Evaluating Synthetic Sentence Coherence Using a Large Language Model

Richard Thompson, Angelos Toutsios, Adam Pease, Mathias Kölsch, Christian Darken
Naval Postgraduate School
1 University Circle
Monterey, CA 93943-5006

Abstract

Fine-tuning a Large Language Model (LLM) to translate imprecise, ambiguous natural language into a formal logic language that supports automated reasoning requires a significant amount of training data. With the assistance of a large ontology, millions of synthetic sentences can be generated in natural language with a corresponding formal representation. A problem arises in that generated sentences are often nonsensical. Detecting and omitting incoherent sentences improves the quality of the training dataset, and provides useful feedback to the ontologist for adding "common sense" rules to the ontology. Using approximately 6,000 human-labeled sentences, this research analyzes three methods for detecting linguistic coherence and conducting high-precision filtering. The first method makes use of expected next-token statistics from an LLM. The second method submits a prompt to an LLM asking it to make a coherence determination. The third method is a composite of the first two. Our results have dramatically improved synthetic training data quality and are expected to contribute to significantly better language reasoning skills.

Introduction

The notable improvement of LLM technology in recent years has created new opportunities for Natural Language Processing (NLP). Informal natural language prompts have proven successful at generating formal programming instructions in Java, Python, and numerous other formal languages. Similarly, automated translation of natural language into a formal logic representation has shown promise. Fig. 1 shows an example English sentence with its logical equivalent. The ultimate goal of our research is to automate the translation of natural language sentence into formal language and then perform automated reasoning using knowledge previously formalized in a large ontology. For an LLM to correctly translate and associate an English word with its correct formal representation in an ontology requires a significant amount of training data. Manually generating millions of unique sentences with their logical equivalent would be an enormously expensive and time consuming initiative.

Our method is to synthetically generate sentences. We select formal terms from a large ontology, along with the English words they have been mapped to (Niles and Pease 2003). SUMO[1] is a large-scale formal ontology with approximately 20,000 concepts and 100,000 human-authored higher-order logic statements (Niles and Pease 2001; Pease 2011), encompassing numerous specialized domains (Pease and Benzmüller 2010). It includes over 6,000 kinds of physical objects, 1,425 process types, hundreds of social roles, and over a thousand relationships, each axiomatically defined in first- and higher-order logic. The Knowledge Interchange Format for the Standard Upper Ontology (SUO-KIF) is a logic language with the purpose of formally representing ontological knowledge in machine-processable form (Pease 2021). We use SUMO terms to build formal SUO-KIF logic statements and their corresponding English equivalents. Words are placed into sentence templates (or frames), and their formal counterparts are placed into a logic formula frame. The simplest version of this is shown in Fig. 2 for illustration. Because the frame structure has well-defined semantics, we can simultaneously generate English sentences and their equivalent, accurate logic formulas.

```
John kicks the ball.

(exists (?J ?K ?B)
   (and
      (instance ?J Human)
      (names "John" ?J)
      (instance ?K Kicking)
      (instance ?B Ball)
      (agent ?K ?J)
      (patient ?K ?B)))
```

Figure 1: English sentence and its equivalent logic formula.

However, unconstrained generation produces many sentences that are linguistically incoherent. Training datasets with higher linguistically semantic coherence has been shown to improve LLM performance (Roberts et al. 2019). Manual filtering is infeasible at scale, motivating the need for automated coherence discriminators. We study the prob-

[1] www.ontologyportal.org

lem of *high-precision filtering* of synthetically generated sentences. Given a set of candidate sentences, the goal is to automatically select a subset that is highly likely to be coherent. Since generating synthetic sentences from the large ontology is relatively low compute, we tolerate a high false negative rate with low recall in order to preserve the precision of the final dataset. Thus, precision and incoherent leakage are more relevant than traditional balanced classification metrics. We are primarily interested in maximizing the number of coherent sentences retained, subject to a strict constraint on incoherent leakage.

