# On the Potential and Limitations of Few-Shot In-Context Learning to Generate Metamorphic Specifications for Tax Preparation Software

**Dananjay Srinivas**[*1] **Rohan Das**[*1] **Saeid Tizpaz-Niari**[2]
**Ashutosh Trivedi**[1] **Maria Leonor Pacheco**[1]
[1] University of Colorado Boulder [2]University of Texas El Paso
[1]{first.last}@colorado.edu [2]saeid@utep.edu

## Abstract

Due to the ever-increasing complexity of income tax laws in the United States, the number of US taxpayers filing their taxes using tax preparation software (henceforth, tax software) continues to increase. According to the U.S. Internal Revenue Service (IRS), in FY22, nearly $50\%$ of taxpayers filed their individual income taxes using tax software. Given the legal consequences of incorrectly filing taxes for the taxpayer, ensuring the correctness of tax software is of paramount importance. *Metamorphic testing* has emerged as a leading solution to test and debug legal-critical tax software due to the absence of correctness requirements and trustworthy datasets. The key idea behind metamorphic testing is to express the properties of a system in terms of the relationship between one input and its slightly metamorphosed twinned input. Extracting metamorphic properties from IRS tax publications is a tedious and time-consuming process. As a response, this paper formulates the task of generating metamorphic specifications as a *translation task* between properties extracted from tax documents—expressed in natural language—to a contrastive first-order logic form. We perform a systematic analysis on the potential and limitations of *in-context* learning with *Large Language Models* (LLMs) for this task, and outline a research agenda towards automating the generation of metamorphic specifications for tax preparation software.

## 1 Introduction

Recent surveys estimate that between 40 to 50% of taxpayers in the United States use tax preparation software to file their taxes (Farrington, 2023). Given the pervasive use of these systems and the penalties associated with filing taxes incorrectly (IRS, 2023), making sure that tax preparation software is free of bugs is of paramount importance. However, there are considerable challenges that make the application of standard software testing approaches infeasible in this domain. Among these, the absence of correctness requirements and the lack of publicly available benchmarks (due to obvious privacy concerns) are primary obstacles to automatic testing and debugging of such systems.

To address these challenges, Tizpaz-Niari et al. (2023) introduced a testing framework for U.S. tax preparation software guided by *metamorphic relations*. The authors define metamorphic relations as relations between two similar individuals who differ in key characteristics that put them in different tax income buckets. This way, they are able to evaluate the outcome of an individual taxpayer in comparison with individuals who are deemed similar to them. With the help of tax preparation experts, they were able to manually derive a comprehensive set of critical correctness properties of tax preparation outcomes expressed in first-order logic, and develop a search strategy to automatically generate test cases for an open-source tax preparation software. While their approach was successful, deriving formal metamorphic specifications is a time-consuming process that requires significant domain expertise. Moreover, any changes to the tax code would require continuous efforts to make sure that specifications remain up to date.

In this paper, we set out to explore whether recent advances in Large Language Models (LLMs) (Brown et al., 2020; Touvron et al., 2023a; Taori et al., 2023; Wang et al., 2023) can help reduce the manual effort required to derive metamorphic specifications for this domain. While supervised learning without sizeable training datasets is infeasible, recent findings show that LLMs can perform surprisingly well in few-shot scenarios for a wide variety of language tasks, including traditional NLP tasks like translation and question-answering (Brown et al., 2020), vision-language tasks (Monajatipoor et al., 2023), and socio-linguistic tasks like morality framing (Roy et al., 2022).

[*]Equal Contribution



*Proceedings of the Natural Legal Language Processing Workshop 2023*, pages 230–243

To perform our study, we first define the extraction of metamorphic specifications for tax preparation software as a *language to first-order logic translation* task. To do this, we curate a dataset of 33 high-quality natural language properties derived from tax documents[*] and their corresponding formal logic representation Tizpaz-Niari et al. (2023). As curating these high-quality examples is expensive, we formulate the problem as a few-shot learning task and experiment with various *in-context learning* strategies to obtain a formal representation for each property. In-context learning is a task-adaptation technique that does not update the parameters of the LLM, but rather primes the model response by providing a sequence of demonstrations (Brown et al., 2020). Recent work has shown that in-context learning can lead to better out-of-domain performance than few-shot fine-tuning (Awadalla et al., 2022; Si et al., 2023).

We perform a comprehensive analysis of our results and show that using few-shot in-context learning, the model makes between 1-2 mistakes per property on average when generating predicates, variables and operators. While this is encouraging for a model that has barely seen any in-domain, tasks-specific data, there is a lot of room for improvement before these specifications are usable for automated testing. This work represents a first step towards the automated generation of metamorphic specifications for tax preparation software. In Sec. 5 we outline a research agenda for this purpose.

## 2 Related Work

Converting natural language utterances to logic forms has a long history in the Natural Language Processing community. Semantic parsing is a long-standing NLP task that looks at this problem (Kamath and Das, 2019). Most of the work in this space has been either tailored to going from natural language to linguistically-motivated meaning representations (Palmer et al., 2010; Banarescu et al., 2013), or to executable programs like SQL queries (Sun et al., 2018) and robotic commands (Dukes, 2014). The solutions employed for these domains are not readily applicable to our case. On the one hand, datasets for linguistically motivated meaning representations usually deal with simple general utterances, and their structured forms are very distinct from the structured forms needed for automated software verification. On the other hand, executable domains like SQL have the advantage that there are large-scale, readily available repositories of in-domain data, which is not the case for low-resource, closed domains like tax preparation software.

More recently, there has been increased interest in exploring NLP techniques to derive formal specifications from technical documentation. For example, Pacheco et al. (2022) look at the task of extracting finite-state machines from network protocol RFCs, Zhang et al. (2020) extract LTL correctness specifications from prose policies for IoT apps, and Chen et al. (2019) look at sentences in a developer guide to extend the finite-state machines for a small set of payment services. In all of these cases, authors are either able to leverage off-the-shelf parsers or employ supervised methods over a curated in-domain dataset.

## 3 Task Description

Metamorphic testing is a well-studied strategy in the systems community (Chen et al., 2020). It was developed to circumvent the absence of oracles (i.e., black box modules that decide whether the output of the system is correct for a given input). The main idea behind metamorphic testing is that correctness can be established by contrasting multiple input-output behaviors (Segura et al., 2016). The running example is search-engine validation; while it might be impossible to know the expected items that should be returned for any search query $q$, if we have a more restrictive query $q'$, we know that the number of retrieved items for $q$ should be greater than the number of retrieved items for $q'$. This allows for developing strategies to test whether systems satisfy the metamorphic relations.

