Can Transformers Reason in Fragments of Natural Language?

Viktor Schlegel [1,2]
viktor_schlegel@asus.com

Kamen V. Pavlov [2]
pavloffkamen@gmail.com

Ian Pratt-Hartmann [2,3]
ian.pratt@manchester.ac.uk

[1]ASUS Intelligent Cloud Services (AICS), Singapore
[2]Department of Computer Science, University of Manchester, United Kingdom
[3]Instytut Informatyki Uniwersytet Opolski, Poland

Abstract

State-of-the-art deep-learning-based approaches to Natural Language Processing (NLP) are credited with various capabilities that involve reasoning with natural language texts. In this paper we carry out a large-scale empirical study investigating the detection of formally valid inferences in controlled fragments of natural language for which the satisfiability problem becomes increasingly complex. We find that, while transformer-based language models perform surprisingly well in these scenarios, a deeper analysis reveals that they appear to overfit to superficial patterns in the data rather than acquiring the logical principles governing the reasoning in these fragments.

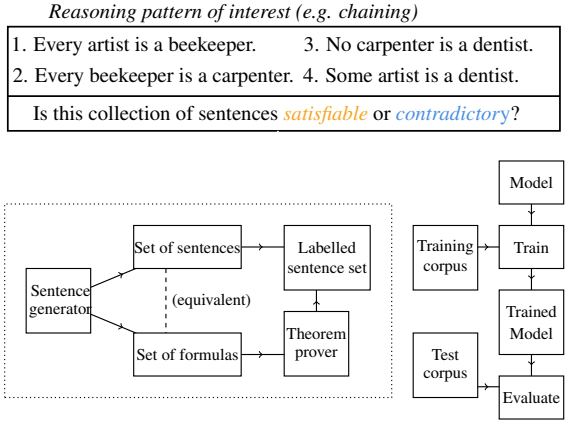

Figure 1: Depiction of our approach to evaluate reasoning capabilities in neural networks.

1 Introduction

The recent success of neural networks in a range of tasks connected with logical inference in natural language is remarkable. Foremost among such systems are those employing transformer-based language models (Vaswani et al., 2017) optimised on large corpora in an unsupervised manner (Devlin et al., 2019) and then further fine-tuned on task-specific datasets (Bowman et al., 2015; Rajpurkar et al., 2016). However, concerns persist regarding so-called "data-set artefacts" (Gururangan et al., 2018; Schlegel et al., 2022, 2020). Have netural networks really acquired the principles of reasoning with natural language, or are they merely responding to superficial patterns in the data?

In fact, two strands of research may be discerned in recent work on natural language inference (NLI). The first formulates the central task as follows: given a pair of sentences in some natural language, a *premise*, P and a *hypothesis*, H, determine whether P (i) *entails* or (ii) *contradicts* (i.e. entails the negation of) H; or (iii) P is *neutral* with respect to H. For example, the premise *A person rides his bicycle in the sand beside the ocean* is taken to entail the hypothesis *A person is on a beach*. The primary issue is the possibility of learning such a mapping on the basis of some data set consisting of many such sentence pairs, each tagged with the a 'correct' (i.e. gold standard) label as determined by human judgement, typically obtained by crowd-sourcing. Examples include the RTE dataset (Bentivogli et al., 2009) and the (much larger) SNLI and MNLI datasets (Bowman et al., 2015; Williams et al., 2018). Neural-network models achieve impressive accuracy on this task (Chen et al., 2017; Devlin et al., 2019). The nature of the challenge here is (deliberately) mixed: on the one hand, solving the NLI problem requires a grasp of the meaning of various closed-class words (*a*, *not*, ...) together with an appreciation of the semantics of the grammatical constructions involved. On the other hand, almost all the entailments or contradictions encountered rely on common sense knowledge (for example, that racing a bicycle involves riding it), which it is the job of the system to acquire. The supposition is that these pieces of commonsense knowledge mediate the entailment of the hypothesis (or its negation) from the premise. The logical basis of the inferences, so

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 11184 - 11199

reconstructed, is typically straightforward, amounting to simple syllogism-like inferences; but the mediating commonsense knowledge is coded in 'soft' form, matching the approximate, probabilistic nature of natural language inference. The operative notion of inference thus eludes a formal definition (Sugawara et al., 2018),ultimately resulting in debatable labels (Schlegel et al., 2020).

Thesecond strand of research investigates the ability of neural networks to recognize formally valid entailments *in sensu stricto*, but nevertheless couched in fragments of natural language. Examples include Salvatore et al. (2019); Richardson et al. (2019); Richardson and Sabharwal (2021) and Talmor et al. (2020). Here the focus is on learning the (potentially complex) inferential patterns inherent in the logical syntax of the closed-class words and grammatical constructions defining the fragment of language under investigation. For example, the premises *Every doctor who is not a philosopher is a baker*, *John is a doctor* and *John is not a baker* entail the hypothesis *John is a philosopher*. No commonsense knowledge is required here: the given premises either entail the given hypothesis formally, or they do not. Since entailment in a formally defined fragment of language is a matter of mathematical fact, not human judgement, inferential problems may be constructed artificially, using random sampling over a grammar, with the correct answers determined by an automated theorem prover (ATP). The question is not whether neural networks can be trained to mimic human annotators' judgements, but rather, whether they can learn the logical principles governing language in question. Note that, because it is formal validity rather than commonsense plausibility that is at issue here, inference tasks of interest typically involve not a *single* premise P, but rather, a *collection* of premises. Individual sentences seldom yield any non-trivial formal entailments.

These two strands of research reflect a difference in motivation. The former aims to understand inferences apparently performed by humans in everyday linguistic settings—inferences which are necessarily messy and approximate in character. From a formal point of view, many of the alleged entailments or contradictions are in fact judgements of *probability*. Thus, for example the premise *Alice is holding a dog* does not, logically speaking, contradict the hypothesis *Alice is holding a cat*, even assuming that no dog is a cat; yet such labellings abound in NLI datasets, reflecting the judgements made by (and instructions given to) the human annotators who generate them. By contrast, the second (strictly logical) strand of research is motivated principally by a desire to understand the rule-based character of natural language syntax and semantics, and in particular, by the question of whether a neural network can learn the rules in question.

In this paper, we report on a large-scale empirical study to investigate whether transformer-based language models can learn the logical syntax of natural language. Specifically, we consider performance on the problem of determining the satisfiability (logical consistency) of sets of English sentences featuring the determiners *every*, *some*, *no*, the negative adverb *not*, and relative clauses, in the context of a vocabulary of count nouns and transitive verbs. Importantly, the sentences in question are drawn from various different *fragments* of English, each characterized by a particular range of the grammatical constructions investigated. In each case, data sets—both for training and evaluation—are artificially generated in order to test the system's grasp of the underlying logic. Figure 1 illustrates the approach, giving an example problem instance for the very simplest of the fragments considered. We find that state-of-the-art deep learning-based approaches achieve impressive performance on this task. However, in-depth analysis of their generalisation capabilities reveals a more nuanced pattern, suggesting a tendency to overfit to the parameters that control the data generation, rather than learning the underlying logical principles.

2 Fragments of Language

By a *fragment* of a natural language, we understand a collection of sentences forming a naturally delineated subset of that language, and equipped with a truth-conditional semantics commanding the assent of its native speakers. For example, consider the fragment of natural language corresponding to the *classical syllogistic* given to us by Aristotle (1989, Book A). In its English-language version, it consists of the sentence forms

Every p is a q	No p is a q
Some p is a q	Some p is not a q,

with schematic variables p and q substituted by common (count) nouns. This fragment, which we here denote $\mathcal{S}$, can be used to formulate examples such as the one shown in Figure 1.

For present purposes, we may consider the truth

conditions of an English sentence to be given by translation to a formal language such as first-order logic, in a way which uncontroversially reconstructs the operative notion of logical entailment. Thus, for example, the sentence forms $\mathcal{S}$ correspond to the respective logical forms $\forall x(p(x) \to \pm q(x))$ and $\exists x(p(x) \wedge \pm q(x))$, where $\pm$ indicates either the presence or absence of negation ($\neg$). An argument in $\mathcal{S}$ is regarded as valid precisely when the first-order translations of its premises entail—in the familiar logical sense—the first-order translation of its conclusion.

