Predicting the Understandability of OWL Inferences

Tu Anh T. Nguyen, Richard Power, Paul Piwek, Sandra Williams

Department of Computing, The Open University, Milton Keynes, UK
{t.nguyen,r.power,p.piwek,s.h.williams}@open.ac.uk

Abstract. In this paper, we describe a method for predicting the understandability level of inferences with OWL. Specifically, we present a probabilistic model for measuring the understandability of a multiple-step inference based on the measurement of the understandability of individual inference steps. We also present an evaluation study which confirms that our model works relatively well for two-step inferences with OWL. This model has been applied in our research on generating accessible explanations for an entailment of OWL ontologies, to determine the most understandable inference among alternatives, from which the final explanation is generated.

1 Introduction

The emergence of the semantic web community during the last decade has led to agreement on a common ontology language for exchanging knowledge called OWL (Web Ontology Language) [1]. Since being adopted as a standard language by the W3C in 2004, OWL has become widespread in many domains. Research on reasoning services for automatically computing logical inferences from OWL ontologies has also been intensively investigated since then, and resulted in automated reasoners such as FaCT++ [15], Pellet [14], and HermiT [10]. However, there has been little research investigating the *cognitive* difficulty of OWL inferences for humans, which is an essential problem in ontology debugging.

An important tool in debugging ontologies is to inspect entailments generated by an automated reasoner. An obviously incorrect entailed statement such as *SubClassOf(Person,Movie)* ("Every person is a movie") signals that something has gone wrong. However, many developers, especially those with limited knowledge of OWL, will need more information in order to make the necessary corrections: they need to understand *why* this entailment follows from the ontology, before they can start to repair it. Various *axiom pinpointing* tools have been proposed to compute *justifications* of an entailment—defined as any minimal subset of the ontology from which the entailment can be drawn—including both reasoner-dependent approaches [13, 2] and reasoner-independent approaches [7, 6]. A justification provides a set of premises for an entailment, so is helpful for diagnosing an erroneous entailment; however, unlike a proof, it does not explain how the premises combine with each other to produce the entailment. A user

study [5] has shown that for many justifications (an example is shown in Table 1) even OWL experts were unable to work out how the conclusion follows from the premises without further explanation. For non-expert developers, the opacity of standard OWL syntaxes such as OWL/RDF, which are designed for efficient processing by computer programs and not for fast comprehension by people, can be another obstacle. As a possible solution to this problem, we are developing a system that explains, in English, why an entailment follows from an ontology.

Table 1. An example explanation generated by our prototype

Input	**Entailment:** *SubClassOf(Person,Movie)* **Justification:** 1. *EquivalentClasses(GoodMovie,ObjectAllValuesFrom(hasRating,FourStarRating))* 2. *ObjectPropertyDomain(hasRating,Movie)* 3. *SubClassOf(GoodMovie,StarRatedMovie)* 4. *SubClassOf(StarRatedMovie,Movie)*
Output	The statement "Every person is a movie" follows because: - everything is a movie (a). Statement (a) follows because: - anything that has as rating something is a movie (from axiom 2), and - everything that has no rating at all is a movie (b). Statement (b) follows because: - everything that has no rating at all is a good movie (c), and - every good movie is a movie (d). Statement (c) follows because axiom 1 in the justification means that "a good movie is anything that has as rating nothing at all, or has as rating only four-star ratings". Statement (d) follows because: - every good movie is a star rated movie (from axiom 3), and - every star rated movie is a movie (from axiom 4).

Table 1 shows an explanation generated by our prototype for the (obviously absurd) entailment "Every person is a movie" based on the proof tree in Figure 1. The key to understanding this proof lies in the step from axiom 1 to statement (c), which is an example of an inference in need of "further elucidation".

To generate such explanations, our system starts from a justification of the entailment, which can be computed using the method described by Kalyanpur et al. [7], and constructs *proof trees* in which the root node is the entailment, the terminal nodes are the axioms in the justification, and other nodes are intermediate statements (i.e., lemmas). Proof trees are constructed from a set of intuitively plausible *deduction rules* which account for a large collection of deduction patterns, with each local tree corresponding to a rule. For a given justification, the deduction rules might allow several proof trees, in which case

we need a criterion for choosing the most understandable one.[1] From the selected proof tree, the system generates an English explanation. Hard inference steps will be identified, and further elucidation will be added when necessary to make them understandable for most people. Such an explanation should be easier to understand than one based on the justification alone, as it replaces a single complex inference step with a number of simpler steps.

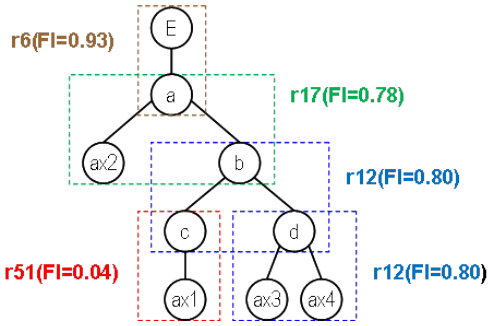

Fig. 1. The proof tree of the explanation in Table 1. The labels r6 etc. refer to rules listed in Table 4. FI values represent how easy it is to understand the rules—their *Facility Indexes*—with values ranging from 0.0 (hardest) to 1.0 (easiest).

As mentioned before, there may be multiple proof trees linking a justification to an entailment, and so multiple potential explanations of how the justification and the entailment are connected, some of which may be easier to follow than others. Therefore, being able to predict the understandability of a proof tree would be of great help in planning effective explanations for a given entailment. Specifically, it would enable the system to identify the most understandable explanation for a given justification. Additionally, when multiple justifications for an entailment are found, it would enable the system to sort explanations in order of decreasing understandability, which is very useful for end-users.

