402`.

[27] Alexis Conneau and Guillaume Lample. "Cross-lingual Language Model Pretraining". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Ed. by Hanna M. Wallach et al. 2019, pp. 7057–7067. URL: `https://proceedings.neurips.cc/paper/2019/hash/c04c19c2c2474dbf5f7ac4372c5b9af1-Abstract.html`.

[28] Alexis Conneau et al. "Supervised Learning of Universal Sentence Representations from Natural Language Inference Data". In: *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Ed. by Martha Palmer, Rebecca Hwa, and Sebastian Riedel. Association for Computational Linguistics, 2017, pp. 670–680. DOI: `10.18653/v1/d17-1070`. URL: `https://doi.org/10.18653/v1/d17-1070`.

[29] Alexis Conneau et al. "Unsupervised Cross-lingual Representation Learning at Scale". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by Dan Jurafsky et al. Association for Computational Linguistics, 2020, pp. 8440–8451. DOI: `10.18653/v1/2020.acl-main.747`. URL: `https://doi.org/10.18653/v1/2020.acl-main.747`.

[30] Alexis Conneau et al. "XNLI: Evaluating Cross-lingual Sentence Representations". In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*. Ed. by Ellen Riloff et al. Association for Computational Linguistics, 2018, pp. 2475–2485. DOI: `10.18653/v1/d18-1269`. URL: `https://doi.org/10.18653/v1/d18-1269`.

[31] Daniel Corral, Shana K. Carpenter, and Sam Clingan-Siverly. “The effects of immediate versus delayed feedback on complex concept learning”. In: *Quarterly Journal of Experimental Psychology* 74.4 (2021), pp. 786–799. DOI: 10.1177/1747021820977739. URL: https://doi.org/10.1177/1747021820977739.

[32] Ido Dagan, Oren Glickman, and Bernardo Magnini. “The PASCAL Recognising Textual Entailment Challenge”. In: *Machine Learning Challenges, Evaluating Predictive Uncertainty, Visual Object Classification and Recognizing Textual Entailment, First PASCAL Machine Learning Challenges Workshop, MLCW 2005, Southampton, UK, April 11-13, 2005, Revised Selected Papers*. Ed. by Joaquin Quiñonero Candela et al. Vol. 3944. Lecture Notes in Computer Science. Springer, 2005, pp. 177–190. DOI: 10.1007/11736790_9. URL: https://doi.org/10.1007/11736790%5C_9.

[33] Ido Dagan et al. *Recognizing Textual Entailment: Models and Applications*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2013. ISBN: 978-1-59829-834-5. DOI: 10.2200/S00509ED1V01Y201305HLT023. URL: https://doi.org/10.2200/S00509ED1V01Y201305HLT023.

[34] Andrew M. Dai and Quoc V. Le. “Semi-supervised Sequence Learning”. In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*. Ed. by Corinna Cortes et al. 2015, pp. 3079–3087. URL: https://proceedings.neurips.cc/paper/2015/hash/7137debd45ae4d0ab9aa953017286b20-Abstract.html.

[35] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Thamar Solorio. Association for Computational Linguistics, 2019, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: https://doi.org/10.18653/v1/n19-1423.

[36] Roberta E. Dihoff et al. “Provision of Feedback During Preparation For Academic Testing: Learning Is Enhanced by Immediate But Not Delayed Feedback”. In: *The Psychological Record* 54.2 (Apr. 1, 2004), pp. 207–231. ISSN: 2163-3452. DOI: 10.1007/BF03395471. URL: https://doi.org/10.1007/BF03395471.

[37] Nan Du et al. "GLaM: Efficient Scaling of Language Models with Mixture-of-Experts". In: *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*. Ed. by Kamalika Chaudhuri et al. Vol. 162. Proceedings of Machine Learning Research. PMLR, 2022, pp. 5547–5569. URL: https://proceedings.mlr.press/v162/du22c.html.

[38] Lisa Ehrlinger and Wolfram Wöß. "Towards a Definition of Knowledge Graphs". In: *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Conference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016*. Ed. by Michael Martin, Martí Cuquet, and Erwin Folmer. Vol. 1695. CEUR Workshop Proceedings. CEUR-WS.org, 2016. URL: http://ceur-ws.org/Vol-1695/paper4.pdf.

[39] Elizabeth Badger and Brenda Thomas. "Open-ended questions in reading". In: *Practical Assessment, Research, and Evaluation* 3 (1991), pp. 1992–1993.