Consider now the argument:

> *Some artist hates no beekeeper*; *every beekeeper hates some artist*; *therefore some artist is not a beekeeper*. (1)

This argument is again intuitively valid (though this takes a little thought to see). On the other hand, it cannot be sensibly cast in the syllogistic, because it so obviously hinges on intrinsically relational information. (Observe the alternation of subject and object of the verb *hates* in the two premises.) Is there, then, a larger fragment of English in which it might be expressed? Take the *relational syllogistic*, denoted $\mathcal{R}$, to be the fragment of English obtained by adding to $\mathcal{S}$ the sentence-forms:

Every p rs every/some q	Some p rs every/some q
No p rs every/any q	Some p does not r every/any q,

where p and q are common count nouns and r a transitive verb. As with $\mathcal{S}$, so too with $\mathcal{R}$: the truth-conditions of sentences can be captured by translation to first-order logic in a way which faithfully reconstructs the intuitive notion of validity. For example, "Some artist hates no beekeeper" may be rendered as $\exists x(\text{artist}(x) \wedge \forall y(\text{beekeeper}(y) \to \neg\text{hate}(x, y)))$, and so on. Under these semantics, argument (1) is confirmed as a valid argument in the fragment $\mathcal{R}$.

There are other ways to extend the fragment $\mathcal{S}$, of course. One (very modest) such extension is actually featured in the works of Aristotle (1963, Ch. 10). Let us say that the *extended classical syllogistic*, denoted $\mathcal{S}^+$, is the fragment of English which adds to $\mathcal{S}$ the sentence-forms

Every non-p is a q	Some non-p is a q
No non-p is a q	Some non-p is not a q,

corresponding to the first-order formulas $\forall x(\neg p(x) \to \pm q(x))$ and $\exists x(\neg p(x) \wedge \pm q(x))$. As we might say, $\mathcal{S}^+$ adds 'noun-level negation' to $\mathcal{S}$. Following in the same vein, we can extend $\mathcal{R}$ with noun-level negation, yielding sentences such as *"No non-carpenter admires any non-artist"* and so on. We call this fourth fragment the *extended relational syllogistic*, denoted $\mathcal{R}^+$. Again, translation into first-order logic is completely standard.

Let $\mathcal{L}$ be a fragment of some natural language. A set of $\mathcal{L}$-sentences is said to be *satisfiable* if there is a structure making the logical translations of these sentences true. The *satisfiability problem* for $\mathcal{L}$, denoted $\text{Sat}(\mathcal{L})$, is the problem of determining whether a given finite set of sentences of $\mathcal{L}$ is satisfiable. Provided $\mathcal{L}$ is equipped with a mechanism for sentence negation (as are all the fragments considered here), any procedure for solving $\text{Sat}(\mathcal{L})$ immediately yields a procedure for recognizing logical entailments in $\mathcal{L}$, since an argument is valid just in case its premises together with the negation of its conclusion is not satisfiable. Therefore, we have no use for the familiar classification of natural language inference problems as *entailment*, *contradiction* and *neutral* (and still less the four-way classification of Tian et al. (2021)): the satisfiability problem is as general as we need. As the fragment $\mathcal{L}$ becomes more expressive, the corresponding satisfiability problem $\text{Sat}(\mathcal{L})$ will, in general, become more increasingly difficult. However, as we shall see, the details of the resulting trade-off between expressiveness and ease of inference are rather intricate. The focus of the present paper is whether neural networks can learn to solve problems such as $\text{Sat}(\mathcal{L})$ for various fragments $\mathcal{L}$ such as $\mathcal{S}$, $\mathcal{S}^+$, $\mathcal{R}$ and $\mathcal{R}^+$.

We remark that the approach taken here is parallel to that taken with respect to the fragments GRL and RCL in (Richardson and Sabharwal, 2021). There are two notable differences, however, Firstly, the fragments considered here are, from a grammatical point of view, more basic, and less clearly a natural language version of propositional logic clauses. In particular, GRL includes the 'sentence' *If carrot and not steak then apples*; and even the more natural RCL is limited to the constructions *Every X who is (not) a Y is (not) a Z*. To allow comparison between this work and the present study we consider the minimal extension of $\mathcal{S}$ by means of *relative clauses*, thus allowing the additional sentence-forms

Every o who is a p is a q	No o who is a p is a q
Some o who is a p is a q	Some o who is a p is not a q

Denote this fragment by $\mathcal{S}_\text{r}$. Similarly, denote by $\mathcal{S}_\text{rn}$ the same fragment additionally allowing negative relative clauses, such as "Every o who is not

a p is a q" and so on. The fragment $\mathcal{S}_{\mathrm{rn}}$ is actually an extension of RCL.

An important motivation for the rather more general approach taken here is that the satisfiability problem Sat($\mathcal{L}$) for various fragments $\mathcal{L}$ of English may be studied from a purely complexity-theoretic point of view. Thus, for example, it is known that the problems Sat($\mathcal{S}$), Sat($\mathcal{S}^+$) and Sat($\mathcal{R}$) are NLOGSPACE-complete, while Sat($\mathcal{R}^+$) is EXPTIME-complete (Pratt-Hartmann and Moss, 2009). Similarly, the satisfiability problems for GRL and RCL are easily seen to be NPTIME-complete, as is the problem Sat($\mathcal{S}_{\mathrm{rn}}$); by contrast, the problem Sat($\mathcal{S}_{\mathrm{r}}$) is PTIME-complete (Pratt-Hartmann, 2014, Theorem 7). The question naturally arises as to whether the ability of neural networks to learn to solve these various satisfiability problems correlates with these complexity-theoretic differences.

Table 1shows all templates used to generate the datasets for each of the fragments. Code to generate the datasets, allowing control of the parameters discussed above, is included in the supplementary material available on github[1].

3 Random problems

In this section we outline the construction of collections of sets of formulas in the fragments $\mathcal{S}^+$, $\mathcal{R}$, $\mathcal{R}^+$, $\mathcal{S}_{\mathrm{r}}$ and $\mathcal{S}_{\mathrm{rn}}$, in which each generated set is labelled as *satisfiable* or *unsatisfiable*. (The fragment $\mathcal{S}$ is not interestingly different from $\mathcal{S}^+$, and will not be considered in the experiments reported here.)These collections are partitioned into training and evaluation sets, enabling us to test the ability of neural networks to learn to recognise satisfiability under a range of conditions.

Any sentence in $\mathcal{S}^+$ translates to a formula having either of the forms $\forall x(\pm p(x) \rightarrow \pm q(x))$ or $\exists x(\pm p(x) \wedge \pm q(x))$. We may thus generate a sentence of $\mathcal{S}^+$ pseudo-randomly by selecting a universal sentence with probability p_u, a negated subject with probability $p_{\bar{s}}$ and a negated object with probability $p_{\bar{o}}$, and then choosing p and q at random from some collection of n nouns. By carrying out this process s times, we obtain a random set Φ of $\mathcal{S}^+$-sentences ($|\Phi| = s$). For definiteness, we set $p_{\bar{s}} = p_{\bar{o}} = 0.5$ and $p_u = 0.8$. In addition, we remove any inconsistent sentences, such as "Some p is not a p." In choosing the various parameters, we fix $n/s = 0.8$, which, as we have empirically established, keeps the proportion of satisfiable instances at roughly 50%. Collections of such problem instances are created for various values of s. The ratio $n/s = 0.8$ corresponds (for $\mathcal{S}^+$) roughly to the critical region of SAT problems studied in (Richardson and Sabharwal, 2021), where it was shown that learning on data from this region is more effective than learning from uniformly sampled data. However, we need to be wary of assuming that such instances are difficult—an issue addressed in Section 4.