In prior work [12], we described how our current set of deduction rules was collected through analysis of a large corpus of approximately 500 OWL ontologies, and reported on an empirical study that allowed us to assign understandability indexes to the deduction rules. We called these indexes *facility indexes* (FIs). An FI of a deduction rule provides our best estimate of the probability that a person will understand the relevant inference step—i.e., that a person will recognise that the conclusion of the rule follows from the premises. Therefore, it ranges from 0.0 to 1.0, and the higher it is, the easier the inference. The result of this work is a list of 51 single-step inferences with known FIs, as shown in Table 4 (at the end of this paper) with the inferences sorted by FIs.

[1] Alternatively the deduction rules might not yield any proof trees, in which case the system has to fall back on simply verbalising the justification. Obviously such cases will become rarer as we expand the set of rules.

This paper focusses on the understandability of an entire proof tree. A proof tree can be viewed as a complex inference. When a tree has no lemma nodes, it corresponds to a single-step inference. Otherwise, it corresponds to a multiple-step inference, as in Figure 1. We propose here a model which predicts the understandability of a multiple-step inference based on the FIs of individual steps. We also report on an evaluation study which confirms that our model works relatively well in detecting differences in understandability of two-step inferences. In this study we analysed both the participants' subjective reported understanding (how difficult they found the task), and their objective performance on the task (how often they got it right). The proposed model has been applied in our system to identify the best explanation for a given justification as well as to sort explanations by decreasing understandability when multiple justifications for a given entailment are found. We envisage that this model can be used by others to predict the understandability of different kinds of inferences.

2 Related Work

Several support tools have been proposed to help ontology developers to identify the causes of class unsatisfiability [8], and to rewrite potentially problematic axioms [9]. Two studies have been conducted [8, 9] to evaluate how ontology developers debug ontologies with and without the tools. However, these studies focus on how people with a good understanding of OWL perform debugging, but not on how well they understand OWL inferences.

In a deduction rule, the conclusion can be viewed as an entailment, and the premises can be viewed as a justification of the entailment. Horridge et al. have proposed a model for measuring the cognitive difficulty of a justification [3]. In this model, they provide a list of *components*, each of which has an associated weight. For a given justification, the model checks for all appearances of these components, sums the weighted number of occurrences of the components, and outputs the result as the justification's difficulty score. The choice of the components and their weights is based on the authors' observations from an exploratory study [5] and their intuitions. Moreover, most of the proposed components are based on the syntactic analysis of justifications such as the number of premises in a justification, and these syntax-based components are mostly assigned a high weight. There are also several components for revealing difficult phenomena such as the trivial satisfaction of universal restriction[2] in OWL; however, the weights of these components are often low and are chosen intuitively. Therefore, this model predicts the difficulty of a justification in a manner that is biassed towards its structural complexity rather than its cognitive difficulty.

An empirical study was conducted by the model's authors to evaluate how well it predicts the difficulty of justifications. In this study, they created a deduction problem, presented in Manchester OWL Syntax [4] with alpha-numeric characters as class and property names, for testing a justification. In each problem, a

[2] That is, if $\langle x, y\rangle \notin R^{\mathcal{I}}$ for all $y \in \Delta^{\mathcal{I}}$ then $x \in (\forall R.C)^{\mathcal{I}}$.

justification and its entailment were given and subjects were asked whether the justification implied the entailment. A weakness of this study was that response bias was not controlled—i.e., if subjects had a positive response bias then they would have answered most questions correctly. Additionally, this study tested the model based on analysis of subjective understanding only.

The above-mentioned complexity model and evaluation study were, in fact, inspired by those of Newstead et al. [11], which were proposed for measuring the difficulty of "Analytical Reasoning" (AR) problems in Graduate Record Examination (GRE) tests. An AR problem is a deductive reasoning problem in which an initial scenario is given along with a number of constraints called *rules*, and the examinee is asked to determine a possible solution for the problem among five choices. Like Horridge et al., Newstead et al. identified a set of difficulty factors and their weights through an intensive pilot study, and they built a preliminary difficulty model based on these factors and weights. After that, a series of large-scale studies was conducted to validate as well as adjust the model. Leaving aside the fact that these reasoning problems are different from OWL inferences, a strength of this work was that response bias of all types was successfully controlled. However, in both Newstead et al.'s and Horridge et al.'s work there was no clear explanation of how weights were assigned, suggesting that the choice might have been based partly on intuition.

3 An Understandability Model

This section describes our model for predicting the understandability of an OWL inference. Of course there is no fixed understandability for a given OWL inference as it depends on the readers' knowledge of OWL as well as their deductive reasoning ability. For this reason, it is impossible to provide an accurate measurement of the understandability of an inference that is correct for most people. However, what we expect from this model is the ability to detect the *difference* in the understandability between any two inferences. For example, if an inference is easier than another then we expect that our model will be able to detect it.

In prior work [12], we reported an empirical study for measuring the understandability of deduction rules that have been combined to construct proof trees for OWL justifications. A deduction rule is an inferential step from premises to a conclusion, which cannot be effectively simplified by introducing substeps (and hence, intermediate conclusions). Therefore, the understandability of a rule is, in fact, the understandability of the associated single-step OWL inference.

To measure the understandability of a deduction rule, we devised a deduction problem in which premises of the rule were given in English, replacing class or property variables by fictional nouns and verbs so that the reader would not be biassed by domain knowledge, and the subjects were asked whether the entailment of the rule followed from the premises.[3] The correct answer was always

[3] Fictional words are nonsense words selected from various sources, such as Lewis Carroll's Jabberwocky poem (`http://en.wikipedia.org/wiki/Jabberwocky`), an automatic generator (`http://www.soybomb.com/tricks/words/`), and so on.