To formulate our task, we build on the definition of metamorphic specifications for tax-preparation software proposed by Tizpaz-Niari et al. (2023). We have a fixed set $P$ of metamorphic properties written in natural language, expressing aspects of the U.S. Individual Income Tax Returns that relate to disability, credits, and deductions. For each property $p \in P$, we want to synthesize a specification $s$ in first-order logic that captures the metamorphic property. Fig. 1 shows an example of the input and expected output. In Appendix A, we detail the syntax and semantics of metamorphic specifications used for the tax preparation software, as well as the

[*]These documents include Form 1040 (U.S. Individual Income Tax Return), Publication 596 (Earned Income Tax Credit), Schedule 8812 (Qualifying Children and Other Dependents), and Form 8863 (Education Credits).

| Input: Property | Output: Specification |
|---|---|
| An individual with the married filing jointly ($MFJ$) status with a disabled spouse must receive similar or higher tax benefits compared to a similar individual but without the disabled spouse. | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{s\_blind} \mathbf{y}) \wedge (\mathbf{x}.s\_blind \wedge \neg \mathbf{y}.s\_blind)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |

Table 1: An example of input and expected output. Here, the quantification is over the domain of taxpaying individuals, the relation $\equiv_{pred}$ relates equal individuals except for predicate $pred$, and the function $\mathcal{F}$ characterizes the tax return for an individual.

full list of formal specifications for metamorphic relations used for demonstration and testing.

## 4 Experimental Evaluation

In this section, we evaluate the performance of LLMs on the natural language to metamorphic specifications translations. We first outline the evaluation dimensions that we consider, then we describe our experimental setup and results.

### 4.1 Evaluation Dimensions

In order to investigate LLMs' abilities to convert natural language properties into metamorphic specifications, we evaluate different dimensions to identify how the number of examples that the model gets to observe, the prompting strategies used, the LLM implementation used, as well as the characteristics of the task affect model performance.

**Number of examples for few-shot learning:** We look at the effect that the number of demonstrations has on model performance. Since we are working with an extremely reduced dataset, we limit the maximum number of examples that the model sees to two.

**In-context learning strategies:** We seek to study if decomposed prompting (Khot et al., 2023) is effective in translating natural language into formal metamorphic specifications. To do this, we compare the performance between an end-to-end **(E2E)** prompting strategy and different decomposed strategies. To do this, we break down the task of generating metamorphic relations into two sub-tasks: (1) Generating the relevant logical predicates, and (2) Generating the specification in first-order logic. We experiment with two decomposed strategies; the first one **(Implicit)** allows the model to access the context used to generate the predicates by prompting each task subsequently, while the second one **(Explicit)** decouples the predicate and FOL generation entirely, allowing each module to completely focus on its sub-task. Example prompts for all strategies are provided in App. B.

**LLM implementation:** We evaluate all three learning strategies on GPT-3.5. However, since we do not have access to the GPT-3.5 weights, we are reliant on APIs provided by OpenAI. In the interest of transparency and reproducibility, we also try out the Explicit strategy on Alpaca (Taori et al., 2023), an open-source instruction-tuned model built over LLaMA 2 (Touvron et al., 2023b).

**Example difficulty:** Different properties can be captured by varying numbers of predicates. In our evaluation, we compare the model performance against the number of predicates required to capture a metamorphic property. We stipulate that the difficulty of capturing a metamorphic expression increases as the number of predicates increases.

**Error types:** We qualitatively evaluate the different types of errors that the model makes. We qualify different errors by looking at specific cases where the model generated incorrect predicates, operators, or variables.

### 4.2 Results

**Experimental Setup:** For GPT-3.5, we prompt via the ChatCompletion API provided by OpenAI. We use the default API settings as outlined in their documentation. For Alpaca, we use the fine-tuned weights from the LoRa-adapted (Hu et al., 2021) Alpaca 7B model, and run inference to generate responses for our prompts.

**Performance Metrics and Annotation:** Standard generation metrics like BLEU, ROGUE, ChrF and BLEURT rely either on word and character matching (Papineni et al., 2002; Lin, 2004; Popović, 2015), or semantic embedding similarity (Sellam et al., 2020). Metrics that rely on lexical matching are ill-suited to evaluate the generation of first-order-logic propositions, as the operators, predicate names and variables used can vary significantly while maintaining semantic consistency (e.g., the operation $x \equiv_S y$ and the predicate $EqualExcept(x, y, S)$ express the same relationship). Similarly, embedding-based metrics that have been trained on large textual repositories

do not account for valid substitutions in logic that lack similarity in natural language (e.g., $\mathcal{F}(x)$ and $TaxReturn(x)$ can be used to denote the same predicate). For this reason, we resort to human evaluation and define a quality score ranging from 1 to 5 based on the following rubric:

| Rating | Explanation |
|---|---|
| 5 | The generated predicates or FOL match the ground truth. |
| 4 | The generated identities have the correct semantic sense, but incorrect format. |
| 3 | There is 1 mistake in the set of predicates, variables, operators |
| 2 | There are 2 mistakes in the set of predicates, variables, operators |
| 1 | The generated identities are completely incorrect. |

Table 2: Rubric for Performance Quality

Using this rubric, the first two authors of this paper annotated the generated texts with their qualitative judgments. A total of 33 examples were evaluated (see Appendix A), and average results are reported in Tables 3 and 4.

**Predicate and FOL quality based on prompting style and number of examples:** In Table 3, we look at the effect that examples have on different styles of prompting. We found that on average, **GPT Explicit** decomposed prompting performed best among all strategies when demonstrations were provided. Additionally, we find higher variation in results for **GPT E2E**, suggesting that decomposed prompting might lead to slightly more consistent results. As expected, demonstrating with a higher number of examples helps the model generate better translations. Nevertheless, the average quality for all strategies sits solidly in the middle of the rubric (about 1-2 errors per example), shedding light on the limitations of few-shot in-context learning as-is for this task. **Alpaca Explicit** produces nearly identical results to **GPT Explicit** in the zero-shot scenario. However, performance does not improve when Alpaca is provided with two examples instead of just one, unlike **GPT Explicit**. The Alpaca implementation tested is significantly smaller (7B vs. 175B for GPT 3.5). We hypothesize that Alpaca 7B likely needs access to more context to be able to improve its performance for this task. We leave an in-depth exploration of the capabilities of smaller LLMs for future work.

**Effect of predicate complexity on translation:** In this evaluation, we fix the number of examples the model sees to two. We then evaluate how different numbers of predicates affect the quality of translations. Results can be seen in Tab. 4. For all strategies, there is a clear drop in performance for harder examples. This is consistent with our hypothesis that more complex propositions are more difficult to translate. Similarly to the previous analysis, we find that performance on decomposed prompting performs better than E2E prompting. Given the overhead of manual evaluation, we limited this test to GPT-3.5.

In Appendix C, we provide generation examples for each score in the annotation rubric.