For $\mathcal{R}^+$, we proceed similarly, generating s sentences at random over a fixed collection of n nouns and v transitive verbs. Every sentence in $\mathcal{R}^+$ is either a sentence in $\mathcal{S}^+$ or translates to a formula having one of the forms $\forall x(\pm p(x) \rightarrow \gamma)$ or $\exists x(\pm p(x) \wedge \gamma)$, where γ is either $\forall y(\pm q(y) \rightarrow \pm r(x,y))$ or $\exists y(\pm q(y) \wedge \pm r(x,y))$. Call sentences in $\mathcal{R}^+ \setminus \mathcal{S}^+$ *relational sentences*. We may thus generate a sentence of $\mathcal{R}^+$ pseudo-randomly by choosing to produce a relational sentence with probability p_r. If we choose to produce a non-relational sentence (i.e. a sentence in $\mathcal{S}^+$), we proceed as above; otherwise a negated verb is chosen with probability $p_{\bar{v}}$, a universally quantified γ with probability p_{uu}; the other parameters, n, v, p_u, p_s and p_o are interpreted as before. By setting $p_{\bar{s}} = p_{\bar{o}} = 0$, we guarantee that every generated relational sentence has a non-negated subject and a non-negated object, and hence is a sentence of $\mathcal{R}$. By repeating this process s times, we obtain an set Φ of sentences in $\mathcal{R}^+$ (or $\mathcal{R}$). When generating instances of Sat($\mathcal{R}$), we thus set $p_r = 0.2$, $p_{\bar{s}} = p_{\bar{o}} = 0$, $p_{\bar{v}} = 0.5$ and $p_u = p_{uu} = 0.8$; in addition, we fix $n/s = 0.6$ and $v/s = 0.15$, which, as we have empirically established, keeps the ratio of satisfiable to non-satisfiable instances at roughly 50%. Likewise, when generating instances of Sat($\mathcal{R}^+$), we set $p_{\bar{s}} = p_{\bar{v}} = 0.5$, with the other parameters as for $\mathcal{R}$; however, we adjust n/s to 0.64 in order to maintain the ratio of satisfiable to non-satisfiable instances.

Finally, for fragments with relative clauses, $\mathcal{S}_{\mathrm{r}}$ and $\mathcal{S}_{\mathrm{rn}}$, in addition to the parameters introduced for $\mathcal{S}^+$, we control the probability of negated relative clauses by the parameter $p_{\bar{r}}$. By setting $p_{\bar{r}} = 0$ we guarantee that every generated sentence has an un-negated relative clause, and hence is a sentence of $\mathcal{S}_{\mathrm{r}}$. We additionally set $p_{\bar{o}} = 0.5$ and $p_u = 0.8$, and for $\mathcal{S}_{\mathrm{rn}}$, we set $p_{\bar{r}} = 0.5$. We empirically establish that setting $(n - 0.225)/s = 0.59$ yields a probability of satisfiability of approximately 0.5

[1] https://github.com/schlevik/nlr

Frag.	Templates	Example sentence
$\mathcal{S}^+$	Every/No (non-)p is a q.	Every artist is a beekeeper.
	Some (non-)p is (not) a q.	Some carpenter is not a dentist.
$\mathcal{R}$	*all of $\mathcal{S}$ and:*	
	Every/Some p rs every/some q.	Every artist chases some beekeeper.
	Some p does not r every/any q.	Some beekeeper does not chase any artist.
	No p rs every/any q.	No beekeeper bewitches any artist.
$\mathcal{R}^+$	*all of $\mathcal{S}^+$ and:*	
	Every/Some (non-)p rs every/some (non-)q.	Every non-artist chases some beekeeper.
	Some (non-)p does not r every/any (non-)q.	Some beekeeper does not chase any non-artist.
	No (non-)p rs every/any (non-) q.	No non-beekeeper bewitches any non-artist.
$\mathcal{S}_\mathrm{r}$	Every/Some/No o who is a p is a q.	Every artist who is a dentist is a carpenter.
	Some o who is a p is not a q.	Some dentist who is a hunter is not a spy.
$\mathcal{S}_\mathrm{rn}$	*all of $\mathcal{S}_\mathrm{r}$ and:*	
	Every/Some/No o who is not a p is a q.	Every artist who is a not dentist is a carpenter.
	Some o who is a not p is not a q.	Some dentist who is a not hunter is not a spy.

Table 1: Templates used to generate the problem instances for all five fragments. Round brackets () denote optionals, forward slashes / denote alternatives.

for both fragments.

The satisfiability of a generated problem instance is determined using the Vampire automated theorem prover (ATP) (Kovács and Voronkov, 2013), which (assuming termination) either reports that the input set is satisfiable or outputs a proof of a contradiction. We record, for each generated problem instance, whether it is satisfiable, and, if not, the number l of lines in the discovered proof of a contradiction (*proof length*) as well as the number of input sentences, d, used in that proof. The ATP terminated on all generated problem instances; there is, however, no general guarantee that the proofs found are the shortest possible.

Generated sentences are created in the relevant fragments of English with templates depicted in Table 1 and realised with dictionaries of nouns that describe categories for unary predicates (e.g. "artist", "beekeeper") and transitive verbs for binary predicates (e.g. "admires", "bewitches"). We use distinct vocabularies for training and evaluation data and use words with non-overlapping semantic fields. Unless stated otherwise, to maintain comparability between the different datasets, we generate 60000 examples for training and 8000 for evaluating model accuracy. This is achieved by generating 3750 and 500 (for training and evaluation sets, respectively) examples for each number of sentences s between 15 and 30.

4 Constructed problems

One attractive feature of the fragments $\mathcal{S}$, $\mathcal{S}^+$ and $\mathcal{R}$ is that their satisfiable sets of formulas admit of a simple graph-theoretic characterization. This gives us an additional means of creating data sets comprising challenging problem instances.

We illustrate with $\mathcal{S}$. Let a set Φ of $\mathcal{S}$-sentences be given, and let $p_1, \ldots, p_n$ be the common nouns (predicates) occurring in Φ. Now let V be the set of expressions $p_i(x)$ or $\neg p_i(x)$ $(1 \leq i \leq n)$. We call the elments of V *literals* and let the variables ℓ and m range over V. If $\ell \in V$, denote by $\bar{\ell}$ the opposite literal obtained by adding or removing the negation sign as appropriate. Now let E be the set of ordered pairs of literals (ℓ, m) such that Φ contains a sentence formalized as either $\forall x(\ell \rightarrow m)$ or as $\forall x(\bar{m} \rightarrow \bar{\ell})$. Thus, $G_\Phi = (V, E)$ is a directed graph. Write $\ell \Rightarrow_\Phi m$ if there is a path in G_Φ from ℓ to m. It can be shown (Pratt-Hartmann and Moss, 2009, Sec. 3) that a set of $\mathcal{S}$-formulas Φ is satisfiable if and only if Φ contains no sentence $\exists x(\ell \wedge m)$ such that either: (i) $\ell \Rightarrow_\Phi \bar{\ell}$; (ii) $m \Rightarrow_\Phi \bar{m}$ or (iii) $\ell \Rightarrow_\Phi \bar{m}$. Thus, determining (un)satisfiability in $\mathcal{S}$ amounts to detecting certain *forbidden configurations* (in this case: paths) in the directed graph G_Φ. We regard the length of the path (if it exists) as the *size* of the forbidden configuration. Satisfiability in $\mathcal{S}^+$ is characterized similarly: we simply have to check that, in addition, V contains no pair of opposite literals o and $\bar{o}$ such that $o \Rightarrow \bar{o}$ and $\bar{o} \Rightarrow o$. Again, if a set of sentences

of $\mathcal{S}^+$ is inconsistent, then it contains a forbidden configuration having a well-defined *size*.

