Conversational Lexicography: Querying Lexicographic Data on Knowledge Graphs with SPARQL through Natural Language

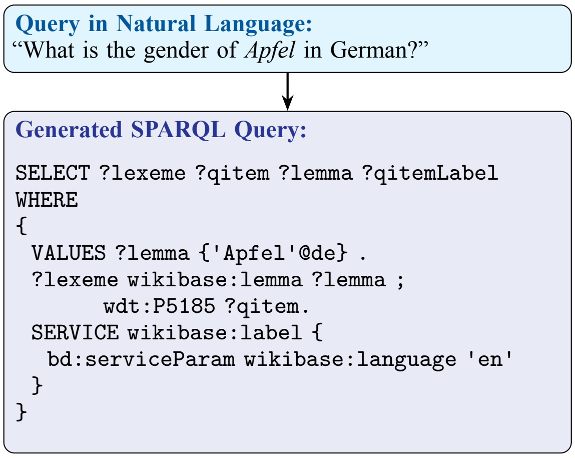
Kilian Sennrich
Department of Informatics
University of Zurich
kilian.sennrich@uzh.ch

Sina Ahmadi
Department of Computational Linguistics
University of Zurich
sina.ahmadi@uzh.ch

Abstract

Knowledge graphs offer an excellent solution for representing the lexical-semantic structures of lexicographic data. However, working with the SPARQL query language represents a considerable hurdle for many non-expert users who could benefit from the advantages of this technology. This paper addresses the challenge of creating natural language interfaces for lexicographic data retrieval on knowledge graphs such as Wikidata. We develop a multidimensional taxonomy capturing the complexity of Wikidata's lexicographic data ontology module through four dimensions and create a template-based dataset with over 1.2 million mappings from natural language utterances to SPARQL queries. Our experiments with GPT-2 (124M), Phi-1.5 (1.3B), and GPT-3.5-Turbo reveal significant differences in model capabilities. While all models perform well on familiar patterns, only GPT-3.5-Turbo demonstrates meaningful generalization capabilities, suggesting that model size and diverse pre-training are crucial for adaptability in this domain. However, significant challenges remain in achieving robust generalization, handling diverse linguistic data, and developing scalable solutions that can accommodate the full complexity of lexicographic knowledge representation.

🤗 **Dataset** | Models (**Phi-1.5** | **GPT-2**)



Query in Natural Language:
"What is the gender of *Apfel* in German?"

Generated SPARQL Query:

```
SELECT ?lexeme ?qitem ?lemma ?qitemLabel
WHERE
{
 VALUES ?lemma {'Apfel'@de} .
 ?lexeme wikibase:lemma ?lemma ;
       wdt:P5185 ?qitem.
 SERVICE wikibase:label {
   bd:serviceParam wikibase:language 'en'
 }
}
```

Figure 1: Conversational lexicography: enabling natural language queries to KGs by automatically generating SPARQL code, eliminating the need for manual query writing

1 Introduction

Knowledge Graphs (KGs) have emerged as scalable and interoperable resources for organizing and accessing the vast volumes of data produced in our digital age. Particularly for lexicographic data, as found in dictionaries, KGs offer an ideal structure for capturing the complex relationships between words, meanings, and linguistic patterns due to the highly interrelated nature of this information (Ahmadi, 2022, p. 14). The preservation and accessibility of lexicographic data is crucial for standardizing language understanding, supporting linguistic research, documenting cultural diversity (Gregson et al., 2015), and crucially, increasing interoperability in language technology. Recent advancements in Large Language Models (LLMs) have opened new pathways for creating natural language interfaces to KGs, potentially democratizing access to this structured linguistic knowledge (Avila et al., 2024).

Despite their advantages, KGs remain largely inaccessible to non-technical users due to the specialized knowledge required to query them effectively. Currently, accessing information in KGs requires proficiency in a query language, notably SPARQL, which presents a significant barrier to entry. Users must not only master this technical query language but also understand the specific ontologies and data models that structure each KG (Ngonga Ngomo et al., 2013). Wikidata[1], a prominent open-source KG, employs a collaboratively developed semantic structure that requires detailed knowledge to navigate effectively. This technical complexity limits the broader utility of KGs,

[1] https://www.wikidata.org

particularly for audiences such as language learners, teachers, and other non-technical stakeholders who could benefit from lexicographic data access (Warren and Mulholland, 2020).