### 4.3 Qualitative Analysis of Errors

In this section, we qualitatively evaluate different errors after qualifying them on the basis of correctness with regard to 1) Predicates; 2) Variables; and 3) Operators. We observe that while the model is somewhat robust in translating natural language into logic, it encounters problems when trying to express identities in metamorphic forms.

**Errors observed in predicates:** The most common error that we observed in the translations was due to a lack of consideration for predicates that are expected in metamorphic templates; such as $EqualExcept(x, y, S)$. where $x$ and $y$ are entities that are similar except through the set of predicates defined in $S$.

**Errors observed in variables:** We see that models often declare variables or omit variables without much structure. Due to the stochastic nature of LLM predictions, it often reuses (or is primed to reuse) the same variables it has declared for different predicates; automatically invalidating the metamorphic translation.

**Errors observed in operators:** Models often imitate the same operators that were provided in the prompts. For instance, when we provided a prompt that used $\geq$; the prompt generated the prediction using the same operator to describe the opposite kind of relationship. Similarly, the model frequently uses commonly occurring pairs of logic notation like ( $\forall$ and $\exists$).

**Summarizing qualitative analysis:** Our qualitative study revealed that while LLMs hold the potential to perform well on metamorphic translation tasks, we also observe simple errors caused due to stochastic dependencies. More training data or post-hoc filtering may benefit this approach from generating erroneous responses.

## 5 Findings, Limitations and Future Work

Our experiments show that while LLMs are able to make some headway in the task of metamorphic translation, more work needs to be done to

| n-shot | GPT E2E | | GPT Implicit | | GPT Explicit | | Alpaca Explicit | |
|---|---|---|---|---|---|---|---|---|
| | Pred. | FOL | Pred. | FOL | Pred. | FOL | Pred. | FOL |
| 0 | 1.77 (0.99) | 1.19 (0.49) | 1.15 (0.46) | 1.03 (0.19) | 1.00 (0.00) | 1.03 (0.19) | 1.00 (0.00) | 1.00 (0.00) |
| 1 | 3.29 (1.38) | 2.25 (1.48) | 3.46 (1.39) | 1.27 (0.60) | 4.03 (1.02) | 2.77 (1.28) | 1.61 (1.09) | 1.74 (2.02) |
| 2 | 3.96 (1.15) | 2.85 (1.70) | 3.65 (1.09) | 1.42 (0.95) | 4.27 (0.87) | 3.16 (1.23) | 1.63 (1.08) | 1.71 (1.16) |

Table 3: Average (S.D.) quality scores for Predicate and FOL generation by in-context learning strategy.

| Strategy | # Predicates $< 4$ | | $4 \leq$ # Predicates $< 6$ | | # Predicates $\geq 6$ | |
|---|---|---|---|---|---|---|
| | Pred. | FOL | Pred. | FOL | Pred. | FOL |
| GPT E2E | 3.83 (1.60) | 3.16 (2.04) | 4.07 (0.95) | 3.31 (1.54) | 3.88 (1.25) | 1.88 (1.46) |
| GPT Implicit | 3.50 (1.22) | 1.83 (1.60) | 3.66 (0.88) | 1.50 (0.79) | 3.75 (1.39) | 1.00 (0.00) |
| GPT Explicit | 4.20 (1.03) | 3.50 (0.97) | 4.58 (0.67) | 3.58 (1.08) | 3.88 (0.83) | 2.13 (1.25) |

Table 4: Average (S.D.) quality scores for Predicate and FOL generation for each in-context learning strategy, grouped by difficulty level in terms of the number of ground truth predicates in the statement.

improve performance in order to build safeguards for tax preparation software. Below, we list three key directions for future work.

**Task-oriented prompt learning:** It is clear that few-shot in-context learning is not enough to fully automate the generation of metamorphic specifications for tax preparation software. Recent work has shown that learning prompts that are tailored specifically to the task at hand can help improve in-context learning performance (Chung et al., 2022; Han et al., 2022; Zhang et al., 2023). Going forward, we would like to explore strategies to dynamically adapt prompting strategies to maximize performance and minimize common mistakes.

**Leveraging additional data sources:** While we do not have access to a comprehensive dataset containing metamorphic specifications, there are several external resources that we could indirectly exploit to design weakly-supervised methods and representation learning strategies for the task at hand. For example, we could exploit unlabeled tax-centric documents, out-of-domain language-to-logic datasets, and LLMs trained over code and other structured representations. There is a body of work on task-adaptive pre-training (Gururangan et al., 2020), domain adaptation (Daumé III, 2007), and code-based LLMs for semantic parsing (Shin and Van Durme, 2022) that we could exploit to either complement or replace vanilla in-context learning.

**Closing the loop:** While we explored the task of generating metamorphic specifications by translating from properties in natural language to contrastive first-order logic forms, this task alone is not sufficient to fully automate the verification of tax preparation software. To close the loop, two tasks remain: (1) Automatically extracting properties from free-form tax documents, and (2) Taking potentially noisy specifications and generating executable tests. While closing the loop is an ambitious task, we hypothesize that these additional steps could help inform the translation process, potentially improving performance. For example, if we could automatically extract more properties - even if noisy- from raw documents, we could easily expand the training examples without additional manual cost. Moreover, connecting the extracted specifications with symbolic, executable modules, could serve as a form of indirect feedback to inform the translation module.

## 6 Acknowledgements

Tizpaz-Niari and Trivedi were partially supported by the NSF under grants CCF-2317206 and CCF-2317207. This work utilized the Alpine high-performance computing resource at the University of Colorado Boulder. Alpine is jointly funded by the University of Colorado Boulder, the University of Colorado Anschutz, Colorado State University, and the NSF (award 2201538).

## References

Anas Awadalla, Mitchell Wortsman, Gabriel Ilharco, Sewon Min, Ian Magnusson, Hannaneh Hajishirzi, and Ludwig Schmidt. 2022. Exploring the landscape of distributional robustness for question answering models. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5971–5987,

Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract Meaning Representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

T. Y. Chen, S. C. Cheung, and S. M. Yiu. 2020. Metamorphic testing: A new approach for generating next test cases.

Yi Chen, Luyi Xing, Yue Qin, Xiaojing Liao, XiaoFeng Wang, Kai Chen, and Wei Zou. 2019. Devils in the guidance: Predicting logic vulnerabilities in payment syndication services through automated documentation analysis. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 747–764, Santa Clara, CA. USENIX Association.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Hal Daumé III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, Prague, Czech Republic. Association for Computational Linguistics.

Kais Dukes. 2014. SemEval-2014 task 6: Supervised semantic parsing of robotic spatial commands. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 45–53, Dublin, Ireland. Association for Computational Linguistics.

Robert Farrington. 2023. How much americans pay to file their taxes. *The College Investor*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Seungju Han, Beomsu Kim, Jin Yong Yoo, Seokjun Seo, Sangbum Kim, Enkhbayar Erdenee, and Buru Chang. 2022. Meet your favorite character: Open-domain chatbot mimicking fictional characters with only a few utterances. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5114–5132, Seattle, United States. Association for Computational Linguistics.

