Supporting Context Monotonicity Abstractions in Neural NLI Models

Julia Rozanova [†], Deborah Ferreira[†], Mokanarangan Thayaparan[†], Marco Valentino[†], André Freitas[†‡]
Department of Computer Science, University of Manchester, United Kingdom[†]
Idiap Research Institute, Switzerland[‡]
{firstname.surname} @manchester.ac.uk

Abstract

Natural language **contexts** display logical regularities with respect to substitutions of related concepts: these are captured in a functional order-theoretic property called *monotonicity*. For a certain class of NLI problems where the resulting entailment label depends only on the context monotonicity and the relation between the substituted concepts, we build on previous techniques that aim to improve the performance of NLI models for these problems, as consistent performance across both upward and downward monotone contexts still seems difficult to attain even for state of the art models. To this end, we reframe the problem of **context monotonicity classification** to make it compatible with transformer-based pre-trained NLI models and add this task to the training pipeline. Furthermore, we introduce a sound and complete simplified monotonicity logic formalism which describes our treatment of contexts as abstract units. Using the notions in our formalism, we adapt targeted challenge sets to investigate whether an intermediate context monotonicity classification task can aid NLI models' performance on examples exhibiting monotonicity reasoning.

1 Introduction

NLI has seen much success in terms of performance on large benchmark datasets, but there are still expected systematic reasoning patterns that we fail to observe in the state of the art NLI models. We focus in particular on *monotonicity reasoning*: a large class of NLI problems that can be described as a form of *substitutional* reasoning which displays logical regularities with respect to substitution of related concepts. In this setting, a subphrase $\mathbf{a}$ of a premise $p(\mathbf{a})$ is replaced with a phrase $\mathbf{b}$, yielding a hypothesis $p(\mathbf{b})$.

Usually, the resulting entailment label relies on exactly two properties: the inclusion relation between concepts $\mathbf{a}$ and $\mathbf{b}$, and the *systematic behaviour* of the context p with respect to such relations.

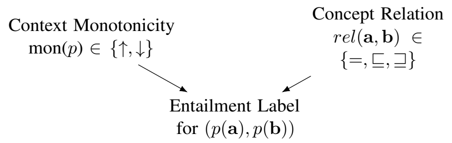

Figure 1: The class of entailment problems under consideration: premise-hypothesis pairs $(p(\mathbf{a}), p(\mathbf{b}))$ whose entailment label depends only on the monotonicity of the context p and the relation between $\mathbf{a}$ and $\mathbf{b}$.

In formal semantics, this is referred to as the *monotonicity* of p (where p is either upward or downward monotone), and this reasoning pattern is referred to as *monotonicity reasoning*. Monotonicity reasoning is incredibly systematic, and thus is a much-probed behaviour in enquiries into the *systematicity* and *generalization capability* of neural NLI models (Goodwin et al., 2020; Yanaka et al., 2020, 2019; Richardson et al., 2020; Geiger et al., 2020).

Determining both the concept relation and the context monotonicity requires significant linguistic understanding of syntactic structure and scope of operators, but in terms of reasoning, this is a very systematic pattern that nevertheless has a history of causing problems for neural models. It has been observed (Yanaka et al., 2019; Geiger et al., 2020) that current state of the art transformer-based NLI models tend to routinely fail in downward monotone contexts, such as those arising in the presence of negation or generalized quantifiers. Recent strategies (Richardson et al., 2020) to address the shortcomings of NLI models in downward-monotone contexts have followed the *inoculation* method (Liu et al., 2019a): additional training data which exhibits the target phenomenon (in this case, downward-monotone reasoning) is used to fine-

Proceedings of the 1st and 2nd Workshops on Natural Logic Meets Machine Learning (NALOMA), pages 41–50
June 16, 2021.

tune existing models. This is done with some success in (Yanaka et al., 2019; Richardson et al., 2020) and (Geiger et al., 2020). In contrast, we wish to investigate a *transfer learning* strategy that directly targets the monotonicity question as an *additional training task* to see if this can *further* improve the monotonicity reasoning performance of popular transformer-based NLI models.

Our contributions are as follows:

- Emphasizing monotonicity as a property of a *context*, we introduce a sound and complete logical formalism which models the monotonicity reasoning phenomenon but abstracts away from specific linguistic operators, treating the context as a single abstract object.

- Extending our treatment of contexts as individual objects to an experimental setting, we introduce an improvement in neural NLI model performance on monotonicity reasoning challenge datasets by employing a context monotonicity classification task in the training pipieline of NLI models. To the best of our knowledge, this is the first use of neural models for this specific task.

- For this purpose, we adapt the **HELP** dataset (Yanaka et al., 2019) into a **HELP-Contexts** dataset, mimicking the structure of our logical formalism.