"Follows". To control for response bias (i.e., favouring a positive, or a negative, answer to *any* question), we included easy questions for both "Follows" and "Does not Follow" as *control questions* (as opposed to *test questions*). The complete discussion of the design of this study can be found in [12].

We used the proportion of correct answers for each test question as an index of understandability of the associated deduction rule, which we call its *facility index*. This index provides our best estimate of the probability that a person will understand the relevant inference step—i.e., that a person will recognise that the conclusion follows from the premises. Therefore, it ranges from 0.00 to 1.00, and the higher this value, obviously, the easier. Values of the FI for 51 rules tested in this study are shown in Table 4, ordered from high values to low. In this table, the rules r6, r12, and r17 used in the explanation in Table 1 are relatively easy, with FIs of 0.93, 0.80, and 0.78. By contrast rule r51, which infers statement (c) from axiom 1 in the example, is the hardest, with an FI of only 0.04.

To understand a more complex inference consisting of multiple inference steps, it is essential to be able to understand each individual inference step within it. Given a proof tree with FIs assigned to each inference step, such as the proof tree in Figure 1, a natural method of combining indexes would be to multiply them, so computing the joint probability of all steps being followed—in other words, the *facility index* of the proof tree. As before, the higher this value, the easier the proof tree. According to this model, the understandability of the proof tree in Figure 1 would be 0.93*0.78*0.80*0.04*0.80 or 0.02, indicating that the proof tree is very difficult to understand. This prediction is supported by the claim from the study conducted by Horridge and colleagues that this inference is very difficult even for OWL experts [5].

4 An Evaluation Study

In this section we report an experiment for evaluating our proposed model. We focussed on how well the model can detect differences in understandability between inferences. We adapted the use of *bins* for grouping inferences having close FIs from the study of Horridge et al. [3], but used a different experimental protocol and materials. Moreover, as mentioned in Section 1, both objective and subjective understanding of the subjects were analysed.[4]

4.1 Materials

We carried out the study with 15 proof trees collected from our ontology corpus. Each proof tree was assigned to an understandability bin on the basis of the FI predicted by our model. For our purpose, a total of five understandability bins were constructed over the range from 0.00 to 1.00, each with an interval of 0.20.[5]

[4] All the materials and results of this study can found at `http://mcs.open.ac.uk/nlg/SWAT/ESWC2013.html`.

[5] The ranges of the five bins were as follows: (B1) $0.80<x\leq1.00$, (B2) $0.60<x\leq0.80$, (B3) $0.40<x\leq0.60$, (B4) $0.20<x\leq0.40$, and (B5) $0\leq x\leq0.20$, respectively. B1 is the easiest bin and B5 is the hardest bin.

The test proof trees were selected so that there would be three for each bin, and additionally they would cover as many deduction rules as possible. In fact, our test proof trees included 25 of 51 rules from Table 4. For simplicity we only tested proof trees consisting of exactly two deduction rules (i.e., two-step inferences). The list of tested inferences and their predicted FIs is shown in Table 2.

Table 2. The list of tested inferences and their predicted FIs

ID	Tested Inference	FI	ID	Tested Inference	FI
1.1	EqvCla(C0,C1) ∧ObjPropDom(r0,C0) →ObjPropDom(r0,C1) *(Rules used: r3, r1)*	**0.96**	2.1	ObjPropRng(r0,C1) ∧SymObjProp(r0) ∧SubClaOf(C1,C0) →ObjPropDom(r0,C0) *(Rules used: r18, r3)*	**0.74**
1.2	SubClaOf(ObjUniOf(C0,C1),C2) ∧SubClaOf(C0,C3) →SubClaOf(C0,ObjIntOf(C2,C3)) *(Rules used: r4, r5)*	**0.90**	2.2	SubClaOf(C0,C1) ∧SubClaOf(C1,C2) ∧ObjPropRng(r0,C0) →ObjPropRng(r0,C2) *(Rules used: r12, r8)*	**0.72**
1.3	SubClaOf(C0,ObjIntOf(C1,C2)) ∧ObjPropRng(r0,C0) →ObjPropRng(r0,C1) *(Rules used: r2, r8)*	**0.86**	2.3	EqvCla(C1,ObjUniOf(C2,C3)) ∧SubClaOf(C0,C2) →SubClaOf(C0,C1) *(Rules used: r10, r12)*	**0.66**
3.1	SubClaOf(ObjCompOf(C1),C2) ∧SubClaOf(C1,C0) ∧SubClaOf(C2,C0) →SubClaOf(⊤,C0) *(Rules used: r25, r24)*	**0.53**	4.1	ObjPropRng(r0,C1) ∧InvObjProp(r1,r0) ∧SubClaOf(C0,ObjSomValF(r1,C2)) →SubClaOf(C0,C1) *(Rules used: r44, r9)*	**0.34**
3.2	SubObjPpOf(r0,r1) ∧SubObjPpOf(r1,r2) ∧ObjPropDom(r2,C0) →ObjPropDom(r0,C0) *(Rules used: r14, r33)*	**0.48**	4.2	SubClaOf(C0,ObjSomValF(r0,C2)) ∧ObjPropRng(r0,C1) ∧DisCla(C1,C2) →SubClaOf(C0,⊥) *(Rules used: r30, r40)*	**0.32**
3.3	SubClaOf(C0,ObjMinCard(1,r1,C2)) ∧SubObjPpOf(r1,r0) ∧SubClaOf(ObjSomValF(r0,C2),C1) →SubClaOf(C0,C1) *(Rules used: r37, r11)*	**0.45**	4.3	SubClaOf(C2,ObjAllValF(r0,C1)) ∧InvObjProp(r0,r1) ∧SubClaOf(C0,ObjSomValF(r1,C2)) →SubClaOf(C0,C1) *(Rules used: r48, r12)*	**0.26**
5.1	FunDataProp(d0) ∧SubClaOf(C0,DataHasVal(d0,l0⋆DT0)) ∧SubClaOf(C0,DataHasVal(d0,l1⋆DT0)), l1≠l0 ∧SubClaOf(C1,ObjMinCard(2,r0,C0)) →SubClassOf(C1,⊥) *(Rules used: r45, r42)*	**0.18**			
5.2	SubClaOf(C1,ObjSomValF(r0,DataHasVal(d0,l0⋆DT0))) ∧DataPropRng(d0,DT1), D0 and DT1 are disjoint ∧SubClaOf(C0,ObjSomValF(r1,C1)) →SubClassOf(C0,⊥) *(Rules used: r49, r42)*	**0.09**			
5.3	EqvCla(C0,ObjAllValF(r0,C1)) ∧ObjPropDom(r0,C0) →SubClaOf(⊤,C0) *(Rules used: r51, r17)*	**0.03**			