J. Edward Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *ArXiv*, abs/2106.09685.

IRS. 2023. Penalties. *Internal Revenue Services Website*.

Aishwarya Kamath and Rajarshi Das. 2019. A survey on semantic parsing. In *Automated Knowledge Base Construction (AKBC)*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Masoud Monajatipoor, Liunian Harold Li, Mozhdeh Rouhsedaghat, Lin Yang, and Kai-Wei Chang. 2023. MetaVL: Transferring in-context learning ability from language models to vision-language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 495–508, Toronto, Canada. Association for Computational Linguistics.

Maria Leonor Pacheco, Max von Hippel, Ben Weintraub, Dan Goldwasser, and Cristina Nita-Rotaru. 2022. Automated attack synthesis by extracting finite state machines from protocol specification documents. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 51–68. IEEE.

Martha Palmer, Daniel Gildea, and Nianwen Xue. 2010. *Semantic Roles*, pages 1–19. Springer International Publishing, Cham.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Shamik Roy, Nishanth Sridhar Nakshatri, and Dan Goldwasser. 2022. Towards few-shot identification of morality frames using in-context learning. In *Proceedings of the Fifth Workshop on Natural Language Processing and Computational Social Science (NLP+CSS)*, pages 183–196, Abu Dhabi, UAE. Association for Computational Linguistics.

Sergio Segura, Gordon Fraser, Ana B. Sanchez, and Antonio Ruiz-Cortés. 2016. A survey on metamorphic testing. *IEEE Transactions on Software Engineering*, 42(9):805–824.

Thibault Sellam, Dipanjan Das, and Ankur Parikh. 2020. BLEURT: Learning robust metrics for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7881–7892, Online. Association for Computational Linguistics.

Richard Shin and Benjamin Van Durme. 2022. Few-shot semantic parsing with language models trained on code. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5417–5425, Seattle, United States. Association for Computational Linguistics.

Chenglei Si, Zhe Gan, Zhengyuan Yang, Shuohang Wang, Jianfeng Wang, Jordan Lee Boyd-Graber, and Lijuan Wang. 2023. Prompting GPT-3 to be reliable. In *The Eleventh International Conference on Learning Representations*.

Yibo Sun, Duyu Tang, Nan Duan, Jianshu Ji, Guihong Cao, Xiaocheng Feng, Bing Qin, Ting Liu, and Ming Zhou. 2018. Semantic parsing with syntax- and table-aware SQL generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 361–372, Melbourne, Australia. Association for Computational Linguistics.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. `https://github.com/tatsu-lab/stanford_alpaca`.

Saeid Tizpaz-Niari, Verya Monjezi, Morgan Wagner, Shiva Darian, Krystia Reed, and Ashutosh Trivedi. 2023. Metamorphic testing and debugging of tax preparation software. In *45th IEEE/ACM International Conference on Software Engineering: Software Engineering in Society, SEIS@ICSE 2023, Melbourne, Australia, May 14-20, 2023*, pages 138–149. IEEE.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

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

Hengyuan Zhang, Dawei Li, Yanran Li, Chenming Shang, Chufan Shi, and Yong Jiang. 2023. Assisting language learners: Automated trans-lingual definition generation via contrastive prompt learning. In *Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)*, pages 260–274, Toronto, Canada. Association for Computational Linguistics.

Shiyu Zhang, Juan Zhai, Lei Bu, Mingsong Chen, Linzhang Wang, and Xuandong Li. 2020. Automated generation of ltl specifications for smart home iot using natural language. In *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 622–625.

## A Metamorphic Relations.

**Syntax and Semantics.** We review the syntax and semantics of first-order logic for metamorphic specification in the context of tax preparation software. Let $X = \{X_1, X_2, \ldots, X_n\}$ is the set of variables corresponding to various fields about an individual in the tax return form and $\mathcal{F} : \mathcal{D}_1 \times \mathcal{D}_2 \times \cdots \times \mathcal{D}_n \to \mathbb{R}_{\geq 0}$ is the *federal tax return* computed by the software, where $\mathcal{D}_i$ is the domain of variable $X_i$. We write $\mathcal{D}$ for $\mathcal{D}_1 \times \mathcal{D}_2 \times \cdots \times \mathcal{D}_n$. These variables correspond to intuitive labels such as `age` (numerical variable), `blind` (Boolean variable), and `sts` (filing status with values such as MFJ, married filing jointly, and MFS, married filing separately). For an individual $\mathbf{x} \in \mathcal{D}$, we write $\mathbf{x}(i)$ for the value of $i$-th variable, or $\mathbf{x}.\texttt{lab}$ for the value of variable `lab`. Let $\mathcal{L}$ be the set of all labels.

For labels $L \subseteq \mathcal{L}$ and inputs $\mathbf{x} \in \mathcal{D}$ and $\mathbf{y} \in \mathcal{D}$, we say that $\mathbf{y}$ is a metamorphose of $\mathbf{x}$ with the exceptions of labels $L$, and we write $\mathbf{x} \equiv_L \mathbf{y}$ if $\forall \ell \notin L$ we have that $\mathbf{x}.\ell = \mathbf{y}.\ell$. A metamorphic relation is a first-order logic formula with variables in $X$, constants from domains in $\mathcal{D}$, relation $\equiv_L$, comparisons $\{<, \leq, =, \geq, >\}$ over numeric variables, predicate $\neg$ (negation) for Boolean valued labels, real-valued function for federal tax return $\mathcal{F} : \mathcal{D} \to \mathbb{R}$, Boolean connectives $\wedge$, $\vee$, $\neg$, $\implies$, $\iff$, and quantifiers $\exists x.\phi(x)$ and $\forall x.\phi(x)$ with natural interpretations. We assume that the formulas are given in the prenex normal form, i.e. a block of quantifiers followed by a quantifier free formula.

**Selected Specifications.** Following (Tizpaz-Niari et al., 2023), we consider aspects of Individual Income Tax Return that relate to disability, credits, and deductions. Specifically, we focus on on fields related to the standard deductions for senior and disable individuals; the Earned Income Tax Credit (EITC), a refundable tax credits for lower income households; the Child Tax Credit (CTC), a non-refundable credit to reduce the taxes owed based on the number of qualifying children under the age of 17; the Educational Tax Credit (ETC) that helps students with the cost of higher education by lowering their owed taxes or increasing their refund; and the Itemized Deduction (ID), an option for taxpayers with significant tax deductible expenses.

We use scenarios and examples described in the policies above to synthesize metamorphic relations. Table 5 shows 33 metamorphic relations in 6 domains for the tax year 2020. For properties #9 to #12, we assume $MAGI$ (modified adjusted gross income) is equivalent to $AGI$. Next, we provide a brief explanation of some of these properties.