This gives us controlled way to generate hard problem instances. Consider the fragment $\mathcal{S}^+$. We begin by simply constructing a forbidden configuration having a given size, d. To obtain an *unsatisfiable* problem instance Φ of size s, we then add $s - d$ random sentences, checking (using a simple algorithm) that doing so does not create any smaller forbidden configurations. To obtain a *satisfiable* problem instance of size s, we reverse one of the implications in the forbidden configuration, and check that the added $s - d$ random sentences do not cause an unsatisfiability. In effect, d is a guaranteed difficulty level; it yields a lower bound on the proof length required to establish unsatisfiability. At the same time, the satisfiable and unsatisfiable instances thus generated are not easily distinguished by any superficial characteristics. We denote the problem $\text{Sat}(\mathcal{S}^+)$ constructed as just described with d in a specified range as $\text{Sat}(\mathcal{S}^+_{[\cdot,\cdot]})$. Thus, for example, in $\mathcal{S}^+_{[2,6]}$, unsatisfiable problem instances have $2 \leq d \leq 6$.

For $\mathcal{R}$, the corresponding characterization of unsatisfiability in terms of forbidden configurations involves several cases (Pratt-Hartmann and Moss, 2009, Sec. 4). For simplicity, we generate difficult instances by focusing on just one of these types of forbidden configuration, which we call an $\forall\forall$-configuration. We begin by constructing an $\forall\forall$-configuration of specific size $6d$ (for $d \geq 1$). (Such a collection of sentences is always unsatisfiable.) To obtain an *unsatisfiable* problem instance of size s, we add $s - 6d$ randomly generated sentences, again checking that the resulting unsatisfiability is due entirely to the forbidden configuration. *Satisfiable* instances are then obtained by reversing one of the implications in the $\forall\forall$-configuration, and checking that this does not lead to unsatisfiability. Denote the problem $\text{Sat}(\mathcal{R})$ constructed as just described with d in a specified range as $\text{Sat}(\mathcal{R}_{\langle\cdot,\cdot\rangle})$. For example, in $\mathcal{R}_{\langle 1,2\rangle}$, inconsistent problem instances are inconsistent because of $\forall\forall$-configurations of size 6 or 12.

It is not possible to characterize $\text{Sat}(\mathcal{R}^+)$ in terms of forbidden configurations in this simple way. As a substitute, we use the proof-lengths of the proofs found by the ATP as a rough guide to difficulty. To generate difficult instances of $\mathcal{R}^+$, therefore, we first generate random instances as in Sec. 3; we then filter out those unsatisfiable instances with short proof-lengths (as reported by the ATP), and then remove satisfiable instances at random to preserve the proportion of satisfiable instances overall.

5 Experimental Setup

In the following study, we investigate whether neural networks, and in particular state-of-the-art transformer-based language models (Devlin et al., 2019) can learn to perform satisfiability checks on examples representing the selected fragments. First, we investigate whether they can do so in principle, by optimising and evaluating classifiers on training and evaluation data drawn from the same distribution. Second, in an attempt to understand whether they reliably learn the underlying logical principles that govern (un-)satisfiability, we evaluate their *generalisation capabilities* on out-of-distribution evaluation data. We do so by altering various parameters of the training and evaluation data generation to control the structure of the generated problem instances.

More specifically, we cast the problem of determining satisfiability as binary text classification and conduct experiments with three transformer based language models, RoBERTa, XLnet and Electra (Liu et al., 2019; Yang et al., 2019; Clark et al., 2020a).which are further described in the Appendix. We employ pre-trained language models because initial experiments showed that these tend to converge faster compared to training from scratch, despite the fact that their pre-training objectives bears little similarity to the task at hand. Following Devlin et al. (2019), we represent each problem in plain English text prepended by the special `[CLS]` token as input to the language model. The text is embedded using the language model, and the embedding of the `[CLS]` token, as output by the final layer of the language model, is projected into a two-dimensional space, representing the odds of the problem being satisfiable. During inference, we pick the highest logit as the model's prediction and during training we minimise the cross-entropy loss between the logits and the expected class, thus optimising the parameters of the language model to produce the expected prediction conditioned on the input. We keep the choice of hyper-parameters (detailed in Appendix) consistent across experiments.

Problem size	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_{\text{r}}$	$\mathcal{S}_{\text{rn}}$	$\mathcal{R}^+$
$15 \leq s \leq 30$	76	93	94	80	74
$30 \leq s \leq 40$[1]	64	89	94	82	72
$40 \leq s \leq 45$	63	86	-	-	71

Table 2: Accuracy of optimised models trained on random examples consisting of 15 to 30 sentences (s) and evaluated on longer random problem instances.

6 Results and Analysis

We report and analyse the results of the conducted empirical study. For all results we measure the error as a confidence interval at $\alpha = 0.05$, using asymptotic normal approximation and omit reportage for brevity, as all measures are in the range of at most two percent points. We average results obtained for all three language models.

Transformers perform well on random examples in all fragments. To seek evidence for the first question we train and evaluate separate models on randomly generated problems instances of each of the five fragments. The results are reported the first row of Table 2 and suggest that the models perform well on randomly generated problems in all fragments, even on the ExpTime-complete $\mathcal{R}^+$ fragment. Surprisingly, the obtained accuracies do not seem to correspond to the complexity classes of the elicited fragments. Note that the results reported in this table are not necessarily that of the best-performing model; for example, by training the RoBERTa model longer and on a larger dataset representing the $\mathcal{R}^+$ fragment, we obtain accuracy scores of up to 81% (from original 79%, see Table 8 in the Appendix). However, to maintain comparability and as the difference is marginal, we use the same training budget for all models optimised on different fragments. The task appears non-trivial, as a simple LSTM-based classifier was not able to outperform the majority class baseline even on the simplest $\mathcal{S}^+$ fragment and after explicit hyper-parameter optimisation.

To investigate whether this performance generalises with the size of the problem instances, we generate evaluation sets with $30 \leq s \leq 45$ sentences for longer problems. Note that we are constrained to problems of the size of up to 512 tokens, as a technical constraint of the pre-trained language model, hence we do not investigate generalisation capabilities to problems beyond 45 (37 for $\mathcal{S}_{\text{r}}$, $\mathcal{S}_{\text{rn}}$) sentences. The remainder of Table 2 shows that models generalise consistently to problems larger than seen during training, with the notable exception of the model optimised on $\mathcal{S}^+$, which exhibits the most significant drop of over 10 percent points.

Superficially, Table 2 appears to indicate good generalisation performance. However, it is important to realise that simply increasing the number of sentences does not make the reasoning problems harder, as witnessed by the fact that the number of sentences, d, required to prove contradiction does not increase: on average, $d = 3.50$ for examples with 15 to 30 sentences, and $d = 3.58$ for problems with 30 to 45 sentences. Thus, the forbidden configurations the network is identifying are, for the most part, small, even for large numbers of sentences. A complimentary analysis in the Appendix reveals further, that these forbidden configurations are likely to be common between different fragments.

Transformers are not robust to distribution shifts. Moving on to the second question, we investigate whether the optimised models truly pick up the reasoning patterns as intended or rather overfit to their training data as an artefact of the configuration of parameters controlling the stochastic generation. Approximating the "hardness" of a problem instance by its proof length l, we find that for all fragments, the overwhelming majority (ranging from 29% in $\mathcal{S}$ to 86% in $\mathcal{S}_{\text{rn}}$) of contradictory examples have short proof lengths of 12 or less, indicating that random problems are, in fact, unsatisfiable for trivial reasons which are easy to show (See also histogram in Appendix). Thus, we collect "hard" examples by (over-) generating a large body of examples and then filtering by proof length and refer to them as *random hard* problem instances. Naturally, in this way we can only capture the hardness of *inconsistent* examples, therefore the evaluation focuses on those.