- For the class of NLI problems described as *monotonicity reasoning*, we demonstrate the impact of the proposed transfer strategy: we show that there can be a strong improvement on downward monotone contexts, previously known to be a bottleneck for neural NLI models. As such, this shows the benefit of directly targeting intermediate abstractions (in this case, monotonicity) present in logical formalisms that underly the true label.

2 Contexts and Monotonicity

2.1 Contexts

Informally, we treat a natural language *context* as a sentence with a "gap", represented by a variable symbol.

A context $p(x)$:

I ate some x for breakfast.

A sentence $S = p(\text{‘fruit’})$:

I ate some fruit for breakfast.

Although every sentence can be viewed as a context with an insertion in as many ways as there are n-grams in the sentence, in this work we shall consider in particular those contexts where the variable corresponds to a slot in the expression that may be filled by an *entity* reference, such as a noun or noun phrase. In the view of Montague-style formal semantics, these contexts correspond to expressions of type $< e, t >$.

2.2 Monotonicity

Given a natural language context p and a pair of nouns/noun phrases $(\mathbf{a}, \mathbf{b})$, we may create a natural language sentence pair $(p(\mathbf{a}), p(\mathbf{b}))$ by substituting the respective subphrases into the natural language context. When treated as a premise-hypothesis pair (as in the experimental NLI task setting), the gold entailment label has a strong relationship with the kinds of *relations* that exist between the insertions $\mathbf{a}$ and $\mathbf{b}$.

In the seminal works on monotonicity (Valencia, 1991; van Benthem, 1988), the relations that are studied are *semantic containment* relations, which are defined analogously to set-theoretic containment relations ($\subseteq$).

	a	b
$\equiv$	couch	sofa
$\sqsubset$	apples South African soccer players dogs with hats	fruit soccer players dogs

Table 1: Examples of the semantic containment relation between concept pairs.

For insertions related by $\sqsubset$, the gold entailment label depends on one other property: the combined *monotonicity profile* of all the linguistic operators within whose scope the insertion is located. If the final monotonicity marking in the insertion's position is "upward", the gold label is entailment. However, if it is "downward", we can deduce entailment of the reversed sentence pair, $(p(\mathbf{b}), p(\mathbf{a}))$. Linguistic operators such as "not" are downward monotone, while generalized quantifiers such as "every" have a more complex monotonicity profile: downward-monotone in the first argument and upward-monotone in the second argument. The monotonicity properties of all the operators compose along the syntax tree, culminating in a final monotonicity marking for the "x" position in the context (the monotonicity is independent of the inserted word). It is this final monotonicity la-

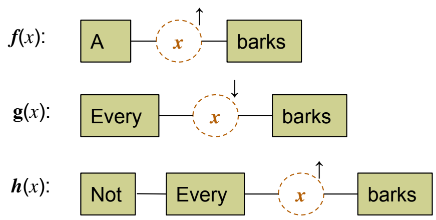

Figure 2: Natural language *contexts* have a property which dictates logical regularities with respect to concept hierarchies: like numerical functions, they can be upward monotone or downward monotone.

bel that determines the entailment patterns with respect to insertion relations. Although there are formalisms that model this logical behaviour (Icard et al., 2017), they aim to model the behaviour of each linguistic linguistic operator and the way they compose given the parse tree of a sentence.

We consider a simplification of this behaviour by abstracting away the linguistic specifics of the context, treating it as a single abstract object. As such, we are not concerned with the exact monotonicity profiles of all the linguistic operators that culminate in the monotonicity of the final context. We describe this behaviour with a simple logic system extending the $\mathcal{L}(all)$ logic of (Moss, 2008a) with the abstracted behaviour of upward and donward monotone contexts. We include a proof that this adaptation is still sound and complete.

2.3 A Context-Abstracted Monotonicity Logic

We extend the syllogistic syntax of the language $\mathcal{L}(all)$ included in (Moss, 2008b) and (Moss, 2008a). In keeping with that style, we present the syntax as natural language sentences. However, we include the corresponding first order formulae as well. In the subsequent proofs, we mix the stylizations somewhat for readability, but the table below should serve as a reminder for the exact correspondence.

Definition 2.1. Let the language $\mathcal{L}$ consist of the following:

1. A countable set $\mathbf{A}$ of constant symbols $\mathbf{a}, \mathbf{b}, \mathbf{a}_1, \mathbf{b}_1, \ldots$
2. Exactly two variables, x and y
3. A binary relation symbol $\sqsubseteq$.
4. A set $\mathbf{P}$ of relation symbols $\sqsubseteq_p$ indexed by a countable set $p, p_1, \ldots$.