1. A senior (over age of 65) must receive similar or better tax benefits when compared to a person without the seniority who is similar in every other aspect (due to higher standard deductions for seniors).

2. A blind individual must receive similar or better tax benefits when compared to a person without the disability who is similar in every other aspect (due to higher standard deductions for blind individuals).

3. An individual with the married filing jointly ($MFJ$) status with a senior spouse must receive similar or higher tax benefits compared to a similar individual but without the senior spouse.

4. An individual with the married filing jointly ($MFJ$) status with a disabled spouse must receive similar or higher tax benefits compared to a similar individual but without the disabled spouse.

5. An individual who files with the head of household ($HoH$) status should receive similar or higher tax return benefits compared to a similar individual who files with the single status.

6. An individual who files the tax with the qualified widow ($QW$) status should receive similar or higher tax return benefits compared to a similar individual who files the tax with single status.

7. An individual who files the tax with the qualified widow ($QW$) status should receive similar or higher tax return benefits compared to a similar individual who files with the head of household ($HoH$) status.

8. An individual with the married filing separately ($MFS$) status who claims EITC credits should receive the same tax return compared to a similar individual (with the same status) who does not claim EITC credits.

9. An individual with the married filing jointly ($MFJ$) status with $AGI$ over $56,844$ who claims EITC credits should receive the same tax return compared to a similar individual

(with the same status) who does not claim EITC credits.

10. An individual with the married filing jointly ($MFJ$) status with $AGI$ less than or equal $56,844$ who claims EITC credits should receive a higher tax returns compared to a similar individual who has $AGI$ greater than $56,844$.

11. Among two qualified individuals with EITC, one with higher EITC claims receives higher or equal tax return benefits.

12. An individual who has the investment income less than \$3,650, computed as the sum of lines 2a, 2b, 3a, and 7 in Form 1040, should receive a higher tax returns compared to a similar individual with the investment income greater than \$3,650.

13. An individual with single status with $AGI$ less than or equal to $15,820$ who is 25 years old or older should receive a higher tax returns compared to a similar individual who is younger than 25 years old.

14. An individual with head of household status with $AGI$ less than or equal to $15,820$ who is 25 years old or older should receive a higher tax returns compared to a similar individual who is younger than 25 years old.

15. An individual with head of household status with $AGI$ less than or equal to $41,757$ and with one qualified children who is younger than 25 years old should receive a similar return compared to a similar individual who is 25 years or older (the age test is not relevant when having at least one qualified children).

16. An individual with qualifying widow status with $AGI$ less than or equal to $15,820$ who is 25 years old or older without any qualified children should receive a higher tax returns compared to a similar individual who is younger than 25 years old.

17. An individual with Qualifying Widow status with $AGI$ less than or equal to $41,757$ and with one qualified children who is younger than 25 years old should receive a similar return compared to a similar individual who is 25 years or older.

18. A married filing jointly status with $AGI$ less than or equal to $21,710$ who is 25 years old or older without any qualified children should receive a higher tax returns compared to a similar individual who is younger than 25 years old.

19. An individual with Qualifying Widow status with $AGI$ less than or equal to $47,646$ and with one qualified children who is younger than 25 years old should receive a similar return compared to a similar individual who is 25 years or older.

20. An individual who is qualified for EITC credit with no qualified children should receive a similar or lower tax returns compared to a similar individual with one qualified children.

21. An individual who is qualified for EITC credit with one or less qualified children should receive a similar or lower tax returns compared to a similar individual with two qualified children.

22. An individual who is qualified for EITC credit with two or less qualified children should receive a similar or lower tax returns compared to a similar individual with three qualified children.

23. An individual who is qualified for EITC credit with three or less qualified children should receive a similar tax returns compared to a similar individual with more than three qualified children.

24. Among two qualified married filing jointly ($MFJ$) individuals, one with higher child tax credits receives higher or equal tax return benefits.

25. This 4-property requires a comparison between four "similar" individuals since there is a relation between two variables of interests: $AGI$ and the number of qualified children/others to claim a CTC. An individual with more qualified dependents must receive higher or similar tax return benefits than an individual with fewer dependents after adjusting for the effects of income levels on the calculations of both the final return and the amounts of CTC claims. Expressing this property requires holding the income of two individuals

the same per each qualified number of children/others.

26. An individual with the married filing separately ($MFS$) status who claims CTC credits should receive similar tax benefits compared to a similar individual with $MFS$ status who does not claim CTC.

27. An individual with the married filing jointly ($MFJ$) status with $AGI$ over $180k$ who claims ETC should receive similar tax benefits compared to a similar individual who does not claim ETC.

28. An individual with the married filing jointly ($MFJ$) status with $AGI$ below $160k$ who claims ETC received higher or similar tax return benefits compared to a similar individual who does not claim ETC or claims a lower amount of ETC credit.

29. This 4-property requires a comparison between four "similar" individuals as the rule changes for individuals with $AGI$ below 160k and between 160k and 180k. By holding $AGI$ constant between two individuals with $AGI$ below 160k (varying the $ETC$ claims) and two individuals with $AGI$ between 160k and 180k (varying the $ETC$ claims with the same rate), the property requires that individuals with lower income (below $160k$) receive higher or similar tax returns.

30. An individual who files with medical/dental expenses ($MDE$) below 7.5% of their $AGI$ and itemizes their deductions receives the same return as a similar individual with no $MDE$ claims.

31. An individual who files with a standard deduction who claims itemized deduction (Line 12) should receive similar tax returns compared to a similar individual who does not claim itemized deduction.

32. An individual who files with itemized deductions below the standard deductions receive a lower or similar tax return benefits compared to a similar individual who files with the standard deductions.

33. An individual who files with itemized deductions above the standard deductions receive a higher or similar tax return benefits compared to a similar individual who claims standard deductions.