[40] Dumitru Erhan et al. "Why Does Unsupervised Pre-training Help Deep Learning?" In: *J. Mach. Learn. Res.* 11 (2010), pp. 625–660. DOI: 10.5555/1756006.1756025. URL: https://dl.acm.org/doi/10.5555/1756006.1756025.

[41] Peter Foltz, Darrell Laham, and T. Landauer. "The intelligent essay assessor: Applications to educational technology". In: *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* (Apr. 1999).

[42] Emily R. Fyfe et al. "ManyClasses 1: Assessing the Generalizable Effect of Immediate Feedback Versus Delayed Feedback Across Many College Classes". In: *Advances in Methods and Practices in Psychological Science* 4.3 (July 2021), p. 251524592110275. ISSN: 2515-2459, 2515-2467. DOI: 10.1177/25152459211027575. URL: http://journals.sagepub.com/doi/10.1177/25152459211027575 (visited on 08/22/2022).

[43] Tianyu Gao, Adam Fisch, and Danqi Chen. "Making Pre-trained Language Models Better Few-shot Learners". In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*. Ed. by Chengqing Zong et al. Association for Computa-

tional Linguistics, 2021, pp. 3816–3830. DOI: 10.18653/v1/2021.acl-long.295. URL: https://doi.org/10.18653/v1/2021.acl-long.295.

[44] Sarah Gibson et al. *Workload Challenge: Analysis of teacher consultation responses*. Department for Education (U.K.), Feb. 2015. URL: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/401406/RR445_-_Workload_Challenge_-_Analysis_of_teacher_consultation_responses_FINAL.pdf (visited on 08/10/2022).

[45] Oren Glickman and Ido Dagan. "Web Based Probabilistic Textual Entailment". In: *Proceedings of the First Challenge Workshop Recognising Textual Entailment* (Jan. 2005).

[46] Tuanji Gong and Xuaxia Yao. "An attention-based deep model for automatic short answer score". In: *International Journal of Computer Science and Software Engineering* 8.6 (2019). Publisher: Dorma Trading, Est. Publishing Manager, pp. 127–132.

[47] *Guardians of the Galaxy*. 2015.

[48] Suchin Gururangan et al. *Annotation Artifacts in Natural Language Inference Data*. Number: arXiv:1803.02324. Apr. 16, 2018. arXiv: 1803.02324[cs]. URL: http://arxiv.org/abs/1803.02324 (visited on 07/29/2022).

[49] Sanda Harabagiu, Andrew Hickl, and Finley Lacatusu. "Satisfying information needs with multi-document summaries". In: *Information Processing & Management* 43.6 (2007). Text Summarization, pp. 1619–1642. ISSN: 0306-4573. DOI: https://doi.org/10.1016/j.ipm.2007.01.004. URL: https://www.sciencedirect.com/science/article/pii/S030645730700026X.

[50] Pengcheng He et al. "Deberta: decoding-Enhanced Bert with Disentangled Attention". In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. URL: https://openreview.net/forum?id=XPZIaotutsD.

[51] Yulan He, Siu Cheung Hui, and Tho Thanh Quan. "Automatic summary assessment for intelligent tutoring systems". In: *Computers & Education* 53.3 (2009), pp. 890–899. ISSN: 0360-1315. DOI: https://doi.org/10.1016/j.compedu.2009.05.008. URL: https://www.sciencedirect.com/science/article/pii/S0360131509001195.

[52] Jesús Herrera de la Cruz et al. *UNED at PASCAL RTE-2 challenge*. Universidad Nacional de Educaci on a Distancia, 2006.

[53] Md Mosharaf Hossain, Dhivya Chinnappa, and Eduardo Blanco. "An Analysis of Negation in Natural Language Understanding Corpora". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Ed. by Smaranda Muresan, Preslav Nakov, and Aline Villavicencio. Association for Computational Linguistics, 2022, pp. 716–723. URL: `https://aclanthology.org/2022.acl-short.81`.

[54] Md Mosharaf Hossain et al. "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 2020, Online, November 16-20, 2020*. Ed. by Bonnie Webber et al. Association for Computational Linguistics, 2020, pp. 9106–9118. DOI: `10.18653/v1/2020.emnlp-main.732`. URL: `https://doi.org/10.18653/v1/2020.emnlp-main.732`.