Only the following are considered sentences in the language $\mathcal{L}$:

Natural Language Stylization	FOL Stylization
all $\mathbf{a}$ are $\mathbf{b}$	$\mathbf{a} \sqsubseteq \mathbf{b}$
if $p(\mathbf{a})$ then $p(\mathbf{b})^*$	$\mathbf{a} \sqsubseteq_p \mathbf{b}^*$
p is upward monotone	$\forall_{x,y}(x \sqsubseteq y \leftrightarrow x \sqsubseteq_p y)$
p is downward monotone	$\forall_{x,y}(x \sqsubseteq y \leftrightarrow y \sqsubseteq_p x)$

Table 2: * For every natural language context p in a set P of contexts, and where $p(\mathbf{a})$ is the substitution of $\mathbf{a}$ into the natural language context p.

This can in many ways be seen as a simplification of previous formalisms (Icard et al., 2017; Hu and Moss, 2018) based on either extending the syllogistic logic $\mathcal{L}(all)$ (Moss, 2008a) or extending typed lambda calculus with monotonicity behaviour. The key difference of this approach is the abstraction away from specific linguistic operators and their monotonicity profiles. On one hand, we are thus only modeling one level of linguistic compositionality, but since the monotonicity profile of every linguistic operator composes into one monotonicity marker which affects the final entailment label (for this class of problem), it encompasses all of the linguistically-specific approaches. This is useful when the monotonicity of a context can be determined by an external system such as a neural classifier or the ccg-2-mono system (Hu and Moss, 2018) . In this case, the monotonicity marking of the entire context is explicit.

2.4 Semantics

Definition 2.2. A model $\mathcal{M}$ of the language $\mathcal{L}$ is a structure

$$\mathcal{M} = (M, [\![\cdot]\!])$$

consisting of a set M and an interpretation function $[\![\cdot]\!]$ where $[\![\mathbf{a}]\!] \subseteq M$, $[\![\sqsubseteq]\!]$ is the $\subseteq$ relation on the powerset $\mathcal{P}(M)$ and $[\![\sqsubseteq_p]\!] \subseteq \mathcal{P}(M) \times \mathcal{P}(M)$ is any binary relation on $\mathcal{P}(M)$. Truth of a formula with respect to a given model is defined as follows:

2.5 Proof Calculus

Our language will be equipped with the following deduction rules and axioms:

$$\begin{array}{lcc}
\mathcal{M} \models \mathbf{a} \sqsubseteq \mathbf{b} & \iff & [\![\mathbf{a}]\!] \subseteq [\![\mathbf{b}]\!] \\
\mathcal{M} \models \mathbf{a} \sqsubseteq_p \mathbf{b} & \iff & [\![\mathbf{a}]\!], [\![\mathbf{b}]\!] \in [\![\sqsubseteq_p]\!] \\
\mathcal{M} \models \forall_{x,y}(x \sqsubseteq y \leftrightarrow x \sqsubseteq_p y) & \iff & \subseteq = [\![\sqsubseteq_p]\!] \\
\mathcal{M} \models \forall_{x,y}(x \sqsubseteq y \leftrightarrow y \sqsubseteq_p x) & \iff & \supseteq = [\![\sqsubseteq_p]\!]
\end{array}$$

$$\frac{\text{ALL } \mathbf{a} \text{ ARE } \mathbf{b} \quad \text{ALL } \mathbf{b} \text{ ARE } \mathbf{c}}{\text{ALL } \mathbf{a} \text{ ARE } \mathbf{c}}\ \textit{BARBARA}$$

$$\frac{\text{ALL } \mathbf{a} \text{ ARE } \mathbf{b} \quad p \text{ IS UPWARD MONOTONE}}{\text{IF } p(\mathbf{a}) \text{ THEN } p(\mathbf{b})} \uparrow$$

$$\frac{\text{ALL } \mathbf{a} \text{ ARE } \mathbf{b} \quad p \text{ IS DOWNWARD MONOTONE}}{\text{IF } p(\mathbf{b}) \text{ THEN } p(\mathbf{a})} \downarrow$$

$$\frac{}{\text{ALL } \mathbf{a} \text{ ARE } \mathbf{a}}\ Axiom1$$

$$\frac{}{\text{IF } p(\mathbf{a}) \text{ THEN } p(\mathbf{a})}\ Axiom2$$

2.6 Soundness and Completeness

Definition 2.3. For a set of $\mathcal{L}$-sentences Γ, we can define the *canonical model* $\mathcal{M}_\Gamma$ as follows:

First, let M be the set of atomic constant symbols $\mathbf{A}$ and define a relation $\leq$ on $\mathbf{A}$ where $\mathbf{a} \leq \mathbf{b} \iff \Gamma \vdash a \sqsubset b$. The interpretation function $[\![\cdot]\!]$ is defined as follows:

Define $[\![\mathbf{a}]\!] = \downarrow \mathbf{a} = \{\mathbf{b} \in P \mid \mathbf{b} \leq \mathbf{a}\}$.