Contributions of this work include an analysis of three LLM-based methods for accomplishing *high-precision filtering*. The first method analyzes LLM expected word statistics with three classifiers, Logistic Regression (LR), Random Forest (RF), and Support Vector Machine (SVM). The second method involves carefully engineered LLM prompts. The third method, which we call the cascade method, is a composite of the first two methods, using the computationally cheaper LR discriminator as a pre-filter before filtering further with the more expensive LLM prompt method. A sentence diversity analysis ensures that these methods are not merely retaining trivial sentences.

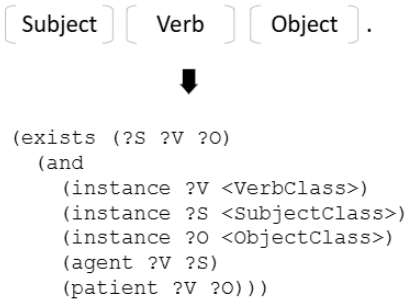

Figure 2: One of many sentence and logic frame structures employed to generate synthetic sentence and logic pair. Image adapted from (Thompson et al. 2025).

The quality of synthetic training data saw dramatic improvements, which is anticipated to yield considerably stronger, more accurate language reasoning abilities. To our knowledge, this is the first work to exploit LLMs for automated filtering to curate a dataset of synthetic English sentence/logic pairs created from a large higher-order ontology.

Related Work

There is a significant body of work in natural language to logic translation. Foundational studies began with controlled English and narrow domains. These early studies include ACE (Fuchs and Schwitter 1996) and CELT (Pease and Murray 2003). These early projects did not leverage Large Language Models, and only CELT employed a large general ontology. LogAnswer was an early attempt at a question and answer system (Furbach et al. 2008) with the help of a large semantic network. It used large portions of MultiNet semantic network (Helbig 2005) and was able to answer basic queries. Other more recent efforts have used LLMs to turn natural language into formal logic without an ontology (Zheng et al. 2025) (Sutcliffe 2026). Frame structures have long been used to generate English/Logic pairs from ontologies (Karp, Peter D 1992), but not on the scale of our work (millions of sentences). Recently, researchers have developed a system to translate natural language into First Order Logic using an LLM (Tammet et al. 2024), including deploying LLMs to assist in generating a synthetic training dataset with thirty-four thousand sentence/logic pairs (Yang et al. 2023), with limited linguistic and logical complexity. Like many statistical AI systems this project provides natural sounding responses, but also combines a limited ontology of a few dozen labels to add some meaning and reasoning behind certain responses. Without definitions to anchor the meaning of terms used in logic expressions, symbols have meaning only to the extent that they exist in the current dialogue. This risks semantic mismatch because the formalizations do not conform to the semantics that people would assign to those labels. For example, "The party is on Thursday." and "The book is on the table." both feature the preposition "on" but with a completely different meaning. Without an ontology to anchor the meaning of the logical symbols, contradictions can easily be generated. Products labeled as ontologies vary widely in complexity, but most are not written in expressive logics. Other upper ontologies are quite small, with a few dozen to a hundred concepts. These include BFO (Otte, Beverley, and Ruttenberg 2022) and DOLCE (Gangemi et al. 2002).

Methodology

Approximately 6,000 sentences were generated as previously described. Human classifiers were instructed to read each sentence and annotate as *coherent* or *incoherent*. Across the dataset, 47.2% of sentences were marked coherent. The dataset is randomly shuffled and split into a training set (80%) and also a test set (20%) for final comparison. Five discriminators were trained as described below.

Method 1: Expected-Token Discriminators

Overview. Our first method leverages token-level statistics from pretrained language model. For each sentence, we first compute expected next-token statistics. Then, three models were generated based on these statistics: Logistic Regression (LR), Random Forest (RF), and Support Vector Machine (SVM). Each model outputs a real-valued coherence score. LR and calibrated SVM/RF scores are interpreted as probability estimates and thresholded for accept/reject decisions. For classification, each model's probability threshold was tuned to maximize precision while meeting statistical levels of confidence.