[55] Md Mosharaf Hossain et al. "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 2020, Online, November 16-20, 2020*. Ed. by Bonnie Webber et al. Association for Computational Linguistics, 2020, pp. 9106–9118. DOI: `10.18653/v1/2020.emnlp-main.732`. URL: `https://doi.org/10.18653/v1/2020.emnlp-main.732`.

[56] Shengluan Hou, Shuhan Zhang, and Chaoqun Fei. "Rhetorical structure theory: A comprehensive review of theory, parsing methods and applications". In: *Expert Systems with Applications* 157 (2020), p. 113421. ISSN: 0957-4174. DOI: `https://doi.org/10.1016/j.eswa.2020.113421`. URL: `https://www.sciencedirect.com/science/article/pii/S0957417420302451`.

[57] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Ed. by Iryna Gurevych and Yusuke Miyao. Association for Computational Linguistics, 2018, pp. 328–339. DOI: `10.18653/v1/P18-1031`. URL: `https://aclanthology.org/P18-1031/`.

[58] Jeremy Howard and Sebastian Ruder. "Universal Language Model Fine-tuning for Text Classification". In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Ed. by

Iryna Gurevych and Yusuke Miyao. Association for Computational Linguistics, 2018, pp. 328–339. DOI: 10.18653/v1/P18-1031. URL: https://aclanthology.org/P18-1031/.

[59] Junjie Hu et al. "XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalisation". In: *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event.* Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 4411–4421. URL: http://proceedings.mlr.press/v119/hu20b.html.

[60] John Hutchins. "From First Conception to First Demonstration: the Nascent Years of Machine Translation, 1947–1954. A Chronology". In: *Machine Translation* 12 (2004), pp. 195–252.

[61] John Hutchins. *The first public demonstration of machine translation: the Georgetown-IBM system, 7th January 1954.* 2006.

[62] Diana Inkpen, Darren Kipp, and Vivi Nastase. "Machine learning experiments for textual entailment". In: *Proceedings of the Second Challenge Workshop Recognising Textual Entailment.* 2006, pp. 17–20.

[63] Juliano Rabelo et al. *COLIEE 2020: Methods for Legal Document Retrieval and Entailment.* 2020. URL: https://sites.ualberta.ca/~rabelo/COLIEE2021/COLIEE_2020_summary.pdf.

[64] Aikaterini-Lida Kalouli et al. "Explaining Simple Natural Language Inference". In: *Proceedings of the 13th Linguistic Annotation Workshop.* Proceedings of the 13th Linguistic Annotation Workshop. Florence, Italy: Association for Computational Linguistics, 2019, pp. 132–143. DOI: 10.18653/v1/W19-4016. URL: https://www.aclweb.org/anthology/W19-4016 (visited on 07/28/2022).

[65] Tushar Khot, Ashish Sabharwal, and Peter Clark. "SciTaiL: A Textual Entailment Dataset from Science Question Answering". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 27, 2018). ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v32i1.12022. URL: https://ojs.aaai.org/index.php/AAAI/article/view/12022 (visited on 08/16/2022).

[66] Yuta Koreeda and Christopher Manning. "ContractNLI: A Dataset for Document-level Natural Language Inference for Contracts". In: *Findings of the Association for Computational Linguistics: EMNLP 2021.* Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 1907–1919. URL: https://aclanthology.org/2021.findings-emnlp.164.

[67] James A. Kulik and Chen-Lin C. Kulik. "Timing of Feedback and Verbal Learning". In: *Review of Educational Research* 58.1 (Mar. 1988), pp. 79–97. ISSN: 0034-6543, 1935-1046. DOI: 10.3102/00346543058001079. URL: http://journals.sagepub.com/doi/10.3102/00346543058001079 (visited on 08/22/2022).

[68] Sachin Kumar, Soumen Chakrabarti, and Shourya Roy. "Earth Mover's Distance Pooling over Siamese LSTMs for Automatic Short Answer Grading." In: *IJCAI*. 2017, pp. 2046–2052.

[69] T. K. Landauer and S. T. Dumais. "A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge." In: *Psychological Review* 104.2 (1997), pp. 211–240.

[70] Leah S. Larkey. "Automatic Essay Grading Using Text Categorization Techniques". In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '98. event-place: Melbourne, Australia. New York, NY, USA: Association for Computing Machinery, 1998, pp. 90–95. ISBN: 1-58113-015-5. DOI: 10.1145/290941.290965. URL: https://doi.org/10.1145/290941.290965.