Define $[\![\sqsubseteq]\!]$ as the $\subseteq$ relation on $\mathcal{P}(M)$.

For each $p \in \mathbf{P}$, we have a conditional definition:

If and only if "p is upward monotone" is the only sentence about p entailed by Γ, we define $[\![\sqsubseteq_p]\!] = \subseteq$.

If and only if "p is downward monotone" is the only sentence about p entailed by Γ, we define $[\![\sqsubseteq_p]\!] = \subseteq$.

In all other cases, $[\![\sqsubseteq_p]\!]$ is defined as set equality in $\mathcal{P}(M)$.

Lemma 1. *For a set Γ of $\mathcal{L}$-sentences, the canonical model $\mathcal{M}_\Gamma \models \Gamma$.*

Proof. The key parts of the proof are a consequence of the fact that $\downarrow a \subseteq \downarrow b \iff a \leq b$, and $\downarrow a \supseteq \downarrow b \iff b \leq a$ which is crucial to the case that Γ contains both "p is upward monotone" and "p is downward monotone". The rest is a routine consequence of the definitions. □

Theorem 2. *Soundness and Completeness*

Proof. We leave the perfunctory soundness proof as an exercise to the reader. Towards showing completeness, let Γ be a set of $\mathcal{L}$-sentences and ϕ another $\mathcal{L}$-sentence. Suppose that for every model $\mathcal{M}$ we have that $\Gamma \models \phi$. In particular, by lemma 1, $\mathcal{M}_\Gamma \models \phi$. All further discussion occurs in this specific model. The sentence ϕ may have one of four forms.

Suppose firstly that ϕ is "if $p(\mathbf{a})$ then $p(\mathbf{b})$". Thus, $([\![\mathbf{a}]\!], [\![\mathbf{b}]\!]) \in [\![\sqsubseteq_p]\!]$. Since the interpretation of $\sqsubseteq_p$ depends on the description of p entailed by Γ, there are three cases: Firstly, if $\Gamma \vdash$ "p is upward monotone" only, then it follows that $[\![\mathbf{a}]\!] \subseteq [\![\mathbf{b}]\!]$. Since this holds if and only if $a \leq b$ by a basic property of down-sets, then we will have $\Gamma \vdash a \sqsubseteq b$ and $\Gamma \vdash$ "p is upward monotone", so that $\Gamma \vdash$ "if $p(\mathbf{a})$ then $p(\mathbf{b})$" by the $\uparrow$ deduction rule.

On the other hand, if we had that $\Gamma \vdash$ "p is downward monotone" only, we can similarly deduce that $[\![\mathbf{a}]\!] \supseteq [\![\mathbf{b}]\!]$, and repeating the same reasoning arrive at $\Gamma \vdash$ "if $p(\mathbf{a})$ then $p(\mathbf{b})$". In the last option for p, we either have that Γ proves neither or both of the statements "p is upward monotone" and "p is downward monotone", and in either case $[\![\sqsubseteq_p]\!]$ is set equality in $\mathcal{M}_\Gamma$. As such, we will be able to conclude that $[\![\mathbf{a}]\!] = [\![\mathbf{b}]\!]$. Equal down-sets imply that $a = b$, so that trivially $\Gamma \vdash$ "if $p(\mathbf{a})$ then $p(\mathbf{b})$" Hence, in all of these cases, $\Gamma \vdash \phi$.

If ϕ is the sentence "p is upward monotone" (we omit the dual, which is similar), then truth in the canonical model gives us that $\subseteq = [\![\sqsubseteq_p]\!]$. In the $\mathcal{M}_\Gamma$, this happens exactly when $\Gamma \vdash$ "p is upward monotone" . The last option for ϕ is covered in the completeness theorem for the basic syllogistic logic with the "BARBARA" deduction rule and Axiom 1.

In conclusion, in all cases we may deduce that $\Gamma \vdash \phi$. □

3 Related Work

The study of monotonicity in natural language has a strongly developed linguistic and mathematical theoretical groundwork, dating back to the monotonicity calculus of (Valencia, 1991) and in semantic studies such as (van Benthem, 1988). Its formal treatments have led to the expansion of typed lambda calculus with an order relation so as to model this order-theoretic behaviour, resulting in a corresponding deduction system and completeness theorem in (Icard et al., 2017). There are varying presentations and some variation in terminology, but for the most part *monotonicity* refers to the order-theoretic property of the context as a function, while the term *polarity* usually refers to the tag assigned to the node in a CCG parse tree or a word in a sentence. The inferential mechanism based on monotonicity properties of quantifiers, determiners and contexts in general is sometimes referred to as *natural logic*, and the underlying principles of natural logic applying to set-theoretic concept relations has led to work on *generalized monotonicity* (MacCartney and Manning, 2009). However, the additional relations such as negation, alternation and cover are no longer order-theoretic notions.