Table 5: Metamorphic properties for five domains in the US tax (2020) policies. $\mathcal{F}$ is federal tax return where negative values mean the individual owns payment to the IRS, $sts$ is filing status, $s\_lab$ is spouse's field $lab$, $MFJ$: married filing jointly, $MFS$ is married filing separately, $S$ is single filing, $HoH$ is head of household filing, $QW$ is qualifying widow filing, $AGI$ is adjusted gross income, $L27$ is line 27 of IRS 1040 for Earned Income Tax Credit (EITC), $QC$ is the number of qualified children, $OD$ is the number of other dependents, $CTC$ is child tax credits, $L19$ is line 19 of IRS 1040 for Child Tax Credit ($CTC$), $L29$ is line 29 of IRS 1040 for Education Tax Credit ($ETC$), $MDE$ is medical/dental expenses reported in line 1 of schedule $A$, $iz$ is to use itemized deductions ($ID$) vs. standard deductions, and $L12$ is total itemized deductions ($ID$) from schedule $A$.

| Id | Domain | Metamorphic Property |
|---|---|---|
| 1 | Disability | $\forall \mathbf{x}, \mathbf{y}((\mathbf{x} \equiv_{age} \mathbf{y}) \wedge (\mathbf{x}.age \geq 65) \wedge (\mathbf{y}.age < 65)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 2 | Disability | $\forall \mathbf{x}, \mathbf{y}((\mathbf{x} \equiv_{blind} \mathbf{y}) \wedge (\mathbf{x}.blind \wedge \neg \mathbf{y}.blind)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 3 | Disability | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{s\_age} \mathbf{y}) \wedge (\mathbf{x}.s\_age \geq 65) \wedge (\mathbf{y}.s\_age < 65)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 4 | Disability | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{s\_blind} \mathbf{y}) \wedge (\mathbf{x}.s\_blind \wedge \neg \mathbf{y}.s\_blind)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 5 | Status | $\forall \mathbf{x}, \mathbf{y}((\mathbf{x} \equiv_{sts} \mathbf{y}) \wedge (\mathbf{x}.sts = HoH) \wedge (\mathbf{y}.sts = S)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 6 | Status | $\forall \mathbf{x}, \mathbf{y}((\mathbf{x} \equiv_{sts} \mathbf{y}) \wedge (\mathbf{x}.sts = QW) \wedge (\mathbf{y}.sts = S)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 7 | Status | $\forall \mathbf{x}, \mathbf{y}((\mathbf{x} \equiv_{sts} \mathbf{y}) \wedge (\mathbf{x}.sts = QW) \wedge (\mathbf{y}.sts = HoH)) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 8 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = MFS) \implies \forall \mathbf{y}(\mathbf{x} \equiv_{L27} \mathbf{y} \wedge \mathbf{x}.L27 > 0.0 \wedge \mathbf{y}.L27 = 0.0) \implies \mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})$ |
| 9 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \wedge (\mathbf{x}.AGI > 56,844) \implies \forall \mathbf{y}(\mathbf{x} \equiv_{L27} \mathbf{y} \wedge \mathbf{x}.L27 > 0.0 \wedge \mathbf{y}.L27 = 0.0) \implies \mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})$ |
| 10 | EITC | $\forall \mathbf{x}((\mathbf{x}.sts = MFJ) \wedge (\mathbf{x}.L27 > 0.0)) \implies \forall \mathbf{y}(\mathbf{x} \equiv_{AGI} \mathbf{y} \wedge \mathbf{x}.AGI \leq 56,844 \wedge \mathbf{y}.AGI > 56,844) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 11 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \wedge (\mathbf{x}.AGI \leq 56,844) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{L27} \mathbf{y}) \wedge \mathbf{x}.L27 \geq \mathbf{y}.L27) \implies \mathcal{F}(\mathbf{x}) \geq \mathcal{F}(\mathbf{y})$ |
| 12 | EITC | $\forall \mathbf{x}(\mathbf{x}.L27 > 0.0) \forall \mathbf{y}(\mathbf{x} \equiv_{\{L2a,L2b,3b,7\}} \mathbf{y} \wedge \mathbf{x}.L2a + \mathbf{x}.L2b + \mathbf{x}.L3b + \mathbf{x}.L7 > 3,650) \wedge \mathbf{y}.L2a + \mathbf{y}.L2b + \mathbf{y}.L3b + \mathbf{y}.L7 \leq 3,650) \implies \mathcal{F}(\mathbf{x}) \leq \mathcal{F}(\mathbf{y})$ |
| 13 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = S) \wedge (\mathbf{x}.AGI \leq 15,820) \wedge (\mathbf{x}.L27 > 0.0) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) < \mathcal{F}(\mathbf{y})$ |
| 14 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = HoH) \wedge (\mathbf{x}.AGI \leq 15,820) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 0) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) < \mathcal{F}(\mathbf{y})$ |
| 15 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = HoH) \wedge (\mathbf{x}.AGI \leq 41,756) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 1) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})$ |
| 16 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = QW) \wedge (\mathbf{x}.AGI \leq 15,820) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 0) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) < \mathcal{F}(\mathbf{y})$ |
| 17 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = QW) \wedge (\mathbf{x}.AGI \leq 41,756) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 1) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})$ |
| 18 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \wedge (\mathbf{x}.AGI \leq 21,710) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 0) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) < \mathcal{F}(\mathbf{y})$ |
| 19 | EITC | $\forall \mathbf{x}(\mathbf{x}.sts = MFJ) \wedge (\mathbf{x}.AGI \leq 47,646) \wedge (\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 1) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{Age} \mathbf{y}) \wedge (\mathbf{x}.Age < 25) \wedge (\mathbf{y}.Age \geq 25)) \implies \mathcal{F}(\mathbf{x}) = \mathcal{F}(\mathbf{y})$ |
| 20 | EITC | $\forall \mathbf{x}(\mathbf{x}.L27 > 0.0) \wedge (\mathbf{x}.QC = 0) \implies \forall \mathbf{y}((\mathbf{x} \equiv_{QC} \mathbf{y}) \wedge (\mathbf{y}.QC = 1) \implies \mathcal{F}(\mathbf{x}) \leq \mathcal{F}(\mathbf{y})$ |

Continuation of Previous Table.