[71] Claudia Leacock and Martin Chodorow. "C-rater: Automated Scoring of Short-Answer Questions". In: *Computers and the Humanities* 37.4 (Nov. 1, 2003), pp. 389–405. ISSN: 1572-8412. DOI: 10.1023/A:1025779619903. URL: https://doi.org/10.1023/A:1025779619903.

[72] Duane Lemley et al. "The effects of immediate and delayed feedback on secondary distance learners". In: *Quarterly Review of Distance Education* 8.3 (2007). Publisher: IAP-Information Age Publishing, Inc. PO Box 79049, Charlotte, NC 28271-7047, pp. 251–260.

[73] Mike Lewis et al. "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by Dan Jurafsky et al. Association for Computational Linguistics, 2020, pp. 7871–7880. DOI: 10.18653/v1/2020.acl-main.703. URL: https://doi.org/10.18653/v1/2020.acl-main.703.

[74] Pengfei Liu et al. "Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing". In: *arXiv:2107.13586 [cs]* (July 28, 2021). arXiv: 2107.13586. URL: http://arxiv.org/abs/2107.13586 (visited on 03/01/2022).

[75] Tiaoqiao Liu et al. “Automatic Short Answer Grading via Multiway Attention Networks”. In: *Artificial Intelligence in Education*. Ed. by Seiji Isotani et al. Cham: Springer International Publishing, 2019, pp. 169–173. ISBN: 978-3-030-23207-8.

[76] Yang Liu et al. “Learning Natural Language Inference using Bidirectional LSTM model and Inner-Attention”. In: *CoRR* abs/1605.09090 (2016). arXiv: 1605.09090. URL: http://arxiv.org/abs/1605.09090.

[77] Yinhan Liu et al. “RoBERTa: A Robustly Optimized BERT Pretraining Approach”. In: *arXiv:1907.11692 [cs]* (July 26, 2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (visited on 03/01/2022).

[78] Bill MacCartney. “Natural Language Inference”. PhD thesis. Stanford University, June 2009. URL: https://nlp.stanford.edu/~manning/dissertations/MacCartney-Bill-nli-diss.pdf (visited on 03/20/2022).

[79] Bill MacCartney and Christopher D. Manning. “An extended model of natural logic”. In: *Proceedings of the Eight International Conference on Computational Semantics*. Tilburg, The Netherlands: Association for Computational Linguistics, Jan. 2009, pp. 140–156. URL: https://aclanthology.org/W09-3714.

[80] Bill MacCartney and Christopher D. Manning. “Natural Logic for Textual Inference”. In: *Proceedings of the ACL-PASCAL@ACL 2007 Workshop on Textual Entailment and Paraphrasing, Prague, Czech Republic, June 28-29, 2007*. Ed. by Satoshi Sekine et al. Association for Computational Linguistics, 2007, pp. 193–200. URL: https://aclanthology.org/W07-1431/.

[81] William C Mann and Sandra A Thompson. “Rhetorical structure theory: Toward a functional theory of text organization”. In: *Text-interdisciplinary Journal for the Study of Discourse* 8.3 (1988). Publisher: De Gruyter Mouton, pp. 243–281.

[82] M. Marelli et al. *A SICK cure for the evaluation of compositional distributional semantic models*. 2014.

[83] Oliver Mason and Ian Grove-Stephensen. *Automated free text marking with paperless school*. Publisher: Loughborough University. 2002.

[84] Wendy McColskey and Mark R. Leary. “Differential effects of norm-referenced and self-referenced feedback on performance expectancies, attributions, and motivation”. In: *Contemporary Educational Psychology* 10.3 (1985), pp. 275–284. ISSN: 0361-476X. DOI: https://doi.org/10.1016/0361-476X(85)90024-4. URL: https://www.sciencedirect.com/science/article/pii/0361476X85900244.

[85] R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. *Right for the Wrong Reasons: Diagnosing Syntactic Heuristics in Natural Language Inference*. Number: arXiv:1902.01007. June 24, 2019. arXiv: 1902.01007[cs]. URL: http://arxiv.org/abs/1902.01007 (visited on 07/29/2022).

[86] Cheryl A. Melovitz Vasan et al. "Analysis of testing with multiple choice versus open-ended questions: Outcome-based observations in an anatomy course". In: *Anatomical Sciences Education* 11.3 (2018), pp. 254–261. DOI: https://doi.org/10.1002/ase.1739. URL: https://anatomypubs.onlinelibrary.wiley.com/doi/abs/10.1002/ase.1739.