Symbolic Implementations There are two flavours of implementations that result in the deductions allowed by monotonicity reasoning. Firstly, works such as (Hu et al.; Abzianidze, 2015) rely on linguistically-informed polarity markings on the nodes of CCG parse trees. They require accurate parses and expertly hand-crafted linguistic rules to mark the nodes with polarity tags, as in (Hu and Moss, 2018). In (Hu et al.), a premise is tagged for monotonicity and a knowledge base of hypotheses created by a substitution known to be truth-preserving is generated. Candidate hypotheses are compared with this set, checking for exact matches. On the other hand, (Abzianidze, 2015) uses the CCG parses to further translate to a lambda logical form for use in a deduction method inspired by tableau calculus. These approaches differ from strategies such as (MacCartney and Manning, 2009), which require an *edit sequence* which transforms the premise into the hypothesis. Atomic edits are tagged with generalized entailment relations which are combined with a join operator based on relational composition to determine whether the transformation is overall truth-preserving, hence yielding a hypothesis entailed by the premise. Later, (Angeli and Manning, 2014) treated these atomic edits as edges in a graph and phrased entailment detection as a graph search problem. Concepts from symbolic approaches to NLI have also been applied in symbolic question answering systems (such as in (Bobrow et al., 2007)), and hybridized with neural systems (such as in (Kalouli et al., 2020)).

Neural NLI Models and Monotonicity State of the art NLI models have previously been shown (Yanaka et al., 2019; Geiger et al., 2020) to perform poorly on examples where the context f is *downward monotone*, as occurs in the presence of negation and various generalized quantifiers such as "every" and "neither". Benchmark datasets such as MNLI are somewhat starved of such examples, as observed by (Yanaka et al., 2019). As a consequence, the models trained on such benchmark datasets as MNLI not only fail in downward monotone contexts, but *systematically* fail: they tend to treat all examples as if the contexts are upward monotone, predicting the *opposite* entailment label with high accuracy (Yanaka et al., 2019; Geiger et al., 2020). Data augmentation techniques and additional fine-tuning with an inoculation (Liu et al., 2019a) strategy have been attempted in (Yanaka et al., 2019; Richardson et al., 2020) and (Geiger et al., 2020). In the latter case, performance on a challenge test set improved without much performance loss on the original benchmark evaluation set (SNLI), but in (Yanaka et al., 2019) there was a significant decrease in performance on the MNLI evaluation set. These studies form the basis on which we aim to build, and their choice of evaluation datasets and models inspires our own choices.

		Previous Work			
Evaluation Datasets		Geiger 2020 (Neural)	Yanaka 2020 (Neural)	Moss 2019 (Neural)	Hu 2020 (Symbolic)
Large, Broad Coverage	MNLI Test		x		
	MNLI Dev (Mismatched)			x	
	SNLI Test	x		x	
Small, Targeted Phenomena	MED		x		
	SICK		x*		x
	FraCaS		x*		x
	MoNLI Test	x			
	Monotonicity Fragments			x	x

Table 3: Evaluation datasets used in previous work investigating monotonicity reasoning. Positions marked $*$ indicate that the dataset is included in another used evaluation dataset.

Neural Transformer-based language models have been shown to implicitly model syntactic structure (Hewitt and Manning, 2019). There is also evidence to suggest that these NLI models are at least representing the concept relations quite well and using this information to predict the entailment label, as corroborated by a study based on *interchange interventions* in (Geiger et al., 2020).

We hypothesise that such models have the capacity for learning monotonicity features. The extent to which the representations capture monotonicity information in the contextual representations of tokens in the sequence is not yet well understood, and this is an investigation we wish to initiate and encourage with this work.

4 Experiments

Building on the observations in the above-mentioned previous papers, we ask the following questions:

- Can a context monotonicity classification task in the model training pipeline further improve performance on targeted evaluation sets which test monotonicity reasoning?
- Does this mitigate the decrease in performance on benchmark NLI datasets?

Our investigation proceeds in three parts: Firstly, we attempt to fine-tune a SOTA NLI model for a context monotonicity classification task.

Secondly, we retrain the above model for NLI and evaluate the performance on several evaluation datasets which specifically target examples of both upward and downward monotonicity reasoning. We examine whether there is any improvement over a previously suggested approach on fine-tuning on a large, automatically generated dataset (HELP) from (Yanaka et al., 2019).