| Id | Domain | Metamorphic Property |
|---|---|---|
| 21 | EITC | $\forall \mathbf{x}(\mathbf{x}.L27{>}0.0){\wedge}(\mathbf{x}.QC{\leq}1) \implies \forall \mathbf{y}((\mathbf{x}{\equiv_{QC}}\mathbf{y}){\wedge}(\mathbf{y}.QC{=}2) \implies \mathcal{F}(\mathbf{x}){\leq}\mathcal{F}(\mathbf{y})$ |
| 22 | EITC | $\forall \mathbf{x}(\mathbf{x}.L27{>}0.0){\wedge}(\mathbf{x}.QC{\leq}2) \implies \forall \mathbf{y}((\mathbf{x}{\equiv_{QC}}\mathbf{y}){\wedge}(\mathbf{y}.QC{=}3) \implies \mathcal{F}(\mathbf{x}){\leq}\mathcal{F}(\mathbf{y})$ |
| 23 | EITC | $\forall \mathbf{x}(\mathbf{x}.L27{>}0.0){\wedge}(\mathbf{x}.QC{=}3) \implies \forall \mathbf{y}((\mathbf{x}{\equiv_{QC}}\mathbf{y}){\wedge}(\mathbf{y}.QC{>}3) \implies \mathcal{F}(\mathbf{x}){=}\mathcal{F}(\mathbf{y})$ |
| 24 | CTC | $\forall \mathbf{x}(\mathbf{x}.sts{=}MFS){\wedge}(\mathbf{x}.AGI{\leq}200k) \implies$ $\forall \mathbf{y}((\mathbf{x}{\equiv_{L19}}\mathbf{y}){\wedge}(\mathbf{x}.L19{\geq}\mathbf{y}.L19)) \implies \mathcal{F}(\mathbf{x}){\geq}\mathcal{F}(\mathbf{y}))$ |
| 25 | CTC | $\forall \mathbf{x},\mathbf{x}'(\mathbf{x}.sts{=}\mathbf{x}'.sts{=}MFJ){\wedge}(\mathbf{x}.AGI{<}400k){\wedge}(\mathbf{x}'.AGI{\geq}400k){\wedge}\lceil \mathbf{x}'.AGI{-}400k\rceil_{1k}{*}$ $0.05{<}\mathbf{x}'.QC*2k+\mathbf{x}.OD*0.5k \implies \forall \mathbf{y},\mathbf{y}'(\mathbf{x}{\equiv_{\{QC,OD\}}}\mathbf{y}){\wedge}(\mathbf{x}'{\equiv_{\{QC,OD\}}}\mathbf{y}') \wedge$ $(0{\leq}\mathbf{y}.QC{=}\mathbf{y}'.QC{\leq}\mathbf{x}.QC{=}\mathbf{x}'.QC \leq 10) \wedge (0{\leq}\mathbf{y}.OD{=}\mathbf{y}'.OD \leq \mathbf{x}.OD{=}\mathbf{x}'.OD \leq$ $10) \implies (\mathcal{F}(\mathbf{x})-\mathcal{F}(y)) \geq (\mathcal{F}(x')-\mathcal{F}(y'))$ |
| 26 | ETC | $\forall \mathbf{x}(\mathbf{x}.sts = MFS) \implies \forall \mathbf{y}(\mathbf{x}{\equiv_{L29}}\mathbf{y} \wedge \mathbf{x}.L29 > 0.0 \wedge \mathbf{y}.L29 = 0.0) \implies$ $\mathcal{F}(\mathbf{x}){=}\mathcal{F}(\mathbf{y})$ |
| 27 | ETC | $\forall \mathbf{x}(\mathbf{x}.sts{=}MFJ) \wedge (\mathbf{x}.AGI{\geq}180k) \implies \forall \mathbf{y}(\mathbf{x}{\equiv_{L29}}\mathbf{y} \wedge \mathbf{x}.L29{>}0.0 \wedge$ $\mathbf{y}.L29{=}0.0) \implies \mathcal{F}(\mathbf{x}){=}\mathcal{F}(\mathbf{y})$ |
| 28 | ETC | $\forall \mathbf{x}(\mathbf{x}.sts{=}MFJ) \wedge (\mathbf{x}.AGI{\leq}160k) \implies \forall \mathbf{y}(\mathbf{x}{\equiv_{L29}}\mathbf{y} \wedge$ $\mathbf{x}.L29{\geq}\mathbf{y}.L29) \implies \mathcal{F}(\mathbf{x}){\geq}\mathcal{F}(\mathbf{y})$ |
| 29 | ETC | $\forall \mathbf{x},\mathbf{x}'(\mathbf{x}.sts{=}\mathbf{x}'.sts{=}MFJ) \wedge (\mathbf{x}.AGI{\leq}160k) \wedge (160k{<}\mathbf{x}'.AGI{<}180k) \implies$ $\forall \mathbf{y},\mathbf{y}'((\mathbf{x} \equiv_{L29} \mathbf{y}) \wedge (\mathbf{x}' \equiv_{L29} \mathbf{y}') \wedge (\mathbf{x}.L29 = \mathbf{x}'.L29 \geq \mathbf{y}.L29 =$ $\mathbf{y}'.L29)) \implies (\mathcal{F}(\mathbf{x})-\mathcal{F}(y)) \geq (\mathcal{F}(x')-\mathcal{F}(y'))$ |
| 30 | ID | $\forall \mathbf{x},\mathbf{y}(\mathbf{x}{\equiv_{MDE}}\mathbf{y}) \wedge (\mathbf{x}.MDE{\leq}\mathbf{x}.AGI*7.5\%) \wedge (\mathbf{y}.MDE{=}0.0) \implies \mathcal{F}(\mathbf{x}){=}\mathcal{F}(\mathbf{y})$ |
| 31 | ID | $\forall \mathbf{x}(\neg\mathbf{x}.iz) \implies \forall \mathbf{y}(\mathbf{x}{\equiv_{MDE}}\mathbf{y} \wedge \mathbf{x}.MDE{>}0.0 \wedge \mathbf{y}.MDE{=}0.0) \implies \mathcal{F}(\mathbf{x}){=}\mathcal{F}(\mathbf{y})$ |
| 32 | ID | $\forall \mathbf{x}(\mathbf{x}.sts{=}MFJ) \implies$ $\forall \mathbf{y}((\mathbf{x}{\equiv_{iz,L12}}\mathbf{y}){\wedge}(\mathbf{x}.iz{\wedge}\neg\mathbf{y}.iz){\wedge}(\mathbf{x}.L12{\leq}24.8k{\wedge}\mathbf{y}.L12{=}0.0)) \implies \mathcal{F}(\mathbf{x}){\leq}\mathcal{F}(\mathbf{y})$ |
| 33 | ID | $\forall \mathbf{x}(\mathbf{x}.sts{=}MFJ) \implies$ $\forall \mathbf{y}((\mathbf{x}{\equiv_{iz,L12}}\mathbf{y}){\wedge}(\mathbf{x}.iz{\wedge}\neg\mathbf{y}.iz){\wedge}(\mathbf{x}.L12{>}24.8k{\wedge}\mathbf{y}.L12{=}0.0)) \implies \mathcal{F}(\mathbf{x}){\geq}\mathcal{F}(\mathbf{y})$ |

## B Prompt Examples

Figure 1: Example of an End2End prompt, that looks at generating predicates and FOL in the same prompt. This is an example for nshot = 1; the example was omitted and another added for 0-shot and 2-shot respectively.

```
For the following statement.

"A senior (over age of 65) must receive similar or better tax benefits when compared to a person without the
seniority who is similar in every other aspect (due to higher standard deductions for seniors)."

In this case, the first order predicates are:

Senior(x) : x is an individual who is senior.

Tax(t, x): t is the tax paid by individual x

EqualExcept(x, z, S): where x and z are equal individuals except differing only by the predicates in set S.

Using these predicates, you can generate the following rules in first order logic.

∀x∀y(Senior(x)  ∧ EqualExcept(x,y,{Senior}) → Tax(t,x) ≤ Tax(t,y)

In the following statement

"An individual with the married filing jointly (MFJ) status with a senior spouse must receive  similar or
higher tax benefits compared to a similar individual but without the senior spouse."

In this statement, what would the predicates look like?

Generation Format: you SHOULD ALWAYS generate your answer in the following format

{your generated predicates}

{your generated first order logic statement}

DO NOT generate any other text.
```