[87] Joshua J. Michalenko, Andrew S. Lan, and Richard G. Baraniuk. "Data-Mining Textual Responses to Uncover Misconception Patterns". In: *Proceedings of the Fourth (2017) ACM Conference on Learning @ Scale*. L@S '17. event-place: Cambridge, Massachusetts, USA. New York, NY, USA: Association for Computing Machinery, 2017, pp. 245–248. ISBN: 978-1-4503-4450-0. DOI: 10.1145/3051457.3053996. URL: https://doi.org/10.1145/3051457.3053996.

[88] George A. Miller. "WordNet: A Lexical Database for English". In: *Commun. ACM* 38.11 (1995), pp. 39–41. DOI: 10.1145/219717.219748. URL: http://doi.acm.org/10.1145/219717.219748.

[89] Omid Mirmotahari et al. "A case-study of automated feedback assessment". In: *2019 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 2019, pp. 1190–1197.

[90] Tom Mitchell et al. "Towards robust computerised marking of free-text responses". In: *Proceedings of the 6th International Computer Assisted Assessment (CAA) Conference*. 2002.

[91] Michael Mohler and Rada Mihalcea. "Text-to-Text Semantic Similarity for Automatic Short Answer Grading". In: *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*. Athens, Greece: Association for Computational Linguistics, Mar. 2009, pp. 567–575. URL: https://aclanthology.org/E09-1065.

[92] Filip Moons, Ellen Vandervieren, and Jozef Colpaert. "Atomic, reusable feedback: a semi-automated solution for assessing handwritten tasks? A crossover experiment with mathematics teachers." In: *Computers and Education Open* 3 (Dec. 2022), p. 100086. ISSN: 26665573. DOI: 10.1016/j.caeo.2022.100086. URL: https://linkinghub.elsevier.com/retrieve/pii/S2666557322000143 (visited on 08/09/2022).

[93] Mary-Anne Neal. "Engaging Students through Effective Questions." In: *Education Canada* 51.1 (2011). Publisher: ERIC, n1.

[94] Yixin Nie et al. “Adversarial NLI: A New Benchmark for Natural Language Understanding”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Ed. by Dan Jurafsky et al. Association for Computational Linguistics, 2020, pp. 4885–4901. DOI: `10.18653/v1/2020.acl-main.441`. URL: `https://doi.org/10.18653/v1/2020.acl-main.441`.

[95] RODNEY D. NIELSEN, WAYNE WARD, and JAMES H. MARTIN. “Recognizing entailment in intelligent tutoring systems”. In: *Natural Language Engineering* 15.4 (2009). Publisher: Cambridge University Press, pp. 479–501. DOI: `10.1017/S135132490999012X`.

[96] F. Noorbehbahani and A. A. Kardan. “The automatic assessment of free text answers using a modified BLEU algorithm”. In: *Computers & Education* 56.2 (2011), pp. 337–345. ISSN: 0360-1315. DOI: `https://doi.org/10.1016/j.compedu.2010.07.013`. URL: `https://www.sciencedirect.com/science/article/pii/S0360131510002058`.

[97] Yasuhiro Ozuru et al. “Comparing comprehension measured by multiple-choice and open-ended questions.” In: *Canadian Journal of Experimental Psychology/Revue canadienne de psychologie expérimentale* 67.3 (2013). Publisher: Educational Publishing Foundation, p. 215.

[98] Ellis B. Page. “The Imminence of... Grading Essays by Computer”. In: *The Phi Delta Kappan* 47.5 (1966). Publisher: Phi Delta Kappa International, pp. 238–243. ISSN: 00317217. URL: `http://www.jstor.org/stable/20371545` (visited on 08/13/2022).

[99] Ellis B. Page. “The Use of the Computer in Analyzing Student Essays”. In: *International Review of Education / Internationale Zeitschrift für Erziehungswissenschaft / Revue Internationale de l’Education* 14.2 (1968). Publisher: Springer, pp. 210–225. ISSN: 00208566, 15730638. URL: `http://www.jstor.org/stable/3442515` (visited on 08/13/2022).

[100] Kishore Papineni et al. “Bleu: a method for automatic evaluation of machine translation”. In: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 2002, pp. 311–318.

[101] Matthew E. Peters et al. “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Ed. by Marilyn A. Walker, Heng Ji, and Amanda Stent. Association for Computational Linguistics, 2018,

pp. 2227–2237. DOI: 10.18653/v1/n18-1202. URL: https://doi.org/10.18653/v1/n18-1202.