Feature Extraction Process. Features were extracted from each synthetic sentence using a teacher-forced token probability approach, where each token position i receives as input all preceding tokens and produces logits over the vocabulary, implemented in Python with the HuggingFace Transformers library. The LLaMA 3.2 model (meta-llama/Llama-3.2-3B) served as the base language model for

computing token-level statistics. Llama 3.2 was chosen because it is capable, but small enough to more quickly process numerous prompts. Each sentence was first tokenized using the LLaMA 3.2 tokenizer. Then the tokenized sequence was passed through the model in teacher-forcing mode. Token-level log probabilities $\log p(t_i|t_{<i})$ were extracted by applying log-softmax to the output logits and gathering the probability assigned to each observed token. Per-token surprisal values were computed as $s_i = -\log p(t_i|t_{<i})$, measured in nats (natural logarithm) (Wilcox et al. 2023). Finally, from these surprisal values and the token sequence, sentence features were calculated.

The extraction pipeline processed sentences sequentially, computing the following features: `n_tokens` (total number of tokens in the sentence), `avg_nll` (overall surprisal), `p95` and `p99` (95th and 99th percentile of per-token surprisal), `max_surp` (maximum token surprisal in the sentence), `spike_frac_35` (fraction of tokens whose surprisal exceeds a threshold of 3.5 nats), `spike_frac_50` (fraction of tokens whose surprisal exceeds 5.0 nats), `rep_1gram` (proportion of repeated unigrams within the sentence), `top1_prob_mean` (mean probability assigned by the model to its most likely token at each position), `top1_minus_true_mean` (mean difference between the probability of the model's top-ranked token and the probability of the observed token), `digit_ratio` (fraction of characters that are numeric digits), and `upper_ratio` (fraction of uppercase characters). Together, these features characterize both sentence-level fluency and localized irregularities, enabling effective discrimination between coherent and incoherent synthetic outputs.

Precision Confidence using Wilson LCB. Observed data across all features is noisy, without a sharp division between coherent and incoherent sentences. This is illustrated in Fig. 3, the t-Distributed Stochastic Neighbor Embedding (t-SNE) chart. The noisiness of the data indicates that high-precision discriminators will result in a smaller subset of selected sentences. Because of this, the measured precision $\hat{p} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$ can be overly optimistic when the number of accepted examples $n = \mathrm{TP} + \mathrm{FP}$ is small. To enforce conservative high-precision, we compute the *Wilson score 95% lower confidence bound* (LCB) on precision. Other confidence interval metrics break down when n is small, or precision is close to the boundary of 0 or 1. Wilson LCB discounts high observed precision that is supported by only a few accepted examples, thus as n grows, the LCB approaches the measured precision (Brown, Cai, and DasGupta 2001).

Precision Constrained Threshold Calibration All three methods (LR, SVM, RF) employed k-fold cross-validation threshold selection on the training set to stabilize threshold selection. For each fold, models were trained on $k-1$ folds and validated on the held-out fold. All three classifiers were trained to produce a predicted probability of coherence for each sentence x: $p(x) = P(\text{coherent} \mid x)$. To convert these probabilities into binary accept/reject decisions, probability thresholds t in $t \in [0, 1]$ from 0.0 to 1.0 at steps of 0.005

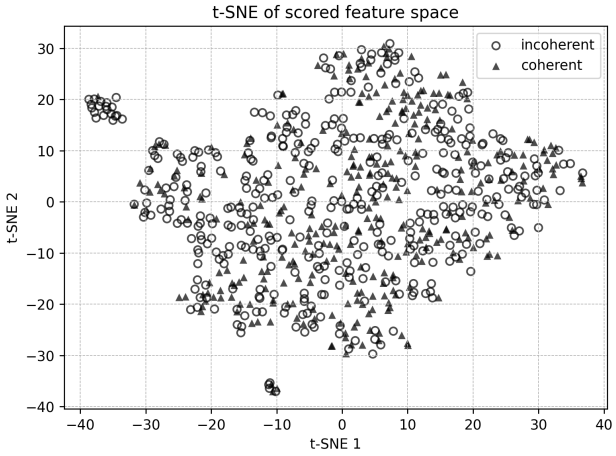

Figure 3: t-SNE visualization of sentence feature vectors (12 token-level statistics; n=750), revealing noisy separation and challenging classification boundaries.