For each proof tree, we devised a *test problem* in which the proof tree was given to the subjects in the form of a simple explanation in English, and the subjects were asked whether the explanation is correct. We also asked the subject to rank how difficult they found the question on a scale from 5 (very easy) to 1 (very difficult). When presenting the test proof trees, we used fictional nouns and verbs so that the reader would not be biassed by domain knowledge, and labels such as (a), (b), and so on, to help subjects in locating the statements quicker. Since the correct answers to all test questions were "Yes", we controlled for response bias (i.e., favouring either positive or negative answers) by including a number of *control problems* as well as test problems. An example test problem in our study is shown in Figure 2.

Our control problems were designed to be similar to our test problems but were obvious to subjects who did the test seriously (rather than responding

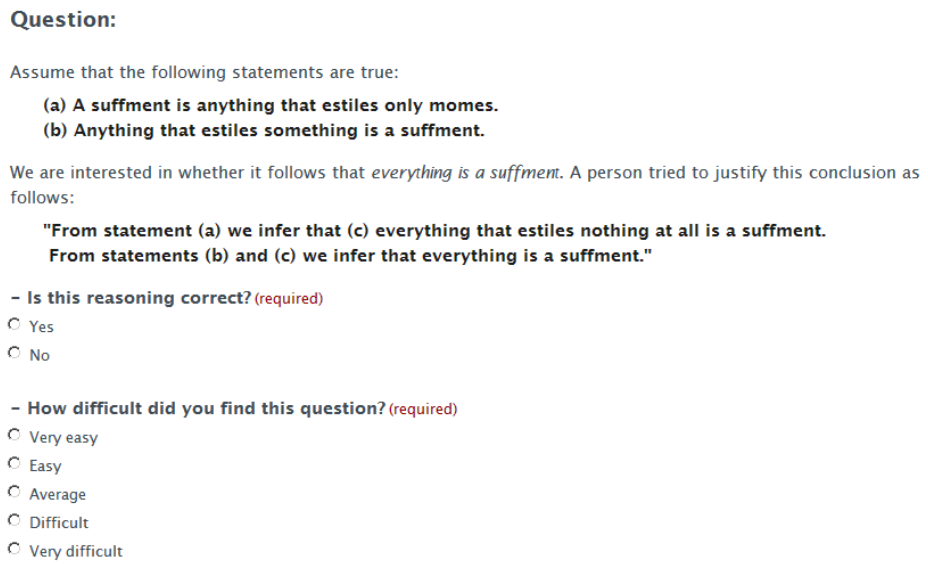

Fig. 2. A test problem in which the FI of the proof tree is 0.03 (0.04 * 0.78)

casually without reading the problem properly). We created two types of control problems: *non-entailment* and *trivial* problems. In a non-entailment problem the test proof tree includes a lemma or a conclusion about an object, a relationship, or both, that are not mentioned in the premises. The correct answer for non-entailment problems is "No", trivially. In order to create such problems, we examined three possibilities for which the entailment is invalid:

1. First inference step is invalid, second inference step is valid
2. First inference step is valid, second inference step is invalid
3. Both inference steps are invalid

Among the three above-mentioned cases, one would expect fewer mistakes for the third case since they had two opportunities to detect a mistake in the reasoning. Therefore, in this study we used either the first or the second case. In both of these cases, we could not introduce unrelated objects into a premise as this violated the assumption of a test problem that all given premises were true; therefore, we only introduced new objects into the lemma in the first case or the entailment in the second case.

A trivial problem was one in which the test proof tree included only obviously correct inferences, so the correct answer was, also trivially, "Yes". Making trivial problems was quite tricky in this study as we could not merely use repetitions of premises, as we did in the previous study [12]. This is because people might get confused about whether a statement explained an entailment if it merely repeated the entailment. Since people usually reason better with individuals than with general statements, we used inferences with individuals in trivial problems.

As mentioned before, there were 15 test problems for which the correct answers were always positive. For balancing, we created 15 additional control problems, five of which having positive answers and the remaining problems having negative answers. This resulted in 20 positive and 10 negative problems—i.e., 67% positive vs. 33% negative.

4.2 Method

The study was conducted on CrowdFlower, a crowdsourcing service that allows customers to upload tasks to be passed to labour channel partners such as Amazon Mechanical Turk[6]. We set up the operation so that tasks were channelled only to Amazon Mechanical Turk, and were restricted to subjects from Australia, the United Kingdom and the United States since we were aiming to recruit as many (self-reported) native speakers of English as possible.