When comparing the (out-of-distribution) performance of models on "easy" and "hard" non-satisfiable problem instances with proof length of at least 42 (Table 3, second and third rows), on average, there is a gap of 25 percent points, This suggests that models in fact overfit to simple problems that tend to dominate the randomly generated datasets without picking up the general principles governing reasoning in the corresponding fragments. Models optimised on "hard" training examples (proof length ≥ 22, complemented with an equal number of random consistent problems), gen-

[1] 37 for $\mathcal{S}_{\text{r}}$ and $\mathcal{S}_{\text{rn}}$ to fit transformers' 512-token limit

Train l	Eval l	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_{\text{r}}$	$\mathcal{S}_{\text{rn}}$	$\mathcal{R}^+$
$l \geq 6$ *(easy)*	satisfiable	61	87	90	53	58
	$l \leq 12$	98	99	99	70	98
	$l \geq 42$	84	74	70	42	70
$l \geq 22$ *(hard)*	satisfiable	57	89	86	50	52
	$l \leq 12$	68	61	33	61	72
	$l \geq 42$	100	94	99	96	73

Table 3: Accuracy of models trained on random satisfiable and easy/hard insatisfiable examples, and evaluated on random satisfiable and easy/hard insatisfiable examples with proof length $l \leq 12$ and $l \geq 42$.

Train On	Evaluate On	Accuracy
$\mathcal{S}^+$	$\mathcal{S}^+_{[2,6]}$	65
$\mathcal{S}^+_{l\geq 22}$	$\mathcal{S}^+_{[2,6]}$	57
$\mathcal{R}$	$\mathcal{R}_{\langle 1,2\rangle}$	53
$\mathcal{R}_{l\geq 22}$	$\mathcal{R}_{\langle 1,2\rangle}$	58
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+_{[2,10]}$	96^2
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}_{\langle 1,3\rangle}$	100
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+$	54^2
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	48
$\mathcal{S}^+_{[2,6]}+\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	88^2

Table 4: Accuracy of model optimised and evaluated on randomly generated and constructed datasets in $\mathcal{S}^+$and $\mathcal{R}$. The datasets $\mathcal{S}^+_{l\geq 22}$ and $\mathcal{R}_{l\geq 22}$ have unsatisfiable examples with at least proof length 22, while $\mathcal{S}^+_{[\cdot,\cdot]}$, $\mathcal{R}_{\langle\cdot,\cdot\rangle}$ have constructed chains in the bracketed range.

eralise well to even harder problems (proof length $\geq$ 42, Table 3, last row). This suggests that the models can learn to classify hard problem instances when presented explicitly by supplying appropriate training data. This performance does not carry over to simpler problems, however, as models optimised on harder problems exhibit a drop in accuracy when evaluated on simpler problems, as the penultimate row of Table 3 shows. In conjunction, these observations suggest that the models tend to overfit to patterns in the generated data arising from the parameterisation of the generation algorithm, rather than learning to perform satisfiability checking in a more general sense. In other words, just like a bad student of logic, they appear to "learn the proofs" rather than the logical principles behind the proofs required for successful systematic generalisation.

Transformers seem unable to reliably learn the distinct reasoning patterns. Finally, as determining non-satisfiability for the fragments $\mathcal{S}^+$ and $\mathcal{R}$ involves detection of one of a handful of forbidden configurations, we generate data that exposes precisely these forbidden configurations.

The four top rows of Table 4 show that models optimised on *random* and *random hard* datasets in $\mathcal{S}^+$ and $\mathcal{R}$ pick these patterns up to a varying degree: while all models perform better than chance, the evaluation performance drops considerably, when comparing to in-distribution performance. In the case of $\mathcal{S}^+$, breaking down the performance by chain depth, as shown in Figure 2, reveals that models optimised on *random* data perform best on examples with chain length two (essentially detecting inconsistencies of the form "All p are q", "All q are s", "Some p are not s") and fail to generalise beyond that, while models trained on *random hard* data perform best on examples with chain length five, with their performance deteriorating for examples with longer and shorter chains. This reinforces the previous point that these models appear to have learned to identify problems that have proofs with similar structure to those in their training data rather than learning to solve the problem in a more general sense.

When we both optimise and evaluate models on the constructed datasets, we find that they are capable of learning these inconsistency patterns well, and generalise to harder problems unseen during training: the RoBERTa model optimised on $\mathcal{S}^+$ with chain lengths between two and six performs just as well on evaluation data with chain lengths of up to 10 (Table 4, row five). However, other models fail to learn these patterns from constructed $\mathcal{S}^+$ data, suggesting that it is a challenging task. Similarly, all models optimised on $\mathcal{R}$ generalise to unseen lengths of the $\forall\forall$-configurations (Table 4, row six). However, even when seemingly learning these patterns, models fail to reliably transfer these capabilities beyond the constructed cases, as evidenced by the poor generalisation performance when trained on the constructed datasets and evaluated on randomly generated data (Table 4, bottom). For $\mathcal{R}$, the bad generalisation capability is expected, as the constructed dataset does not contain inconsistency patterns other than the $\forall\forall$-configurations (e.g. examples where only the non-relational statements are inconsistent). However, this is inconsistent with the case of $\mathcal{S}^+$, as all possible inconsistency patterns are covered in the constructed dataset, yet the optimised RoBERTa models fail to transfer to random $\mathcal{S}^+$ data.

[2]Results for RoBERTa only, as other models failed to converge on constructed $\mathcal{S}^+$ data.

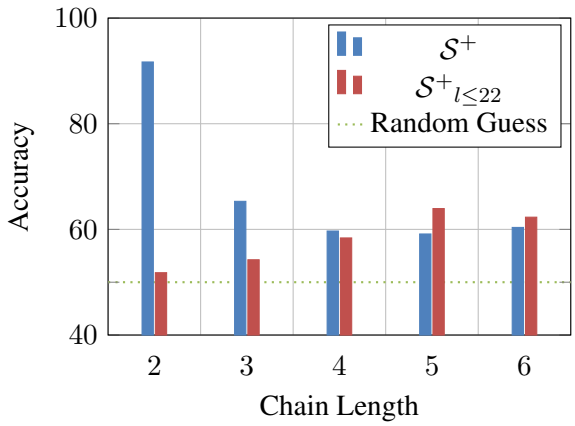

Figure 2: Accuracy by chain length for RoBERTa models optimised on $\mathcal{S}^+$ and $\mathcal{S}^+_{l\leq 22}$, evaluated on $\mathcal{S}^+_{[2,6]}$.

7 Related Work and Conclusion

The work reported in this paper represents a continuation of various studies carried out on "challenge sets", proposed to investigate the ability of (optimised) neural networks to perform different kinds of reasoning. Such studies range from monotonicity entailments (Geiger et al., 2020; Yanaka et al., 2020) to probing lexical knowledge (Glockner et al., 2018) to logical connectives (Salvatore et al., 2019; Richardson et al., 2019) or verb symmetry (Mitra et al., 2020). These studies are related to ours, in that they seek to isolate capabilities of interest and perform controlled experiments using synthetic datasets. They have in common that models optimised on crowd-sourced datasets, such as MNLI, (Williams et al., 2018), perform poorly on the challenge set data exhibiting the elicited phenomenon, but fine-tuning the optimised model on portions of these data improves the performance.

However, as Rozen et al. (2019) show, good scores after the fine-tuning probably stem from the fact that the investigated model has learned to adapt to the regularities of the challenge set rather than learning a general notion of the investigated phenomenon. Our analysis is similar: transformer-based neural networks perform well on randomly generated data, but this performance is brittle, and the models overfit to the problem space as set out by the dataset generation method. They do not generalise well to examples outside of that space, suggesting that they do not generalise *systematically* (Fodor and Pylyshyn, 1988), i.e. they struggle to identify a finite set of rules and to apply them repeatedly. Similar to our findings, research on systematic generalisation suggests that neural networks tend to generalise without systematicity in supervised learning scenarios (Johnson et al., 2017; Lake and Baroni, 2017; Goodwin et al., 2020), although these studies, unlike ours, did not concern pre-trained models.

One might argue that it is unfair to expect a statistical model that relies on correlations to learn patterns that are not obviously present in the data. However, it seems that such claims are being made in the literature (Clark et al., 2020b). While we observe that models optimised on constructed examples generalise well to harder problems unseen during training, we also show that this capability appears not to transfer to examples that are only superficially different. Thus, in our experiments, similar to Richardson and Sabharwal (2021), we find that the optimised models are not able to reliably disentangle and acquire the different reasoning patterns required to successfully complete the task of determining satisfiability.