[102] Adam Poliak. "A Survey on Recognizing Textual Entailment as an NLP Evaluation". In: *CoRR* abs/2010.03061 (2020). arXiv: 2010.03061. URL: https://arxiv.org/abs/2010.03061.

[103] Arya Prabhudesai and Ta N. B. Duong. "Automatic Short Answer Grading using Siamese Bidirectional LSTM Based Regression". In: *2019 IEEE International Conference on Engineering, Technology and Education (TALE)*. 2019, pp. 1–6. DOI: 10.1109/TALE48000.2019.9226026.

[104] Alec Radford and Karthik Narasimhan. "Improving Language Understanding by Generative Pre-Training". In: 2018.

[105] Alec Radford et al. "Language Models are Unsupervised Multitask Learners". In: 2019.

[106] Jack W. Rae et al. "Scaling Language Models: Methods, Analysis & Insights from Training Gopher". In: *CoRR* abs/2112.11446 (2021). arXiv: 2112.11446. URL: https://arxiv.org/abs/2112.11446.

[107] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer". In: *J. Mach. Learn. Res.* 21 (2020), 140:1–140:67. URL: http://jmlr.org/papers/v21/20-074.html.

[108] Rian Adam Rajagede and Rochana Prih Hastuti. "Stacking Neural Network Models for Automatic Short Answer Scoring". In: *IOP Conference Series: Materials Science and Engineering* 1077.1 (Feb. 2021). Publisher: IOP Publishing, p. 012013. DOI: 10.1088/1757-899x/1077/1/012013. URL: https://doi.org/10.1088/1757-899x/1077/1/012013.

[109] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: *arXiv:1908.10084 [cs]* (Aug. 27, 2019). arXiv: 1908.10084. URL: http://arxiv.org/abs/1908.10084 (visited on 04/14/2022).

[110] Brian Riordan et al. "Investigating neural architectures for short answer scoring". In: *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Copenhagen, Denmark: Association for Computational Linguistics, Sept. 2017, pp. 159–168. DOI: 10.18653/v1/W17-5017. URL: https://aclanthology.org/W17-5017.

[111] Adam Roberts, Colin Raffel, and Noam Shazeer. “How Much Knowledge Can You Pack Into the Parameters of a Language Model?” In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Ed. by Bonnie Webber et al. Association for Computational Linguistics, 2020, pp. 5418–5426. DOI: 10.18653/v1/2020.emnlp-main.437. URL: https://doi.org/10.18653/v1/2020.emnlp-main.437.

[112] Tim Rocktäschel et al. “Reasoning about Entailment with Neural Attention”. In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2016. URL: http://arxiv.org/abs/1509.06664.

[113] Fátima Rodrigues and Paulo Oliveira. “A system for formative assessment and monitoring of students’ progress”. In: *Computers & Education* 76 (2014), pp. 30–41. ISSN: 0360-1315. DOI: https://doi.org/10.1016/j.compedu.2014.03.001. URL: https://www.sciencedirect.com/science/article/pii/S0360131514000517.

[114] Lorenza Romano et al. “Investigating a Generic Paraphrase-Based Approach for Relation Extraction”. In: *11th Conference of the European Chapter of the Association for Computational Linguistics*. Trento, Italy: Association for Computational Linguistics, Apr. 2006, pp. 409–416. URL: https://aclanthology.org/E06-1052.

[115] Alexey Romanov and Chaitanya Shivade. “Lessons from Natural Language Inference in the Clinical Domain”. In: *arXiv:1808.06752 [cs]* (Aug. 21, 2018). arXiv: 1808.06752. URL: http://arxiv.org/abs/1808.06752 (visited on 08/27/2018).

[116] Guilherme Moraes Rosa et al. “Billions of Parameters Are Worth More Than In-domain Training Data: A case study in the Legal Case Entailment Task”. In: *CoRR* abs/2205.15172 (2022). DOI: 10.48550/arXiv.2205.15172. arXiv: 2205.15172. URL: https://doi.org/10.48550/arXiv.2205.15172.

[117] Carolyn P. Rose et al. “A Hybrid Text Classification Approach for Analysis of Student Essays”. In: *In Building Educational Applications Using Natural Language Processing*. 2003, pp. 68–75.

[118] Lawrence M Rudner and Tahung Liang. “Automated essay scoring using Bayes’ theorem”. In: *The Journal of Technology, Learning and Assessment* 1.2 (2002).

[119] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter”. In: *CoRR* abs/1910.01108 (2019). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108.