were evaluated. For a given threshold t, a sentence was accepted as coherent if $p(x) \geq t$. For each t, we evaluate the resulting accepted set and compute the Wilson 95% lower confidence bound on precision. We select the threshold that maximizes the achievable lower bound on true precision, yielding the strongest conservative precision guarantee supported by the data. For example, the highest attainable Wilson lower confidence bound for LR occurred at $t = 0.81$, meaning that we are 95% confident that the true precision is at least 0.81. The final threshold was chosen as the median of the thresholds selected across folds, and incorporated into the final decision rules for each discriminator.

Logistic Regression Discriminator A LR discriminator was trained using gradient descent with L2 regularization on features derived from expected-token statistics. After threshold calibration the learned coefficients were transformed from standardized space back to raw feature space for efficient computation. The final deployed discriminator is implemented as a linear rule in raw feature space:

$$\text{keep sentence if} \quad \sum_{f \in \mathcal{F}} a_f \, x_f + c \;\geq\; 0,$$

where x_f is the raw feature value and $\mathcal{F}$ is the retained feature set.

Random Forest Discriminator We evaluate an RF coherence discriminator trained on the same expected-token features. RF provides a group of decision trees capable of capturing higher-order feature interactions that are not expressible by a single linear decision boundary. The RF discriminator was trained using a standard method of using decision trees with bootstrap aggregation. Each tree was trained on a random sample of the training data and a random subset of features at each split, to improve robustness to feature noise. Input features and training set were identical to the LR

training. As with the other discriminators, training was performed exclusively on the training set, using human labels. Rather than relying on the RF's default decision rule, RF produces a posterior probability estimate $p(y = 1 \mid x)$ for each sentence, representing the fraction of trees voting for the coherent class (using 500 trees) (Pedregosa et al. 2011). After threshold selection, the RF model was retrained on the full training set.

Support Vector Discriminator We evaluate a Support Vector Machine (SVM) classifier trained on the same expected-token feature representation. The SVM discriminator was implemented using a radial basis function (RBF) kernel. Prior to training, all feature dimensions were standardized to zero mean and unit variance using statistics computed on the training folds only. Standardization is necessary for SVMs to ensure that features with different numeric scales contribute comparably to the kernel distance computation. Although SVMs typically rely on a fixed decision boundary, our approach is training to output probabilistic scores via Platt scaling, enabling threshold-based decision making rather than fixed margin classification (Pedregosa et al. 2011).

Method 2: Prompt-Based LLM Discriminator

As a baseline, we evaluate a prompt-based discriminator using an Ollama-hosted LLaMA 3.2 model. Each sentence is presented to the model with a classification prompt requesting a binary coherence judgment.[2] The model's response is parsed into an accept/reject decision. Unlike Method 1, this approach relies entirely on the direct text generation of the LLM and does not expose intermediate probabilistic signals or allow explicit calibration of decision thresholds.

Method 3: Cascade Discriminator (Composite Method)

Prompt-based LLM discriminators have been observed in our implementation to run approximately 20 times longer than expected-token discriminators, though this ratio may vary depending on hardware, batch size, and specific deployment configurations. As will be shown experimentally, the Prompt-Based LLM Discriminator demonstrated excellent precision. By using a Method 1 discriminator as a pre-filter, and then sending the resulting set to the Prompt-Based LLM Discriminator, we can increase precision, but with a tradeoff of a lower acceptance rate.

Experiments

We run two experiments: the first to compare and contrast performance of our various discriminators, and the second to investigate sentence diversity of filtered sentences.