Our study highlights one of the issues with empirically postulating neural networks various capabilities by means of good performance on challenge sets. They have only negative predictive power (Gardner et al., 2020): while low performance indicates the lack of a capability, the converse does not necessarily hold. This can be taken as a motivation to develop formal verification methods for neural networks (Shi et al., 2020), or investigate worst-case bounds for different phenomena (Raghunathan et al., 2018; Jia et al., 2019).

Our formulated task naturally allows us to expand the scope of the controlled experiments: for example, by increasing the closed-class vocabulary. Another possible avenue is to focus on improving the systematic generalisation of neural approaches, for example by providing the formulas required to prove that a set of sentences is unsatisfiable as additional supervision signal, or by relying on modular or model-based approaches (Andreas et al., 2016; Lake et al., 2013).

Limitations

The design of the study limits our findings by design - by removing the need of inducing meaning postulates (i.e. commonsense reasoning and world knowledge) we explicitly focus on logico-syntactic capabilities, analogous to how "reasoning" is defined for symbolic approaches to AI. Arguably, In "real world" application scenarios, commonsense reasoning and world knowledge cannot be fully dis-

connected from the requirement to perform reasoning, which allows our inquiry to draw fundamental conclusions about the capabilities of transformer-based language models rather than to make recommendations which are of immediate relevance to practitioners.

Our study also suffers from the inductive dilemma. We find that multiple transformer-based language models follow the trends reported in this paper, specifically that they fail to robustly identify the reasoning patterns necessary for reliably determining satisfiability in the elicited fragments. However, due to the empirical nature of this research, this finding is of course not a guarantee that some neural architecture (transformer-based or otherwise) could still perform well, when tested in our out-of-distribution evaluation settings.

Acknowledgements

The authors would like to acknowledge the use of the Computational Shared Facility at The University of Manchester and thank the anonymous reviewers from the ARR December 2021 cycle for their valuable feedback.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Neural Module Networks.

Aristotle. 1963. *Aristotle's Categories and* De Interpretatione. Clarendon Press, Oxford. (J.R. Ackrill, Tr.).

Aristotle. 1989. *Prior Analytics*. Hackett, Indianapolis, IN. (R. Smith, Tr.).

Luisa Bentivogli, Ido Dagan, Hoa Trang Dang, Danilo Giampiccolo, and Bernardo Magnini. 2009. The fifth pascal recognizing textual entailment challenge. In *In Proc Text Analysis Conference (TAC'09*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Stroudsburg, PA, USA. Association for Computational Linguistics.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017. Enhanced LSTM for Natural Language Inference. *ACL 2017 - 55th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*, 1:1657–1668.

Kevin Clark, Minh-Thang Luong, Google Brain, Quoc V Le Google Brain, and Christopher D Manning. 2020a. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. In *International Conference on Learning Representations (ICLR)*.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020b. Transformers as Soft Reasoners over Language. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pages 3882–3890. International Joint Conferences on Artificial Intelligence.

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, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jerry A. Fodor and Zenon W. Pylyshyn. 1988. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71.

Matt Gardner, Yoav Artzi, Victoria Basmov, Jonathan Berant, Ben Bogin, Sihao Chen, Pradeep Dasigi, Dheeru Dua, Yanai Elazar, Ananth Gottumukkala, Nitish Gupta, Hannaneh Hajishirzi, Gabriel Ilharco, Daniel Khashabi, Kevin Lin, Jiangming Liu, Nelson F. Liu, Phoebe Mulcaire, Qiang Ning, Sameer Singh, Noah A. Smith, Sanjay Subramanian, Reut Tsarfaty, Eric Wallace, Ally Zhang, and Ben Zhou. 2020. Evaluating Models' Local Decision Boundaries via Contrast Sets. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1307–1323, Stroudsburg, PA, USA. Association for Computational Linguistics.

Atticus Geiger, Kyle Richardson, and Christopher Potts. 2020. Neural natural language inference models partially embed theories of lexical entailment and negation. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 163–173. Association for Computational Linguistics.

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking NLI Systems with Sentences that Require Simple Lexical Inferences. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 650–655, Stroudsburg, PA, USA. Association for Computational Linguistics.

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, Stroudsburg, PA, USA. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A Smith. 2018. Annotation Artifacts in Natural Language Inference Data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, Stroudsburg, PA, USA. Association for Computational Linguistics.

Robin Jia, Aditi Raghunathan, Kerem Göksel, and Percy Liang. 2019. Certified Robustness to Adversarial Word Substitutions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4127–4140, Stroudsburg, PA, USA. Association for Computational Linguistics.

Justin Johnson, Li Fei-Fei, Bharath Hariharan, C Lawrence Zitnick, Laurens Van Der Maaten, and Ross Girshick. 2017. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2901–2910.

Laura Kovács and Andrei Voronkov. 2013. First-Order Theorem Proving and Vampire. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8044 LNCS:1–35.

Brenden M. Lake and Marco Baroni. 2017. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. *35th International Conference on Machine Learning, ICML 2018*, 7:4487–4499.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. 2013. One-shot learning by inverting a compositional causal process. *University of Toronoto web domain*, 26.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arxiv:1907.11692*.

Arindam Mitra, Ishan Shrivastava, and Chitta Baral. 2020. Enhancing Natural Language Inference Using New and Expanded Training Data Sets and New Learning Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *Autodiff Workshop @ NIPS 2017*.

I. Pratt-Hartmann and L. Moss. 2009. Logics for the relational syllogistic. *Review of Symbolic Logic*, 2(4):647–683.

Ian Pratt-Hartmann. 2014. Semantic complexity in natural language. In *The Handbook of Contemporary Semantic Theory*, 2nd edition edition, pages 429–454. Wiley Blackwell.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21:1–67.

Aditi Raghunathan, Jacob Steinhardt, and Percy Liang. 2018. Semidefinite relaxations for certifying robustness to adversarial examples. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems*, pages 10900–10910.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kyle Richardson, Hai Hu, Lawrence S. Moss, and Ashish Sabharwal. 2019. Probing Natural Language Inference Models through Semantic Fragments. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Kyle Richardson and Ashish Sabharwal. 2021. Pushing the limits of rule reasoning in transformers through natural language satisfiability. (forthcoming in AAAI 22).

Ohad Rozen, Vered Shwartz, Roee Aharoni, and Ido Dagan. 2019. Diversify Your Datasets: Analyzing Generalization via Controlled Variance in Adversarial Datasets. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 196–205, Stroudsburg, PA, USA. Association for Computational Linguistics.

Felipe Salvatore, Marcelo Finger, and Roberto Hirata Jr. 2019. A logical-based corpus for cross-lingual evaluation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 22–30, Stroudsburg, PA, USA. Association for Computational Linguistics.

Viktor Schlegel, Goran Nenadic, and Riza Batista-Navarro. 2022. A survey of methods for revealing and overcoming weaknesses of data-driven Natural Language Understanding. *Natural Language Engineering*, pages 1–31.

Viktor Schlegel, Marco Valentino, André Andre Freitas, Goran Nenadic, and Riza Batista-Navarro. 2020. A Framework for Evaluation of Machine Reading Comprehension Gold Standards. In *Proceedings of The 12th Language Resources and Evaluation Conference*, pages 5359–5369, Marseille, France. European Language Resources Association.

Zhouxing Shi, Huan Zhang, Kai-Wei Chang, Minlie Huang, and Cho-Jui Hsieh. 2020. Robustness Verification for Transformers. In *8th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and Policy Considerations for Deep Learning in NLP. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650, Stroudsburg, PA, USA. Association for Computational Linguistics.

Saku Sugawara, Kentaro Inui, Satoshi Sekine, and Akiko Aizawa. 2018. What Makes Reading Comprehension Questions Easier? In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4208–4219, Stroudsburg, PA, USA. Association for Computational Linguistics.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-Of-Thought: Teaching Pre-Trained Models to Systematically Reason Over Implicit Knowledge. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*.