[120] Timo Schick and Hinrich Schütze. “Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference”. In: *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*. Ed. by Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty. Association for Computational Linguistics, 2021, pp. 255–269. DOI: 10.18653/v1/2021.eacl-main.20. URL: https://doi.org/10.18653/v1/2021.eacl-main.20.

[121] Soumya Sharma et al. “Incorporating Domain Knowledge into Medical NLI using Knowledge Graphs”. 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 2019, Hong Kong, China, November 3-7, 2019*. Ed. by Kentaro Inui et al. Association for Computational Linguistics, 2019, pp. 6091–6096. DOI: 10.18653/v1/D19-1631. URL: https://doi.org/10.18653/v1/D19-1631.

[122] Noam Shazeer et al. “Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer”. In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=B1ckMDqlg.

[123] Chaitanya Shivade. *MedNLI — A Natural Language Inference Dataset For The Clinical Domain*. Type: dataset. 2017. DOI: 10.13026/C2RS98. URL: https://physionet.org/content/mednli/ (visited on 08/15/2022).

[124] V.J. Shute and S. Rahimi. “Review of computer-based assessment for learning in elementary and secondary education: Computer-based assessment for learning”. In: *Journal of Computer Assisted Learning* 33.1 (Feb. 2017), pp. 1–19. ISSN: 02664909. DOI: 10.1111/jcal.12172. URL: https://onlinelibrary.wiley.com/doi/10.1111/jcal.12172 (visited on 08/10/2022).

[125] Valerie Shute. “Focus on Formative Feedback”. In: *Review of Educational Research* 78 (Mar. 2008), pp. 153–189. DOI: 10.3102/0034654307313795.

[126] Vivian Dos Santos Silva, André Freitas, and Siegfried Handschuh. “Building a Knowledge Graph from Natural Language Definitions for Interpretable Text Entailment Recognition”. In: *Proceedings of the Eleventh*

International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018. Ed. by Nicoletta Calzolari et al. European Language Resources Association (ELRA), 2018. URL: `http://www.lrec-conf.org/proceedings/lrec2018/summaries/190.html`.

[127] Vivian Dos Santos Silva, André Freitas, and Siegfried Handschuh. "Exploring Knowledge Graphs in an Interpretable Composite Approach for Text Entailment". In: *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*. AAAI Press, 2019, pp. 7023–7030. DOI: `10.1609/aaai.v33i01.33017023`. URL: `https://doi.org/10.1609/aaai.v33i01.33017023`.

[128] Chris Singleton. "Computer-based assessment in education." In: *Educational and Child Psychology* 18.3 (2001). Place: United Kingdom Publisher: British Psychological Society, pp. 58–74. ISSN: 2396-8702(Electronic),0267-1611(Print).

[129] Shaden Smith et al. "Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, A Large-Scale Generative Language Model". In: *CoRR* abs/2201.11990 (2022). arXiv: `2201.11990`. URL: `https://arxiv.org/abs/2201.11990`.

[130] Kevin Stowe, Prasetya Utama, and Iryna Gurevych. "IMPLI: Investigating NLI Models' Performance on Figurative Language". In: *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 5375–5388. DOI: `10.18653/v1/2022.acl-long.369`. URL: `https://aclanthology.org/2022.acl-long.369` (visited on 08/02/2022).

[131] Jana Z. Sukkarieh, Stephen G. Pulman, and Nicholas Raikes. "Automarking: using computational linguistics to score short, free text responses. Paper presented at the 29th annual conference". In: *of the International Association for Educational Assessment (IAEA*. 2003.

[132] Jana Z. Sukkarieh and Svetlana Stoyanchev. "Automating Model Building in C-Rater". In: *Proceedings of the 2009 Workshop on Applied Textual Inference*. TextInfer '09. Suntec, Singapore: Association for Computational Linguistics, 2009, pp. 61–69. ISBN: 9781932432480.

[133] Chul Sung, Tejas Indulal Dhamecha, and Nirmal Mukhi. "Improving Short Answer Grading Using Transformer-Based Pre-training". In: *Artificial Intelligence in Education*. Ed. by Seiji Isotani et al. Cham: Springer International Publishing, 2019, pp. 469–481. ISBN: 978-3-030-23204-7.

[134] Maite Taboada and William C Mann. "Applications of rhetorical structure theory". In: *Discourse studies* 8.4 (2006). Publisher: Sage Publications Sage CA: Thousand Oaks, CA, pp. 567–588.