To eliminate responses from 'scammers' (people who respond casually without considering the problem seriously), we used CrowdFlower's quality control service which is based on *gold-standard data*: we provided problems called *gold units* for which the correct answer is specified, allowing CrowdFlower to filter automatically any subjects whose performance on gold units falls below a threshold (75%). In our study, we selected five of our of fifteen control problems as gold units. The management of these gold units was internal to CrowdFlower, and the order for which these gold units would be presented varied randomly on subjects. As in our previous study, the control problems were used only in checking response biases and were not be counted in our main analysis.

It is important to note that in CrowdFlower subjects are not required to complete all problems. They can give up whenever they want, and their responses will be accepted so long as they perform well on gold units. CrowdFlower randomly assigns non-gold problems to subjects until it collects up to a specified number of valid responses for each problem. In our study we specified 80. However, since we were only interested in responses in which all 30 problems were answered, we selected only 59 valid responses.

5 Results

5.1 Control Problems

Figure 3 shows that for the 59 participants, there are 7 who answered fewer than 70% of the control questions correctly, suggesting that they were not performing the test seriously; their results were accordingly discarded. Of the 52 subjects remaining, only one claimed familiarity with OWL, 45 reported no familiarity, and the others did not specify (this question was optional).

[6] `http://crowdflower.com/` and `http://www.mturk.com/`

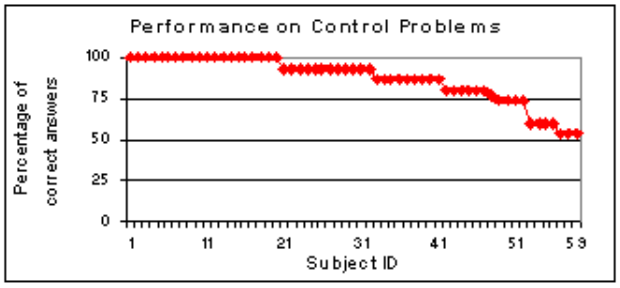

Fig. 3. The subjects' performance on the control problems sorted decreasingly

5.2 Response Bias

Table 3 shows the absolute frequencies of the subjects' responses "Yes" (+Y) and "No" (−Y) for all problems in the study—both control and test. It also subdivides these frequencies according to whether the response was correct (+C) or incorrect (−C). Thus for instance the cell +Y+C counts cases in which subjects answered "Yes" when this was the correct answer, while +Y−C counts cases in which they answered "Yes" when this was incorrect.

Table 3. The distribution of the subjects' responses—"Yes" (+Y) and "No" (−Y)—according to their correctness—"Correct" (+C) and "Incorrect" (−C)

	+Y	**−Y**	**TOTAL**
+C	774	458	1232
−C	59	265	324
TOTAL	833	723	1556

Recall that for 67% of the problems the correct answers were "Yes", and for all the remaining problems they were "No". If subjects had a positive response bias we would expect an overall rate much higher than 67%, but in fact we obtained 833/1556 or 54%, suggesting no positive response bias.

Looking at the distribution of incorrect answers, we can also ask whether subjects erred through being too ready to accept invalid conclusions (+Y−C), or too willing to reject conclusions that were in reality valid (−Y−C). The table shows a clear tendency towards the latter, with 265 responses in −Y−C compared with an expected value of 324*723/1556=151 calculated from the overall frequencies. In other words, subjects were more likely to err by rejecting a valid conclusion than by accepting an invalid one, a finding confirmed statistically by the extremely significant association between response (±Y) and correctness (±C) on a 2×2 chi-square test (χ^2=205.3, df=1, p<0.0001).

5.3 Analysis of Objective Understanding

Figure 4 shows the relationship between the predicted FIs and the proportions of correct answers for tested proof trees. Our analysis indicates a statistically significant relationship between the two values ($r=0.88$, $p<0.0001$) (Pearson's r correlation). For most tested proof trees the predicted FIs are lower than the actual proportions of correct answers. A possible explanation is that all of the control questions in this study are two-step inferences whereas those in the previous study [12] are single-step inferences, and the use of more complex control questions in this study may have caused us to recruit better subjects than those of the previous study. However, for detecting differences in understandability of proof trees, our model works relatively well. Among the 15 tested trees in this study, there are 105 pairs on which difficulty comparisons can be made; of these, 93 comparisons were ordered in difficulty as predicted (i.e., an accuracy of 89%).

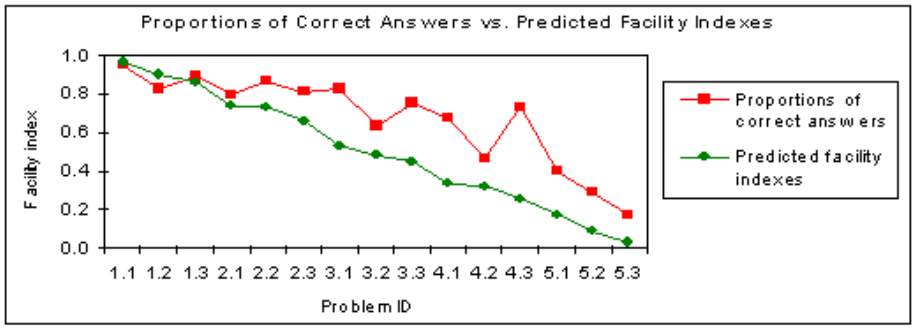

Fig. 4. The predicted FIs vs. the proportions of correct answers

We also tested how well our model can detect differences in understandability of proof trees by analysing the performance of the subjects by bins. For each of the 52 subjects, we counted the number of correct answers for the three questions in each bin, so obtaining a value of 0 to 3 for the associated bin. After that, we applied a Friedman test on the obtained values, which confirmed that there were statistically significant differences in performance between the five bins ($\chi^2=108.95$, df=4, $p<0.0001$). Follow-up pairwise comparisons using a Wilcoxon Signed Ranks test showed that there were statistically significant differences in performance between any bin pair ($p<0.05$) except between bins 2 and 3. (This could be because subjects found questions 3.1 and 3.3 easier than expected, thus reducing the difference between bins 2 and 3.)