Jidong Tian, Yitian Li, Wenqing Chen, Liqiang Xiao, Hao He, and Yaohui Jin. 2021. Diagnosing the first-order logical reasoning ability through LogicNLI. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3738–3747, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008.

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, Stroudsburg, PA, USA. 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, Stroudsburg, PA, USA. Association for Computational Linguistics.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. 2019. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *In Proceedings of Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, pages 5753–5763.

A Additional Details on the Experimental Setup

Regarding models, We choose RoBERTa, Electra and XLnet as representatives of "BERTology" as they all employ different pretraining objectives to obtain the contextualised representations. We do not perform explicit hyperparemeter optimisation for each of the models for each of the datasets, as we are not invested in finding a best performing model, but rather, we are concerned with more general questions about the learnability of the presented problems. We find that our obtained results are similar, and in this regard we expect results to be similar for other transformer-based approaches, as their performance stems from the amount of pre-training data and language model size, rather than architectural choices (Raffel et al., 2020). Finally, investigating multiple models (e.g. more transformer models or hyperparemeter optimisation) increases the amount of computation and thus the carbon footprint (Strubell et al., 2019), which we deem unnecessary given the research questions.

As both data generation and model optimisation are stochastic processes, we are concerned with the impact of chance on the results of our experiments. To investigate whether model convergence is impacted, we optimise five models on $\mathcal{R}$ datasets generated from different random seeds. For model performance, we evaluate one optimised model on five different $\mathcal{R}$ evaluation sets. The results are summarised in Table 5: the variance of evaluation scores is negligible, while for optimisation, the influence of randomness is more noticeable. The presented loss variance translates to differences in accuracy of up to 2 percent points.

B Additional training details

We implement the training and inference in PyTorch 1.10.0 (Paszke et al., 2017). We use the pre-trained language models available in the `transformers`[2] library. We train the `roberta` model on an Nvidia V100 GPU with 16 GB of RAM. to keep a consistent set of hyperparamters, We train the XLNet models on an Nvidia A100 GPU with 80GB of RAM since these models do not support gradient checkpointing, and the chosen batch size results in large ($\geq$ 16GB) memory requirements during training.

We fix the random seed to maintain deterministic behaviour and the hyper-parameters used for training all models are

- **Batch size:** relying on gradient checkpointing, we are able to set the batch size to 56.
- **Learning Rate:** We schedule the learning rate to linearly warm up from zero to $4 \cdot 10^{-6}$, linearly decaying it to zero, as it was found to perform best across all fragments. We use the ADAM optimiser with the default parameters $\epsilon = 1 \times 10^{-8}, \beta_1 = 0.99$ and $\beta_2 = 0.999$. Note that this comparatively low learning rate prohibits us from using mixed precision optimisation.
- **Train Epochs:** We train the models for 6 epochs on all fragments to maintain the same training budget.
- **Maximal sequence length**: As the input length varies for different fragments, we ensure that the sequence lengths are set in a way that allows to embed. In practice this varies from 288 to 432 tokens. Note that padding input sequences to max length does not impact the training procedure as the padding tokens are not attended to when calculating the embedding of the `[CLS]` input token (nor for any other input tokens).

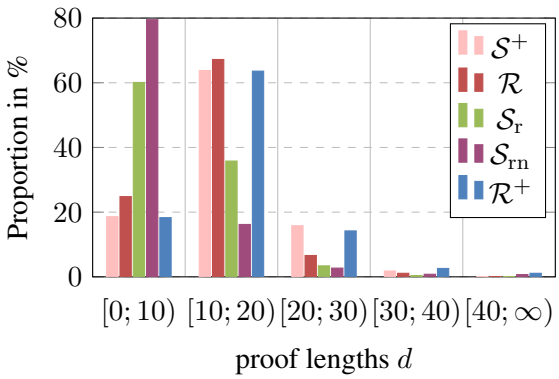

Figure 3: Proof length distribution.

[2] https://github.com/huggingface/transformers

Experiment	Metric	Mean	Std. dev.
train stability	Loss	0.07696	0.008
eval stability	Accuracy	0.953	0.0013

Table 5: Impact of randomness in the data generation process on model optimisation (first row) and on model performance (second row).

Evaluated on ↓	Trained on →					
	maj. class	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_\text{r}$	$\mathcal{S}_\text{rn}$	$\mathcal{R}^+$
$\mathcal{S}^+$	54	**86**	61	61	55	70
$\mathcal{R}$	52	79	**95**	78	54	85
$\mathcal{S}_\text{r}$	55	45	48	**96**	90	49
$\mathcal{S}_\text{rn}$	53	47	50	78	**91**	48
$\mathcal{R}^+$	53	73	63	60	55	**79**

Table 6: Accuracy of RoBERTa models evaluated on all fragments. Grey results denote generalisation performance to *harder* fragments.

We find this setting works well for all conducted experiments, thus we keep the same set of hyper-parameters to maintain comparability. To replicate our experiments, please see the separately supplied code.

C Additional Results

We further investigate whether transformers—perhaps due to their pre-training—can generalise to fragments they have *not* encountered during training. To that end, we evaluate the models on *all* fragments, not only those they have been optimised on. The results are reported in Table 6 and reveal counter-intuitive patterns. Surprisingly, none of the trained models generalises particularly well to the *simpler* $\mathcal{S}^+$ fragment (Table 6, first row). While the results of models trained on the $\mathcal{S}_\text{r}$ and $\mathcal{S}_\text{rn}$ fragments could be explained by the different problem structure (during training, these models do not encounter sentences as they appear in $\mathcal{S}^+$), the same explanation is not valid for the $\mathcal{R}$ and $\mathcal{R}^+$ fragments: problems in these fragments consist of 80% syllogistic statements on average. In fact, an estimated 90% of the unsatisfiable problems in $\mathcal{R}$ and $\mathcal{R}^+$ contain an inconsistency in the non-relational statements. Therefore, it is reasonable to expect models optimised on $\mathcal{R}$ and $\mathcal{R}^+$ to perform well on $\mathcal{S}^+$, which appears not to be the case. Contrariwise, models optimised on the simpler fragments $\mathcal{S}^+$ fragment do generalise well to the harder fragments $\mathcal{R}$ and $\mathcal{R}^+$, as one would expect given their high number of non-relational inconsistencies. This contradictory evidence suggests that the models struggle to reliably identify the configurations leading to insatisfiability. Another observation that eludes a simple explanation is the good performance on the $\mathcal{R}$ fragment of the model optimised on $\mathcal{S}_\text{r}$, as these fragments are generated from non-overlapping sentence templates.

Problem size	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_\text{r}$	$\mathcal{S}_\text{rn}$	$\mathcal{R}^+$
$15 \leq s \leq 30$	82	96	95	91	82
$30 \leq s \leq 40$	72	92	94	93	77
$40 \leq s \leq 45$	70	89	-	-	76

Table 7: Accuracy of optimised Electra models trained on random examples consisting of 15 to 30 sentences (s) and evaluated on longer random problem instances.

Problem size	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_\text{r}$	$\mathcal{S}_\text{rn}$	$\mathcal{R}^+$
$15 \leq s \leq 30$	86	95	96	91	79
$30 \leq s \leq 40$	62	92	95	94	76
$40 \leq s \leq 45$	61	90	-	-	75

Table 8: Accuracy of optimised RoBERTa models trained on random examples consisting of 15 to 30 sentences (s) and evaluated on longer random problem instances.

Problem size	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_\text{r}$	$\mathcal{S}_\text{rn}$	$\mathcal{R}^+$
$15 \leq s \leq 30$	61	89	92	60	62
$30 \leq s \leq 40$	60	83	93	60	63
$40 \leq s \leq 45$	59	80	-	-	64

Table 9: Accuracy of optimised XLNet models trained on random examples consisting of 15 to 30 sentences (s) and evaluated on longer random problem instances.