Experiment 1: Classifier Comparison

In the first experiment the test set is evaluated on 1,200 human-labeled sentences by each of our five discriminators. The test set was not used in the training of any of the classifiers. Given the high-precision filtering objective of this work, we report the following evaluation metrics. We use the number of true positives, false positives, true negatives, and false negatives (TP, FP, TN, and FN) to compute standard measures of recall, precision and F1 as well as the following

- **Coherent-kept rate**, the fraction of all sentences that are both coherent and accepted: $\text{Coherent-kept} = \frac{\text{TP}}{\text{TP+FN+TN+FP}}$.
- **Incoherent leakage**, $1 - precision$, the fraction of accepted sentences that are incoherent: $\text{Leakage} = \frac{\text{FP}}{\text{TP+FP}}$.

We emphasize incoherent leakage and precision are the most critical metrics, as they directly measure the contamination rate of the accepted corpus.

Results. Table 1 displays classifier performance. The system is designed such that recall is intentionally sacrificed in order to guarantee corpus quality. Consequently, balanced metrics such as F1 are less diagnostic than precision, leakage, and coherent-kept rate.Results reveal a trade-off between precision and coherent-kept rate. LR demonstrated the highest precision and lowest leakage rate of any Method 1 discriminator. However, RF and SVM were comparable. The prompt-based discriminator had better precision, but kept a lower percentage of coherent sentences. The Cascade methods each had 100% precision on our test set, albeit returning a very small set of coherent sentences. The prompt model appears to be conservative, and the cascade further restricts acceptance, reducing false positives at the expense of recall. Because the cascade accepts a very small number of sentences, these precision estimates have high variance and should be interpreted cautiously.

Experiment 2: Sentence Diversity of Filtered Sentences

A potential failure of high-precision filtering is that a discriminator might preferentially retain only short, formulaic, repetitive sentences, or otherwise negatively reduce the training set distribution. Additionally we want to ensure we have good coverage of terms from our ontology for training a language to logic translator. To evaluate whether this occurs, we compare distributional diversity statistics between the sentences selected by each discriminator and the unfiltered synthetic corpus. We report three complementary measures. Sentence length is summarized by the average token count per sentence, AvgTok Lexical diversity is measured using the type-token ratio (TTR),

$$\text{TTR} = \frac{|\mathcal{V}|}{\sum_{s \in \mathcal{S}} |s|},$$

where $\mathcal{V}$ is the set of unique tokens in $\mathcal{S}$. Higher TTR indicates greater lexical variety normalized by corpus length. To approximate semantic diversity of a filtered set, we compute the mean pairwise cosine similarity between sentence representations in Term Frequency-Inverse Document Frequency (TF-IDF) space. Given TF-IDF vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, cosine

[2]Prompt and code for experiments described in this paper can be found at `https://github.com/ontologyportal/sumonlp/tree/master/src/weirdness_detector`

Method	Precision	Recall	F1	Coherent-kept	Leakage
LR	0.771	0.130	0.223	0.0617	0.229
RF	0.760	0.195	0.310	0.0926	0.240
SVM	0.657	0.192	0.297	0.0909	0.343
Prompt-based LLaMA 3.2	0.905	0.100	0.180	0.0475	0.095
Cascade (Prompt $\wedge$ LR)	1.000	0.032	0.061	0.0150	0.000
Cascade (Prompt $\wedge$ RF)	1.000	0.039	0.074	0.0183	0.000
Cascade (Prompt $\wedge$ SVM)	1.000	0.040	0.078	0.0192	0.000

Table 1: Comparison of coherence filtering methods on the held-out test set.

similarity is defined as $\cos(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\| \, \|\mathbf{x}_j\|}$, and the reported value is the mean similarity over all sentence pairs in the filtered set. Lower values indicate greater semantic diversity, and point to a greater diversity of terms selected from the ontology. Together, these metrics allow us to assess whether precision-oriented filtering collapses into trivial or repetitive patterns.

Results. Table 2 displays the results of our diversity experiment. The average token length does drop somewhat as sentences are filtered, however TTR shows that the diversity of sentences is not reduced to trivial, repetitive sentences. While TF-IDF cosine similarity of sentences does increase from the baseline of all 6,000 test sentences, it is still very low, and sentence diversity does not collapse.