It is also clear from Figure 4 that there are exceptional cases for which the subjects performed much better than we expected, such as proof trees 4.3, 4.1, 3.3, and 3.2. The changes of verbalisations used in this study may be the main reason for these exceptions. Proof trees 4.1 and 4.3 are the only two cases which include an *InverseObjectProperties(r1,r0)* axiom. In the previous study [12], we

used the verbalisation "X r0 Y if and only if Y r1 X" to present this axiom in rules 44 and 48 (in Table 4). The FIs we measured for these rules when using this verbalisation are 0.40 and 0.32 respectively. In this study, we used the verbalisation ""X r0 Y" means the same as "Y r1 X"", which is less technical than the former, for testing trees 4.1 and 4.3; this might explain why participants performed better on these trees than we expected. The proportions of correct answers for trees 4.1 and 4.3 are 0.67 and 0.73.

Similarly, proof trees 3.2 and 3.3 are the only two cases which include *SubObjectPropertyOf(r1,r0)* axioms. In our previous study [12], we used the verbalisation "The property r1 is a sub-property of r0" to present this axiom in rules 33 and 37 (in Table 4). The FIs we measured for these rules when using this verbalisation are 0.61 and 0.55. In the present study, we used the less technical verbalisation "If X r1 Y then X r0 Y", which might again explain why performance on these trees was better than expected. The proportions of correct answers for trees 3.2 and 3.3 are 0.63 and 0.75.

5.4 Analysis of Subjective Understanding

Figure 5 plots the predicted FIs for test problems against the mean difficulty ratings (ranging from 1, very difficult, to 5, very easy) reported by subjects. The correlation between FIs and difficulty ratings is high (r=0.85) and significant (p<0.0001) (Pearson's r correlation).

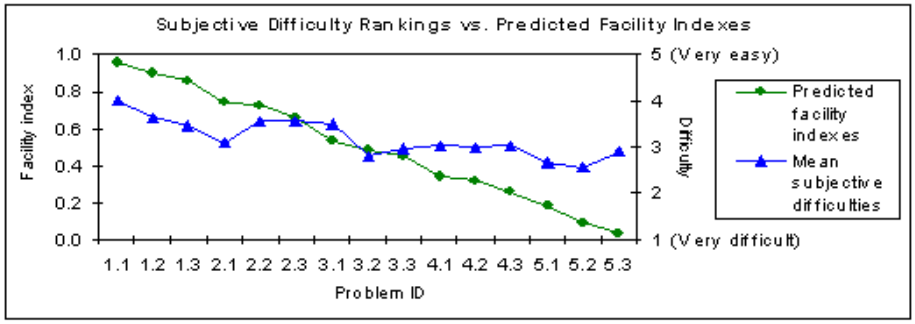

Fig. 5. The predicted FIs vs. the mean subjective difficulty ratings

As in the analysis of objective understanding, we tested how our model can detect differences in understandability of proof trees by analysing difficulty rankings by bins. For each of the 52 subjects, we computed the mean value of difficulty rankings for the three questions of each bin, and so obtained a value of 0 to 5 for the associated bin. After that, we applied a Friedman test on the obtained values, which confirmed that there were statistically significant differences in difficulty ranking between the five bins (χ^2=88.66, df=4, p<0.0001). Follow-up pairwise comparisons using a Wilcoxon Signed Ranks test showed that there

were statistically significant differences in difficulty ranking between any bin pair ($p<0.05$) except between bins 3 and 4, for which the results might have been affected (as explained in section 5.3) by the more accessible verbalisations used in the present study for the proof trees 3.2, 3.3, 4.1, and 4.3. Proof tree 5.3 is an exception as it was ranked as easier than 5.2 while our model predicted the opposite direction. Our prediction was supported by the analysis of objective understanding presented previously. This result suggests a failure in understanding this proof tree—that is, the subjects thought that they had understood the inference correctly but actually they had not.

6 Conclusions and Future Work

This paper describes a method for predicting the understandability of OWL inferences, focussing on people with limited knowledge of OWL. We present a probabilistic model for measuring the understandability of a multiple-step inference based on measurement of the understandability of single-step inferences. First the FIs of 51 single-step inferences were measured in an empirical study resulting in estimates of the probability that a person will understand the inference. Then by multiplying the FIs of individual inference steps, we can compute the joint probability of all steps being followed as the FI of the associated multiple-step inference. We also report an evaluation study which confirms that our model works relatively well for two-step inferences in OWL. This model has been applied in our research on generating accessible explanations for entailments derived from OWL ontologies, to determine the most understandable among alternative inferences from a justification, as well as to sort explanations in order of decreasing understandability when multiple justifications are found.[7]

The proposed model grounds FIs in a well-established probabilistic interpretation. This gives us confidence that the good performance of the model on two-step inferences will extend to n-step inferences for $n>2$. This has, however, to be balanced with the somewhat better performance of the theoretically less well-founded approach of taking the minimum, which for two-step inferences achieves an accuracy of 94%. Further work is needed to compare these models for inferences with more than two steps.

In addition to improving the understandability model, we will aim to make our explanations for absurd entailments more focused; for instance, by tracing from the entailment in Table 1 to a sequence of absurd lemmas, including "Everything is a movie", "Everything that has no rating at all is a movie", and "Everything that has no rating at all is a good movie", and finally reaching the misused axiom "A good movie is anything that has as ratings only four stars". Leaving aside the way the proposed model was used in our work, we believe it can be used by others to predict the understandability of different kinds of inferences, and so is worth reporting as a resource for other researchers.