Models We start with existing NLI models pretrained on benchmark NLI datasets. In particular (and for best comparison with related studies) we use RoBERTa (Liu et al., 2019b) pretrained on MNLI (Williams et al., 2018) and BERT (Devlin et al., 2019) pretrained on SNLI (Bowman et al., 2015). These are two benchmark NLI datasets which contain examples derived from naturally occurring text and crowd-sourced labels, aiming for scale and broad coverage. We do not deviate from the architecture, as we are only investigating the effect of training on different tasks (monotonicity classification and NLI) and datasets.

4.1 Retraining NLI Models to Classify Context Monotonicity

Traditionally, symbolic approaches treat monotonicity classification as the task of labeling of each node in a CCG parse tree with either an upward or downward polarity marking. Our emphasis of monotonicity as a property of a *context* allows for a different framing of this problem: we consider monotonicity classification as a binary classification task by explicitly indicating (with a variable) the "slot" in the sentence for which we wish to know the polarity. Different positions of the variable in a partial sentence may yield a context with a different monotonicity label; a typical example of this is sentences featuring generalized quantifiers such as "every", which may be monotone up in one argument but monotone down in another.

4.1.1 Input Representation

The NLI models which we wish to start with are transformer-based models, in line with the current state of the art approaches to NLI. Transformer models represent a sentence as a sequence of tokens: we take a naive approach to representing a context by indicating the variable with an uninformative 'x' token. We refrain from using the mask token to indicate the variable, as the underlying pretrained transformer language models are trained to embed the mask token in such a way as to correspond with high-likelihood insertions in that position, which we would prefer to avoid.

4.1.2 Dataset

In order to ensure our monotonicity classification task does not add any unseen data (when compared to only fine-tuning on the HELP dataset) we adapt the HELP dataset for this task. The HELP dataset (Yanaka et al., 2019) consists of a set of $(p(\mathbf{a}), p(\mathbf{b}))$ pairs which included labels for the entailment relationship between the full sentences, the inclusion relation between $\mathbf{a}$ and $\mathbf{b}$, and the monotonicity classification of p. As such, we extract only the contexts p and the monotonicity label into dataset which we will call "HELP-contexts", which we split into a train, development and test set in a 50:20:30 ratio. Examples of this dataset are presented on Table 4. [1]

Context	Context Monotonicity
There were no x today.	downward monotone
There is no time for x.	downward monotone
Every x laughed.	downward monotone
There is little if any hope for his x .	downward monotone
Some x are allergic to wheat.	upward monotone
Tom is buying some flowers for x.	upward monotone
You can see some wild rabbits in the x.	upward monotone

Table 4: Examples from the HELP-contexts dataset, with respective labels.

[1] The original HELP dataset also contains a few non-monotone examples: in the current state of this work, these are omitted in favor of a focus on the specific confusion in existing models where downwards monotone contexts are often treated as upwards monotone ones.

4.1.3 Results

As presented in Table 5, the task of predicting the monotonicity of a context can be solved using fine-tuned transformer models. This suggests a potential path for inducing a bias for context classification in downstream tasks such as NLI, which could benefit from better modeling of context monotonicity.

Model	Evaluation Data					
	HELP-Contexts Dev			HELP-Contexts Test		
	Precision	Recall	F1-Score	Precision	Recall	F1-Score
bert-base	98.74	99.08	98.91	98.00	95.24	96.54
bert-large	98.23	98.88	98.55	97.51	95.70	96.57
roberta-large-mnli	99.62	98.92	99.26	98.73	96.64	97.64
roberta-large	99.81	99.46	99.27	98.99	96.41	97.62
roberta-base	99.81	99.46	99.63	98.10	95.56	96.76
bert-base-uncased-snli	98.88	98.19	8.53	98.92	97.29	98.07

Table 5: Performance of state of the art models for the context prediction task. Each model was trained on HELP context (training set).

4.2 Improving NLI Performance on Monotonicity Reasoning

A few datasets have been curated to either fine-tune or evaluate NLI models with monotonicity reasoning in mind: their uses in previous related works are detailed in table 3. We use the following datasets for training and evaluation respectively:

4.2.1 Training Data

We start by once again using the HELP dataset (Yanaka et al., 2019), which was designed specifically as a balanced additional training set for the improvement of NLI models with respect to monotonicity reasoning. We create a split of this dataset which is based on the HELP-contexts dataset by assigning each example either to the train, development or test set depending on which split its associated context f is in the HELP-contexts dataset. This is to ensure there is no overlap between the examples' contexts accross the three data partitions. Our approach combined this strategy with an additional step based on the context monotonicity task described in section 4.1.