Method	Avg tok.	TTR	TF-IDF cos.
All sentences (baseline)	7.74	0.118	0.0088
RF	6.19	0.244	0.0140
SVM	6.13	0.289	0.0114
LR	6.01	0.291	0.0164
Prompt-based LLaMA 3.2	6.55	0.340	0.0116
Cascade (Prompt $\wedge$ LR)	5.86	0.416	0.0220
Cascade (Prompt $\wedge$ RF)	6.09	0.388	0.0179
Cascade (Prompt $\wedge$ SVM)	5.70	0.485	0.0143

Table 2: Diversity statistics. TTR measures normalized lexical diversity (higher is more diverse). TF-IDF cosine is the mean sampled pairwise similarity (lower is more diverse).

Discussion of Results

These results show that token-level signals provide a solid basis for coherence filtering. Prompt-based discriminators can achieve extremely low leakage but are opaque, difficult to calibrate, and computationally expensive at scale. In contrast, the expected-token approach yields simpler models over a small number of features, enabling fast inference, explicit threshold control, and principled tradeoff selection.

Future Work

Results are model-specific and may vary with different base language models or generation procedures. Additionally, the prompt-based discriminator was evaluated using a single prompt; alternative prompts may yield different tradeoffs.

A diversity study should more thoroughly investigate distributional shift introduced by high-precision filtering, including Kullback–Leibler (KL) divergence. KL divergence measures more generically whether the filtered set approximates the full (ground truth) distribution of desired sentences sufficiently well. Given the stated goal of high precision filtering, a semantic shift is expected, thus analysis behind any shift, and ultimate impact to logic translation capabilities is essential. A more interesting result may be obtained by capturing not only that a sentence violates common sense, but the exact reason for common sense violation, and adding preventative logic constraints. For example, "John eats the boat dock." violates the constraint that people only eat food (outside of metaphor, which is handled by our metaphor translator (Singley, Pease, and Thompson 2025)). More complex restrictions are possible, and would add to the common-sense knowledge in SUMO, improving the capabilities of the reasoning system, as well as our ability to translate language to logic. As part of the final sentence generation pipeline, created sentences can then be tested against an automated theorem prover, which can provably demonstrate sentence incoherence. This will comprise a significant phase of our future work.

Conclusion

This work addresses a critical challenge in automated natural language to formal logic translation: ensuring the quality of synthetically generated training data. We demonstrated that LLM-based coherence filtering can dramatically improve the quality of synthetic sentence-logic pairs, providing a scalable solution to what would otherwise require prohibitively expensive manual curation. Our comparative analysis reveals important tradeoffs between precision, computational efficiency, and coverage. Expected-token discriminators offer fast filtering with explicit threshold control, achieving 77% precision. The prompt-based discriminator achieves 90% precision at reduced throughput. The cascade method demonstrates perfect precision on our test set, though at significantly reduced recall. Critically, our diversity analysis confirms that high-precision filtering maintains lexical and semantic variety rather than collapsing to trivial patterns. By establishing a methodology for precision-constrained filtering with explicit confidence bounds, we provide a framework applicable to any synthetic data generation pipeline where quality dominates quantity. The methods presented here are expected to yield substantially better performance in synthetic sentence generation, natural language inference, data to text generation, among other natural language reasoning tasks.

References

Brown, L. D.; Cai, T. T.; and DasGupta, A. 2001. Interval Estimation for a Binomial Proportion. *Statistical Science* 16(2):101–133.

Fuchs, N. E., and Schwitter, R. 1996. Attempto Controlled English (ACE). In *The First International Workshop on Controlled Language Applications*. Katholieke Universiteit Leuven.

Furbach, U.; Glöckner, I.; Helbig, H.; and Pelzer, B. 2008. LogAnswer-A Deduction-Based Question Answering System (System Description). In *Automated Reasoning: 4th International Joint Conference, IJCAR 2008 Sydney, Australia, August 12-15, 2008 Proceedings 4*, 139–146. Springer.