Training the model on a combined sample of 12000 problem instances from all fragments results in accuracy scores of 73%, 89%, 94%, 94% and 74%, for the fragments $\mathcal{S}^+$, $\mathcal{R}$, $\mathcal{S}_\text{r}$, $\mathcal{S}_\text{rn}$ and $\mathcal{R}^+$, respectively.

The distribution of different proof lengths for randomly generated data is shown in Figure 3 and supports the hypothesis that most randomly generated inconsistent problem instances are "easy" in the sense that they have short proof lengths that lead to refutation.

Figure 2 shows the breakdown by chain length for models optimised on random and random hard $\mathcal{S}^+$ data and evaluated on constructed $\mathcal{S}^+$ examples.

Finally, Tables 7-15 report all performances discussed in the main paper broken down by model. We see similar trends across models, with Electra performing best and XLNet often performing worst. We caution to over-interpret these differences, as these could be due to the amount of pre-training of each architecture and the resulting sensitivity to hyper-parameters (Lan et al., 2020).

Train l	**Eval** l	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_{\mathrm{r}}$	$\mathcal{S}_{\mathrm{rn}}$	$\mathcal{R}^+$
$l \geq 6$ *(easy)*	satisfiable	72	92	91	86	71
	$l \leq 12$	99	100	100	100	98
	$l \geq 42$	92	77	79	48	66
$l \geq 22$ *(hard)*	satisfiable	74	92	89	51	65
	$l \leq 12$	57	64	30	59	64
	$l \geq 42$	100	95	99	96	79

Table 10: Accuracy of Electra models trained on random satisfiable and easy/hard insatisfiable examples, and evaluated on random satisfiable and easy/hard insatisfiable examples.

Train l	**Eval** l	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_{\mathrm{r}}$	$\mathcal{S}_{\mathrm{rn}}$	$\mathcal{R}^+$
$l \geq 6$ *(easy)*	satisfiable	78	92	92	87	72
	$l \leq 12$	99	99	99	99	97
	$l \geq 42$	71	68	64	37	56
$l \geq 22$ *(hard)*	satisfiable	60	90	86	48	53
	$l \leq 12$	64	63	28	67	72
	$l \geq 42$	100	94	98	99	75

Table 11: Accuracy of RoBERTa models trained on random satisfiable and easy/hard insatisfiable examples, and evaluated on random satisfiable and easy/hard insatisfiable examples.

Train l	**Eval** l	$\mathcal{S}^+$	$\mathcal{R}$	$\mathcal{S}_{\mathrm{r}}$	$\mathcal{S}_{\mathrm{rn}}$	$\mathcal{R}^+$
$l \geq 6$ *(easy)*	satisfiable	33	77	87	53	31
	$l \leq 12$	95	99	99	70	98
	$l \geq 42$	91	78	68	41	90
$l \geq 22$ *(hard)*	satisfiable	37	85	84	50	38
	$l \leq 12$	84	55	32	57	81
	$l \geq 42$	100	93	99	92	67

Table 12: Accuracy of XLNet models trained on random satisfiable and easy/hard insatisfiable examples, and evaluated on random satisfiable and easy/hard insatisfiable examples.

Train On	**Evaluate On**	**Accuracy**
$\mathcal{S}^+$	$\mathcal{S}^+_{[2,6]}$	70
$\mathcal{S}^+_{l\geq 22}$	$\mathcal{S}^+_{[2,6]}$	58
$\mathcal{R}$	$\mathcal{R}_{\langle 1,2\rangle}$	51
$\mathcal{R}_{l\geq 22}$	$\mathcal{R}_{\langle 1,2\rangle}$	61
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+_{[2,10]}$	–
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}_{\langle 1,3\rangle}$	100
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+$	–
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	48
$\mathcal{S}^+_{[2,6]}$+$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	–

Table 13: Accuracy of Electra models optimised and evaluated on random and constructed datasets in $\mathcal{S}^+$,$\mathcal{R}$.

Train On	**Evaluate On**	**Accuracy**
$\mathcal{S}^+$	$\mathcal{S}^+_{[2,6]}$	68
$\mathcal{S}^+_{l\geq 22}$	$\mathcal{S}^+_{[2,6]}$	58
$\mathcal{R}$	$\mathcal{R}_{\langle 1,2\rangle}$	57
$\mathcal{R}_{l\geq 22}$	$\mathcal{R}_{\langle 1,2\rangle}$	58
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+_{[2,10]}$	96
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}_{\langle 1,3\rangle}$	100
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+$	54
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	48
$\mathcal{S}^+_{[2,6]}$+$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	88

Table 14: Accuracy of RoBERTa models optimised and evaluated on random and constructed datasets in $\mathcal{S}^+$,$\mathcal{R}$.

Train On	**Evaluate On**	**Accuracy**
$\mathcal{S}^+$	$\mathcal{S}^+_{[2,6]}$	56
$\mathcal{S}^+_{l\geq 22}$	$\mathcal{S}^+_{[2,6]}$	56
$\mathcal{R}$	$\mathcal{R}_{\langle 1,2\rangle}$	52
$\mathcal{R}_{l\geq 22}$	$\mathcal{R}_{\langle 1,2\rangle}$	55
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+_{[2,10]}$	–
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}_{\langle 1,3\rangle}$	100
$\mathcal{S}^+_{[2,6]}$	$\mathcal{S}^+$	–
$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	48
$\mathcal{S}^+_{[2,6]}$+$\mathcal{R}_{\langle 1,2\rangle}$	$\mathcal{R}$	–

Table 15: Accuracy of XLNet models optimised and evaluated on random and constructed datasets in $\mathcal{S}^+$,$\mathcal{R}$.

D $\forall\forall$-configurations in $\mathcal{R}$

In the fragment $\mathcal{R}$, satisfiability is characterized by a finite number (half a dozen or so) of so-called *forbidden configurations*: families of unsatisfiable sets of formulas, with the instances of each family characterized by a numerical parameter related to that instance's cardinality. It can be shown that any unsatisfiable set of $\mathcal{R}$-formulas contains an instance of one of these forbidden configurations. The $\forall\forall$-configuration is the most complex of these, and, therefore, the hardest to learn to recognize. Hard unsatisfiable formula sets in $\mathcal{R}$ were constructed in the experiments reported here using the $\forall\forall$-configuration. We briefly outline its form here. We use abbreviated logical notation, writing $\forall(p, q)$ instead of $\forall x(p(x) \rightarrow q(x))$, or $\forall(p, \exists(q, \neg r))$ instead of $\forall x(p(x) \rightarrow \exists y(\neg r(x, y) \wedge q(y)))$, and so on.

An instance of a $\forall\forall$-configuration with parameter d consists of six sets of d formulas (hence, $6d$ in all). The first two lists entail that all ps are o_1s

and all ps are o_2s:

$$\forall(p, p_1), \ldots, \forall(p_{d-1}, o_1)$$
$$\forall(p, p'_1), \ldots, \forall(p'_{d-1}, o_2).$$

It follows that, if some ps exist, then some o_1s are o_2s. The second two lists entail that all qs are related by r to all o_1s and to no o_2s:

$$\forall(q, q_1), \ldots, \forall(q_{d-2}, q_{d-1}), \forall(q_{d-1}, \forall(o_1, r))$$
$$\forall(q, q'_1), \ldots, \forall(q'_{d-2}, q'_{d-1}), \forall(q'_{d-1}, \forall(o_2, \neg r)).$$

It follows that, if some qs exist, then no o_1s are o_2s. Hence, these four lists entail that, if some ps exist, then no qs exist. The final two lists entail that there are both ps and qs:

$$\exists(u_0, u_1), \forall(u_1, \exists(u_2, \pm r)), \ldots, \forall(u_{d-1}, \exists(p, \pm r))$$
$$\exists(u'_0, u'_1), \forall(u'_1, \exists(u'_2, \pm r)), \ldots, \forall(u'_{d-1}, \exists(q, \pm r)).$$