4.2.2 Training Procedure

We rely on the architecture implementations and pretrained models available with the *transformers* library (Wolf et al., 2020). Starting with the pretrained models (which we shall henceforth tag as "bert-base-uncased-snli" and "roberta-large-mnli"), we first fine-tune these models for the context monotonicity classification task using the training partition of the HELP-contexts dataset. We re-use the classification head of the pretrained models for this purpose, but only use two output states for the classification.

4.2.3 Evaluation Data

Evaluation datasets are typically small, challenging and categorized by certain target semantic phenomena. Following previous work in this area, we evaluate our approach using the MED dataset introduced in (Yanaka et al., 2020), which is annotated with monotonicity information and draws from various expertly-curated diagnostic challenge sets in NLI such as SICK, FraCaS and the SuperGlue Diagnostic set. It features a balanced split between upward and downward monotone contexts, in contrast to the benchmark MNLI dataset. Additionally, we include evaluation on the MoNLI dataset (Geiger et al., 2020) which also features a labeled balance of upward and downward monotone examples. However, the latter dataset's downward monotone examples are only exemplary of contexts featuring the negation operator *"not"*, whereas MED (Yanaka et al., 2020) also includes more complex downward monotone operators such as generalized quantifiers and determiners. We refer to these respective papers (Yanaka et al., 2020; Geiger et al., 2020) for full breakdowns and analyses of these datasets.

4.2.4 Baselines

Although the main comparison to be made is the improvement introduced when including the context-monotonicity-classification training on top of the current state-of-the art roberta-large-mnli model trained on HELP, we include an additional baselines: roberta-large-mnli fine-tuned on the *monotonicity fragment* from the *semantic fragments* (Richardson et al., 2020) dataset. The strategy in this work is the same as with the HELP dataset, but we include this in the evaluation on the chosen challenge sets for a more complete comparison.

4.2.5 Results

We present the results on the challenge sets MED and MoNLI in Table 6, with a break-down by upward and downward monotone contexts. Furthermore, we have re-run each model on the original benchmark evaluation datasets SNLI and MNLI, with the results visible in Table 7.

Model	Additional Training Data	Challenge Datasets					
		MoNLI Test			MED		
		Upward Monotone	Downward Monotone	All	Upward Monotone	Downward Monotone	All
bert-base-uncased-snli	-	37.74	56.49	46.15	53.58	43.91	49.36
bert-base-uncased-snli	HELP	30.89	85.02	**55.19**	43.4	72.43	60.18
bert-base-uncased-snli	**HELP + HELP-Contexts**	21.6	97.67	**55.19**	32.56	87.13	**66.22**
roberta-large-mnli	-	95.19	5.32	58.84	82.12	25.76	46.09
roberta-large-mnli	Monotonicity Fragments (Easy)	92.68	79.62	86.81	74.54	65.68	70.05
roberta-large-mnli	Monotonicity Fragments (All)	50.00	50.00	50.00	35.42	61.80	49.78
roberta-large-mnli	HELP	94.72	98.67	96.48	64.47	86.25	**77.4**
roberta-large-mnli	**HELP + HELP-Contexts**	98.78	97.17	**98.06**	65.24	85.12	76.44

Table 6: Performance of NLI models on challenge datasets designed to test performance on monotonicity reasoning.

Model	Additional Training Data	Benchmark Datasets							
		MNLI (m*) Dev		MNLI (mm*) Dev		SNLI Dev		SNLI Test	
		Acc	Δ	Acc	Δ	Acc	Δ	Acc	Δ
bert-base-uncased-snli	-	44.96	-	45.52	-	41.54	-	40.78	-
bert-base-uncased-snli	HELP	35.13	-9.83	34.37	-11.5	25.93	-15.61	25.92	-14.86
bert-base-uncased-snli	**HELP + HELP-Contexts**	36.91	**-8.05**	37.36	**-8.16**	36.54	**-5.00**	37.20	**-3.58**
roberta-large-mnli	-	94.11	-	93.88	-	93.33	-	93.14	-
roberta-large-mnli	HELP	82.66	**-11.45**	83.38	**-10.50**	74.77	-18.56	74.39	-18.75
roberta-large-mnli	**HELP + HELP-Contexts**	81.00	-13.11	82.01	-11.87	82.99	**-10.34**	82.31	**-10.83**

Table 7: Fine-tuning state of the art NLI models with the aim of improving monotonicity has tended to result in lower performance on the original benchmark NLI datasets. We compare these performance losses in addition to tracking performance on the the challenge datasets. * MNLI (m) and (mm) refers to the matched and mismatched dataset respectively. For MNLI, only the *Dev* set is publically available.