Figure 2: Example of a prompt to generate predicates. The prompt is the same for explicit and implicit setups. This is an example for nshot = 1; the example was omitted and another added for 0-shot and 2-shot respectively.

```
For the following statement.

"A senior (over age of 65) must receive similar or better tax benefits when compared to a
person without the seniority who is similar in every other aspect (due to higher standard
deductions for seniors)."

The first order predicates for this statement would look like:
Senior(x) : x is an individual who is senior.
Tax(t, x): t is the tax paid by individual x
EqualExcept(x, z, S): where x and z are equal individuals except differing only by the
predicates in set S.

For the following statement.
"An individual with the married filing jointly (MFJ) status with a senior spouse must receive
similar or higher tax benefits compared to a similar individual but without the senior
spouse."

In this statement, what would the predicates look like?
Generation Format: you SHOULD ALWAYS generate the predicates in the following format
{your generated predicates}
DO NOT generate any other text.
```

Figure 3: Example of a prompt to generate FOL, given that predicates were already generated. This setup is only used in explicit cases (as predicates are provided through context in implicit cases). This is an example for nshot = 1; the example was omitted and another added for 0-shot and 2-shot respectively.

```
Take a look at the following statement.
"A senior (over age of 65) must receive similar or better tax benefits when compared to a person without the seniority who is
similar in every other aspect (due to higher standard deductions for seniors)."

The first order predicates for this statement would look like:
Senior(x) : x is an individual who is senior.
Tax(t, x): t is the tax paid by individual x
EqualExcept(x, z, S): where x and z are equal individuals except differing only by the predicates in set S.

These predicates can be used to generate the following first order logic.
∀x∀y(Senior(x)  ∧ EqualExcept(x,y,{Senior}) → Tax(t,x) ≤ Tax(t,y)
For the following statement.

"An individual with the married filing jointly (MFJ) status with a senior spouse must receive similar or higher tax benefits
compared to a similar individual but without the senior spouse."

The first order predicates for this statement would look like:
Senior(x): x is an individual who is senior.
MarriedFilingJointly(x): x has the married filing jointly status.
Tax(t, x): t is the tax paid by individual x.
Spouse(x, y): x is the spouse of y.
EqualExcept(x, z, S): where x and z are equal individuals except differing only by the predicates in set S.
Using these predicates, represent the statement in first order logic.
Generation Format: you SHOULD ALWAYS generate the first order logic in the following format
{your generated first order logic statement}
DO NOT generate any other text.
```

## C Example Generations for Annotation Rubric

| Metamorphic Statement | Generated Predicates | Generated First Order Logic | Explanation | Score |
|---|---|---|---|---|
| An individual with the married filing jointly (MFJ) status with AGI over $56,844 who claims EITC credits should receive the same tax return compared to a similar individual (with the same status) who does not claim EITC credits. | MFJ(x): x is an individual with the married filing jointly status.<br>AGI(y, x): y is the adjusted gross income of individual x.<br>ClaimEITC(x): x claims EITC credits.<br>TaxReturn(t, x): t is the tax return of individual x.<br>EqualExcept(x, z, S): where x and z are equal individuals except differing only by the predicates in set S. | $\forall x \forall y (MFJ(x) \wedge AGI(y,x) > 56844 \wedge ClaimEITC(x) \wedge EqualExcept(x,y,ClaimEITC) \implies TaxReturn(t,x) = TaxReturn(t,y)$ | The model correctly predicts that AGI criteria is applicable to both individuals in question, and only differs on EITC status. | 5 |
| An individual with single status with AGI less than or equal to $15,820 who is 25 years old or older should receive a higher tax returns compared to a similar individual who is younger than 25 years old. | Single(x): x is an individual with single status.<br>AGI(x, a): x has adjusted gross income of a.<br>Younger(x, y): x is younger than y years old.<br>TaxReturns(r, x): x receives r in tax returns. | $\forall x \forall y (Single(x) \wedge AGI(x, a) \leq 15820 \wedge Older(x, 25) \wedge Younger(y, 25) \wedge EqualExcept(x,y,AGI) \implies TaxReturns(r, x) > TaxReturns(r, y)$ | All the predicates have been generated correctly. However, the predicate "EqualExcept" should only apply to age of the individual and not the AGI. This makes some semantic sense, but is incorrect (not usable). | 4 |
| An individual who files the tax with the qualified widow (QW) status should receive similar or higher tax return benefits compared to a similar individual who files the tax with single status. | QW(x): x is an individual who files the tax with the qualified widow status.<br>Single(x): x is an individual who files the tax with single status.<br>TaxReturn(t, x): t is the tax return received by individual x.<br>SimilarOrHigher(x, y): individual x receives similar or higher tax return benefits compared to individual y. | $\forall x \forall y (QW(x) \wedge EqualExcept(x,y,QW) \implies SimilarOrHigher(x,y)$ | The system makes a mistake in predicates - it does not use the "Single" status to make a judgement on tax return status. | 3 |
| An individual with the married filing jointly (MFJ) status with a disabled spouse must receive similar or higher tax benefits compared to a similar individual but without the disabled spouse. | MFJ(x): x is an individual with the married filing jointly status.<br>Disabled(y): y is an individual who is disabled.<br>Tax(t, x): t is the tax paid by individual x.<br>SimilarOrHigher(x, z, S): where x and z are similar individuals except for the predicates in set S. | $\forall x \forall y (MFJ(x) \wedge Disabled(y) \wedge SimilarOrHigher(x, y, Disabled) \implies Tax(t, x) \leq Tax(t, y)$ | Here, the system misses the predicate "Spouse(x)". In addition to this, the greater than operator is flipped (or there is a mistake in the "Tax(t, x)" predicate as well). | 2 |
| An individual with the married filing jointly (MFJ) status with a senior spouse must receive similar or higher tax benefits compared to a similar individual but without the senior spouse. | MFJ(x): x is an individual with the married filing jointly status.<br>SeniorSpouse(y): y is a senior spouse of an individual.<br>Tax(t, x): t is the tax paid by individual x.<br>EqualExcept(x, z, S): where x and z are equal individuals except differing only by the predicates in set S. | $\forall x \forall y (MFJ(x) \wedge SeniorSpouse(y) \wedge EqualExcept(x, y, MFJ, SeniorSpouse) \implies Tax(t, x) \leq Tax(t, y)$ | Here, the SeniorSpouse is incorrectly applied to individual y. Additionally, the predicate EqualExcept takes more than one predicate (and is redundant). Lastly, the $\leq$ is flipped (or the "Tax" predicate is incorrect). | 1 |

Table 6: In this table, we show examples of our qualitative score rubric. A rating of 5 usually means that the generated logic can be used for metamorphic testing. All these examples were sourced from 2-shot examples in the explicit setup.