[7] We have implemented a prototype of this model as a plug-in of the SWAT ontology editing tool, which will be published soon at `http://mcs.open.ac.uk/nlg/SWAT/`.

7 Acknowledgments

This research was undertaken as part of the SWAT (Semantic Web Authoring Tool) project, supported by the UK Engineering and Physical Sciences Research Council (EPSRC grant no. G033579/1). We thank our colleagues and the anonymous viewers.

References

1. OWL 2 Web Ontology Language Document Overview (Second Edition). `http://www.w3.org/TR/owl2-overview/`, Last Accessed: 1st February 2013
2. Baader, F., Peñaloza, R., Suntisrivaraporn, B.: Pinpointing in the Description Logic $\mathcal{EL}^+$. In: German Conference on Advances in Artificial Intelligence (KI 2007). pp. 52–67 (2007)
3. Horridge, M., Bail, S., Parsia, B., Sattler, U.: The Cognitive Complexity of OWL Justifications. In: International Semantic Web Conference (ISWC 2011). pp. 241–256 (2011)
4. Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., Wang, H.: The Manchester OWL Syntax. In: International Workshop on OWL: Experiences and Directions (OWLED 2006) (2006)
5. Horridge, M., Parsia, B., Sattler, U.: Lemmas for Justifications in OWL. In: International Workshop on Description Logics (DL 2009) (2009)
6. Ji, Q., Qi, G., Haase, P.: A Relevance-Directed Algorithm for Finding Justifications of DL Entailments. In: Asian Semantic Web Conference (ASWC 2009). pp. 306–320 (2009)
7. Kalyanpur, A., Parsia, B., Horridge, M., Sirin, E.: Finding All Justifications of OWL DL Entailments. In: International Semantic Web Conference (ISWC 2007) (2007)
8. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.: Debugging Unsatisfiable Classes in OWL Ontologies. Journal of Web Semantics 3(4), 268–293 (2005)
9. Lam, J.S.C., Sleeman, D., Pan, J.Z., Vasconcelos, W.: A Fine-Grained Approach to Resolving Unsatisfiable Ontologies. Journal of Data Semantics pp. 62–95 (2008)
10. Motik, B., Shearer, R., Horrocks, I.: A Hypertableau Calculus for $\mathcal{SHIQ}$. In: International Workshop on Description Logics (DL 2007). pp. 419–426 (2007)
11. Newstead, S.E., Bradon, P., Handley, S.J., Dennis, I., Evans, J.S.B.T.: Predicting the difficulty of complex logical reasoning problems. Thinking & Reasoning 12:1, 62–90 (2006)
12. Nguyen, T.A.T., Power, R., Piwek, P., Williams, S.: Measuring the Understandability of Deduction Rules for OWL. In: International Workshop on Debugging Ontologies and Ontology Mappings (WoDOOM 2012) (2012)
13. Schlobach, S., Cornet, R.: Non-standard Reasoning Services for the Debugging of Description Logic Terminologies. In: International Joint Conference on Artificial Intelligence (IJCAI 2003). pp. 355–360 (2003)
14. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A Practical OWL-DL Reasoner. Journal of Web Semantics 5, 51–53 (2007)
15. Tsarkov, D., Horrocks, I.: FaCT++ Description Logic Reasoner: System Description. In: International Joint Conference on Automated Reasoning (IJCAR 2006). pp. 292–297 (2006)

Table 4. Deduction rules and their facility indexes (FI). For short, the names of OWL functors are abbreviated.