5 Discussion

Average Performance Firstly, we confirm previous observations that the starting pretrained transformer model roberta-large-mnli (which is considered a high-performing NLI model, achieving over 93% accuracy on the large MNLI development set) has a dramatic performance imbalance with respect to context monotonicity. The fact that performance on downward monotone contexts is as low as 5% suggests that this model perhaps routinely assumes upward monotone contexts. It was noted in (Yanaka et al., 2019) that the MNLI benchmark dataset is strongly skewed in favor of upward monotone examples, which may account for this.

Our approach outperforms or matches the baseline models in three of the summary accuracy scores, and is competitive in the fourth. Furthermore, in most cases we observe less performance loss on the benchmark sets.

Performance by Monotonicity Category As evident from Table 6, we observe a substantial improvement for the bert-base-uncased NLI models for downward monotone contexts. For the much larger roberta-large-mnli models, any gains over the model trained on HELP only are quite small. A common observation is the notable trade-off between accuracy on upward and downward monotone contexts; training that improves one of these over a previous baseline generally seem to decrease performance of the other. This is especially evident in the MED dataset, which is larger and representative of a more diverse set of downward monotone examples (the MoNLI dataset is limited to the "No" operator). Sensibly, a decrease in performance in upward monotone contexts also leads to a decrease in performance on the original SNLI and MNLI datasets 7 (which are skewed in favor of upward monotone examples). However, in most cases (except for the roberta-large-mnli model on the MNLI benchmark) our method results in a *smaller* performance loss.

6 Conclusion and Future Work

Introducing context monotonicity classification into the training pipeline of NLI models provides some performance gains on challenge datasets designed to test monotonicity reasoning. We see contexts as crucial objects of study in future approaches to natural language inference. The ability to detect their logical properties (such as monotonicity) opens the door for hybrid neuro-symbolic NLI models and reasoning systems, especially in so far as dealing with out of domain insertions that may confuse out-of-the-box NLI models. The linguistic flexibility that transformer-based language models bring is too good to lose; leveraging their power in situations where only part of our sentence is in a model's distribution would be helpful for domain-specific use cases with many out-of-distribution nouns. Overall, we are interested in furthering both the *analysis* and *improvement* of emergent modelling of abstract logical features in neural natural language processing models.

References

Lasha Abzianidze. 2015. A tableau prover for natural logic and language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2492–2502, Lisbon, Portugal. Association for Computational Linguistics.

Gabor Angeli and Christopher D Manning. 2014. Naturalli: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 534–545.

Johan van Benthem. 1988. Essays in logical semantics. *Studia Logica*, 47(2):172–173.

D. Bobrow, Bob Cheslow, C. Condoravdi, Tracy Holloway King, R. Nairn, and A. Zaenen. 2007. Parc's bridge and question answering system.

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

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

A. Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. *arXiv: Computation and Language*.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4129–4138, Minneapolis, Minnesota. Association for Computational Linguistics.

Hai Hu, Qi Chen, Kyle Richardson, Atreyee Mukherjee, Lawrence S Moss, and Sandra Kuebler. Monalog: a lightweight system for natural language inference based on monotonicity. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2020*, page 319.

Hai Hu and Larry Moss. 2018. Polarity computations in flexible categorial grammar. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 124–129, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Icard, Lawrence Moss, and William Tune. 2017. A monotonicity calculus and its completeness. In *Proceedings of the 15th Meeting on the Mathematics of Language*, pages 75–87, London, UK. Association for Computational Linguistics.

Aikaterini-Lida Kalouli, Richard Crouch, and Valeria de Paiva. 2020. Hy-NLI: a hybrid system for natural language inference. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5235–5249, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Nelson F. Liu, Roy Schwartz, and Noah A. Smith. 2019a. Inoculation by fine-tuning: A method for analyzing challenge datasets. *CoRR*, abs/1904.02668.

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

Bill MacCartney and Christopher D. Manning. 2009. An extended model of natural logic. In *Proceedings of the Eight International Conference on Computational Semantics*, pages 140–156, Tilburg, The Netherlands. Association for Computational Linguistics.

Lawrence S. Moss. 2008a. *Completeness theorems for syllogistic fragments*, pages 143–174. De Gruyter Mouton.

Lawrence S. Moss. 2008b. Syllogistic Logics with Verbs. *Journal of Logic and Computation*, 20(4):947–967.

Kyle Richardson, H. Hu, L. Moss, and A. Sabharwal. 2020. Probing natural language inference models through semantic fragments. *ArXiv*, abs/1909.07521.

V.M.S. Valencia. 1991. *Studies on Natural Logic and Categorial Grammar*. Universiteit van Amsterdam.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6105–6117, Online. Association for Computational Linguistics.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, Kentaro Inui, Satoshi Sekine, Lasha Abzianidze, and Johan Bos. 2019. HELP: A dataset for identifying shortcomings of neural models in monotonicity reasoning. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 250–255, Minneapolis, Minnesota. Association for Computational Linguistics.