[135] Pete Thomas et al. "E-Assessment Using Latent Semantic Analysis in the Computer Science Domain: A Pilot Study". In: *Proceedings of the Workshop on ELearning for Computational Linguistics and Computational Linguistics for ELearning*. eLearn '04. event-place: Geneva. USA: Association for Computational Linguistics, 2004, pp. 38–44.

[136] James Thorne et al. "FEVER: a Large-scale Dataset for Fact Extraction and VERification". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*. Ed. by Marilyn A. Walker, Heng Ji, and Amanda Stent. Association for Computational Linguistics, 2018, pp. 809–819. DOI: 10.18653/v1/n18-1074. URL: https://doi.org/10.18653/v1/n18-1074.

[137] Seilin Uhm et al. "Impact of tailored feedback in assessment of communication skills for medical students". In: *Medical Education Online* 20.1 (Jan. 2015), p. 28453. ISSN: 1087-2981. DOI: 10.3402/meo.v20.28453. URL: https://www.tandfonline.com/doi/full/10.3402/meo.v20.28453 (visited on 07/29/2022).

[138] Ghent University. *SUBTLEXus - Department of Experimental Psychology - Ghent University*. URL: https://www.ugent.be/pp/experimentele-psychologie/en/research/documents/subtlexus/overview.htm (visited on 06/05/2022).

[139] Lucy Vanderwende, Arul Menezes, and Rion Snow. "Microsoft Research at RTE-2: Syntactic Contributions in the Entailment Task: an implementation". In: *Proceedings of the Second PASCAL Recognising Textual Entailment Challenge Workshop*. Edition: Proceedings of the Second PASCAL Recognising Textual Entailment Challenge Workshop. Jan. 2006. URL: https://www.microsoft.com/en-us/research/publication/microsoft-research-at-rte-2-syntactic-contributions-in-the-entailment-task-an-implementation/.

[140] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 5998–6008. URL: `https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html`.

[141] Sinong Wang et al. “Entailment as Few-Shot Learner”. In: *arXiv:2104.14690 [cs]* (Apr. 29, 2021). arXiv: `2104.14690`. URL: `http://arxiv.org/abs/2104.14690` (visited on 03/01/2022).

[142] Zichao Wang et al. “A Meta-Learning Augmented Bidirectional Transformer Model for Automatic Short Answer Grading.” In: *EDM*. 2019.

[143] Zikang Wang, Linjing Li, and Daniel Zeng. “Knowledge-Enhanced Natural Language Inference Based on Knowledge Graphs”. In: *Proceedings of the 28th International Conference on Computational Linguistics*. Barcelona, Spain (Online): International Committee on Computational Linguistics, Dec. 2020, pp. 6498–6508. DOI: `10.18653/v1/2020.coling-main.571`. URL: `https://aclanthology.org/2020.coling-main.571`.

[144] Barbara A Wasik and Annemarie H Hindman. “Realizing the promise of open-ended questions”. In: *The Reading Teacher* 67.4 (2013), pp. 302–311.

[145] Aaron Steven White et al. “Inference is Everything: Recasting Semantic Resources into a Unified Evaluation Framework”. In: *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Taipei, Taiwan: Asian Federation of Natural Language Processing, Nov. 2017, pp. 996–1005. URL: `https://aclanthology.org/I17-1100`.

[146] Dave Whittington and Helen Hunt. “Approaches to the computerized assessment of free text responses”. In: *Proceedings of the Sixth International Computer Assisted Assessment Conference* (1999). Publisher: Loughborough University.

[147] Adina Williams, Nikita Nangia, and Samuel Bowman. “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)*. event-place: New Orleans, Louisiana. Association for Computational Linguistics, 2018, pp. 1112–1122. URL: `http://aclweb.org/anthology/N18-1101`.

[148] *World Athletics*. URL: `https://www.worldathletics.org/records/by-discipline/sprints/100-metres/outdoor/men` (visited on 08/02/2022).

[149] Linting Xue et al. *mT5: A massively multilingual pre-trained text-to-text transformer*. Number: arXiv:2010.11934. Mar. 11, 2021. arXiv: `2010.11934[cs]`. URL: `http://arxiv.org/abs/2010.11934` (visited on 05/17/2022).

[150] Peter Young et al. "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions". In: *Trans. Assoc. Comput. Linguistics* 2 (2014), pp. 67–78. DOI: `10.1162/tacl_a_00166`. URL: `https://doi.org/10.1162/tacl%5C_a%5C_00166`.