Gangemi, A.; Guarino, N.; Masolo, C.; Oltramari, A.; and Schneider, L. 2002. Sweetening Ontologies with DOLCE. In *International Conference on Knowledge Engineering and Knowledge Management*, 166–181. Springer.

Helbig, H. 2005. *Knowledge Representation and the Semantics of Natural Language*. Springer Science & Business Media.

Karp, Peter D. 1992. *The Design Space of Frame Knowledge Representation Systems*. SRI International.

Niles, I., and Pease, A. 2001. Towards a standard upper ontology. In *Proceedings of the International Conference on Formal Ontology in Information Systems - Volume 2001*, FOIS '01, 2–9. New York, NY, USA: Association for Computing Machinery.

Niles, I., and Pease, A. 2003. Linking Lexicons and Ontologies: Mapping WordNet to the Suggested Upper Merged Ontology. In *Proceedings of the IEEE International Conference on Information and Knowledge Engineering*, 412–416.

Otte, J. N.; Beverley, J.; and Ruttenberg, A. 2022. BFO: Basic Formal Ontology. *Applied Ontology* 17(1):17–43.

Pease, A., and Benzmüller, C. 2010. Ontology Archaeology: Mining a Decade of Effort on the Suggested Upper Merged Ontology. *The ECAI-10 Workshop on Automated Reasoning about Context and Ontology Evolution*.

Pease, A., and Murray, W. 2003. An English to Logic Translator for Ontology-Based Knowledge Representation Languages. In *International Conference on Natural Language Processing and Knowledge Engineering, 2003. Proceedings. 2003*, 777–783. IEEE.

Pease, A. 2011. *Ontology: A Practical Guide*. Articulate Software Press.

Pease, A. 2021. Choosing a Logic to Represent the Semantics of Natural Language. In *In Proceedings of the 4th International Conference on Logic and Argumentation (CLAR2021)*.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12:2825–2830. Accessed 26 Jan 2026. See: `https://scikit-learn.org/stable/modules/svm.html` for Support Vector Machines module.

Roberts, A.; Raffel, C.; Lee, K.; Matena, M.; Shazeer, N.; Liu, P. J.; Narang, S.; Li, W.; and Zhou, Y. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. Technical report, Google.

Singley, J.; Pease, A.; and Thompson, R. 2025. Metaphor detection and translation for representing natural language in formal logic. In *Proc. of International Conference on Computational Science and Computational Intelligence (CSCI2025)*. IEEE.

Sutcliffe, G. 2026. ShZZaM: An LLM+ATP Natural Language to Logic Translator. Personal communication.

Tammet, T.; Järv, P.; Verrev, M.; and Draheim, D. 2024. Experiments with LLMs for Converting Language to Logic. In *International Conference on Neural-Symbolic Learning and Reasoning*, 305–314. Springer.

Thompson, R.; Pease, A.; Kölsch, M.; and Toutsios, A. 2025. Grounding terms from an ontology for use in autoformalization: Tokenization is all you need. In H. Gilpin, L.; Giunchiglia, E.; Hitzler, P.; and van Krieken, E., eds., *Proceedings of The 19th International Conference on Neurosymbolic Learning and Reasoning*, volume 284 of *Proceedings of Machine Learning Research*, 130–136. PMLR.

Wilcox, E. G.; Pimentel, T.; Meister, C.; Cotterell, R.; and Levy, R. P. 2023. Testing the Predictions of Surprisal Theory in 11 Languages. *Transactions of the Association for Computational Linguistics* 11:1451–1470.

Yang, Y.; Xiong, S.; Payani, A.; Shareghi, E.; and Fekri, F. 2023. Harnessing the power of large language models for natural language to first-order logic translation. *arXiv preprint arXiv:2305.15541*.

Zheng, X.; Li, N.; Luan, X.; Wang, K.; Shi, L.; Sun, M.; and Wang, H. 2025. Beyond Correctness: Exposing LLM-generated Logical Flaws in Reasoning via Multi-step Automated Theorem Proving. *arXiv preprint arXiv:2512.23511*.