ID	Rule	FI	ID	Rule	FI
1	EqvCla(X,Y...) →SubClaOf(X,Y)	**1.00**	2	SubClaOf(X,ObjIntOf(Y,Z...)) →SubClaOf(X,Y)	**0.96**
3	ObjPropDom(r0,X) ∧ SubClaOf(X,Y) →ObjPropDom(r0,Y)	**0.96**	4	SubClaOf(ObjUniOf(X,Y...),Z) →SubClaOf(X,Z)	**0.96**
5	SubClaOf(X,Y) ∧ SubClaOf(X,Z) →SubClaOf(X,ObjIntOf(Y,Z))	**0.94**	6	SubClaOf(⊤,Y) →SubClaOf(X,Y)	**0.93**
7	SubClaOf(X,ObjSomValF(r0,⊤)) ∧ SubClaOf(X,ObjAllValF(r0,Y)) →SubClaOf(X,ObjSomValF(r0,Y))	**0.90**	8	ObjPropRng(r0,X) ∧ SubClaOf(X,Y) →ObjPropRng(r0,Y)	**0.90**
9	ObjPropDom(r0,Y) ∧ SubClaOf(X,ObjSomValF(r0,Z)) →SubClaOf(X,Y)	**0.86**	10	EqvCla(X,ObjUniOf(Y,Z...)) →SubClaOf(Y,X)	**0.82**
11	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(ObjMinCard(1,r0,Y),Z)) →SubClaOf(X,Z)	**0.82**	12	SubClaOf(X,Y) ∧ SubClaOf(Y,Z) →SubClaOf(X,Z)	**0.80**
13	SubClaOf(X,ObjCompOf(X)) →SubClaOf(X,⊥)	**0.80**	14	SubObjPpOf(r0,r1) ∧ SubObjPpOf(r1,r2) →SubObjPpOf(r0,r2)	**0.79**
15	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,Z) →SubClaOf(X,ObjSomValF(r0,Z))	**0.79**	16	EqvCla(X,ObjIntOf(Y,Z...)) →SubClaOf(X,Y)	**0.79**
17	ObjPropDom(r0,X) ∧ SubClaOf(ObjAllValF(r0,⊥),X) →SubClaOf(⊤,X)	**0.78**	18	ObjPropRng(r0,X) ∧ SymObjProp(r0) →ObjPropDom(r0,X)	**0.77**
19	SubClaOf(Y,X) ∧ SubClaOf(ObjCompOf(Y),X) →SubClaOf(⊤,X)	**0.77**	20	ObjPropDom(r0,⊥) →SubClaOf(⊤,ObjAllValF(r0,⊥))	**0.76**
21	ObjPropRng(r0,⊥) →SubClaOf(⊤,ObjAllValF(r0,⊥))	**0.76**	22	DisCla(X,Y...) ∧ SubClaOf(Z,X) ∧ SubClaOf(W,Y) →DisCla(Z,W)	**0.76**
23	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,ObjSomValF(r0,Z)) ∧ TrnObjProp(r0) →SubClaOf(X,ObjSomValF(r0,Z))	**0.75**	24	SubClaOf(X,ObjUniOf(Y,Z)) ∧ SubClaOf(Y,W) ∧ SubClaOf(Z,W) →SubClaOf(X,W)	**0.73**
25	SubClaOf(ObjCompOf(X),Y) →SubClaOf(⊤,ObjUniOf(X,Y))	**0.72**	26	SubClaOf(X,ObjUniOf(Y,Z)) ∧ SubClaOf(Y,Z) →SubClaOf(X,Z)	**0.71**
27	SubClaOf(ObjSomValF(r0,X),Y) ∧ SubClaOf(ObjAllValF(r0,⊥),Y) →SubClaOf(ObjAllValF(r0,X),Y)	**0.71**	28	ObjPropDom(r0,X) ∧ SymObjProp(r0) →ObjPropRng(r0,X)	**0.69**
29	SubClaOf(X,ObjSomValF(r0,ObjSomValF(r0,Y))) ∧ TrnObjProp(r0) →SubClaOf(X,ObjSomValF(r0,Y))	**0.68**	30	ObjPropRng(r0,Z) ∧ SubClaOf(X,ObjSomValF(r0,Y)) →SubClaOf(X,ObjSomValF(r0,ObjIntOf(Y,Z)))	**0.64**
31	SubClaOf(⊤,Y) ∧ DisCla(X,Y) →SubClaOf(X,⊥)	**0.64**	32	SubClaOf(X,ObjExtCard(n1,r0,Y)) →SubClaOf(X,ObjMinCard(n2,r0,Y)), 0<n2≤n1	**0.63**
33	ObjPropDom(r0,X) ∧ SubObjPpOf(r1,r0) →ObjPropDom(r1,X)	**0.61**	34	SubClaOf(X,Y) ∧ DisCla(X,Y) →SubClaOf(X,⊥)	**0.57**
35	SubClaOf(X,Y) ∧ SubClaOf(X,Z) ∧ DisCla(Y,Z) →SubClaOf(X,⊥)	**0.56**	36	TrnObjProp(r0) ∧ InvObjProp(r0,r1) →TrnObjProp(r1)	**0.55**
37	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubObjPropOf(r0,r1) →SubClaOf(X,ObjSomValF(r1,Y))	**0.55**	38	ObjPropRng(r0,X) ∧ SubObjPropOf(r1,r0) →ObjPropRng(r1,X)	**0.52**
39	SubClaOf(X,Y) ∧ SubClaOf(X,ObjCompOf(Y)) →SubClaOf(X,⊥)	**0.51**	40	SubClaOf(X,ObjSomValF(r0,ObjIntOf(Y,Z...))) ∧ DisCla(Y,Z) →SubClaOf(X,⊥)	**0.50**
41	SubClaOf(X,ObjMinCard(n1,r0,Dor⊤)) ∧ SubClaOf(X,ObjMaxCard(n2,r0,⊤)), 0<n2<n1 →SubClaOf(X,⊥)	**0.48**	42	SubClaOf(X,ObjSomValF(r0,Y)) ∧ SubClaOf(Y,⊥) →SubClaOf(X,⊥)	**0.45**
43	FuncDatProp(d0) ∧ SubClaOf(X,DatMinCard(n,d0,DR0)), n>1 →SubClaOf(X,⊥)	**0.41**	44	ObjPropRng(r0,X) ∧ InvObjProp(r0,r1) →ObjPropDom(r1,X)	**0.40**
45	FuncDatProp(d0) ∧ SubClaOf(X,DatHasVal(d0,l0⋆DT0)) ∧ SubClaOf(X,DatHasVal(d0,l1⋆DT1)) where DT0 and DT1 are disjoint or l0 ≠ l1 →SubClaOf(X,⊥)	**0.40**	46	FuncObjProp(r0) ∧ SubClaOf(X,ObjHasVal(r0,i0)) ∧ SubClaOf(X,ObjHasVal(r0,i1)) ∧ DiffInd(i0,i1...) →SubClaOf(X,⊥)	**0.39**
47	ObjPropDom(r0,X) ∧ InvObjProp(r0,r1) →ObjPropRng(r1,X)	**0.38**	48	SubClaOf(X,ObjAllValF(r0,Y) ∧ InvObjProp(r0,r1) →SubClaOf(ObjSomValF(r1,X),Y)	**0.32**
49	DatPropRng(d0,DR0) ∧ X⊑ObjSomValF(r0,DatHasVal(d0,l0⋆DT1)) where DR0 & DT1 are disjoint →SubClaOf(X,⊥)	**0.19**	50	DatPropRng(d0,DR0) ∧ SubClaOf(X,DatSomeValFrm(d0,DR1)) where DR0 & DR1 are disjoint →SubClaOf(X,⊥)	**0.18**
51	EqvCla(X,ObjAllValF(r0,Y)) →SubClaOf(ObjAllValF(r0,⊥),X)	**0.04**			