This paper addresses the significant research gap in creating effective natural language interfaces for lexicographic data retrieval on KGs such as Wikidata. To that end, we develop a multidimensional taxonomy that captures the complexity of Wikidata's lexicographic data ontology module, systematically categorizing the diverse information requests that may be queried on the KG. Additionally, we create a template-based dataset that maps natural language utterances to corresponding SPARQL queries, designed to reflect the variety of possible information requests identified in our taxonomy. Finally, we conduct preliminary experiments using transformer-based language models of modest parameter sizes to generate SPARQL queries from natural language inputs, as exemplified in Figure 1, evaluating their performance on both seen and unseen utterances to assess the impact of model parameter size and training method.

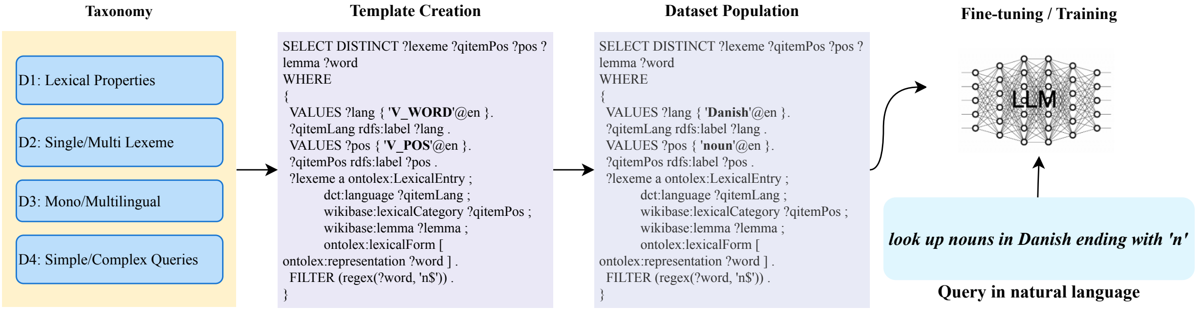
2 Related Work

The translation of natural language queries into SPARQL has received significant attention in recent years, particularly with the advent of LLMs and the increasing importance of KGs. This section provides a brief description of datasets, generation techniques and evaluation methods.

Datasets The development of specialized datasets has accelerated progress in natural language interfaces to KGs. The Question Answering over Linked Data (QALD) series represents a foundational contribution, with QALD-10 offering the most recent iteration supporting both DBpedia and Wikidata queries (Usbeck et al., 2023). Building on this foundation, the Large-Scale Complex Question Answering Dataset (LC-QuAD 2.0) expands the scope with 30,000 natural language utterances paired with corresponding SPARQL queries (Dubey et al., 2019). The DBpedia Natural Language Question Answering (DBNQA) dataset stands as one of the most comprehensive resources, containing nearly 900,000 data tuples for training and evaluation (Hartmann et al., 2018). Addressing the critical need for cross-domain generalization, Kosten et al. (2023) introduce Spider4SPARQL with over 10,000 manually crafted SPARQL queries. Experimental evaluations using LLMs demonstrate that Spider4SPARQL presents substantial challenges in achieving high accuracy.

Generation Approaches to generating SPARQL queries from natural language have evolved from traditional machine learning to increasingly sophisticated neural architectures. Early work by Soru et al. (2018, 2017) establish the foundational *Neural SPARQL Machine* paradigm, comprising a template-based *generator*, a sequence-to-sequence *learner*, and an *interpreter* that translates user inputs into SPARQL. Alternative approaches leverage structural properties of KGs to extract potential RDF triples (Hu et al., 2018; Lin and Lu, 2022), while subsequent advances explore diverse neural architectures, including pre-trained models like BART and T5 (Banerjee et al., 2022). A persistent challenge is handling incomplete vocabulary, particularly entity identifiers in KGs, e.g., Wikidata's `Q811486` for 'tree', that may not appear during training; researchers have addressed this through Named Entity Disambiguators (Xu et al., 2023) and entity masking techniques. For specialized domains, Zou et al. (2021) develope a text-to-SPARQL model utilizing a pointer network-based encoder with relation-aware attention mechanisms, while Qi et al. (2024) introduce Triplet Structure Enhanced T5, which undergoes a specialized pre-training phase to better handle complex query structures. The emergence of LLMs has further transformed this landscape (Perevalov and Both, 2024). D'Abramo et al. (2025) apply in-context learning using Mixtral (8x7B), Llama-3 (70B), and CodeLlama (70B) to achieve state-of-the-art results, while other approaches demonstrate success through fine-tuning (Brei et al., 2024) and one-shot learning (Pliukhin et al., 2023). Rony et al. (2022) propose SGPT, employing transformer encoders with GPT-2 as the decoder and entity placeholders for post-processing.

Evaluation The evaluation of natural language to SPARQL systems has traditionally relied on metrics such as accuracy, BLEU (Papineni et al., 2002), F1-score, or a combination of those (Rony et al., 2022). However, these metrics have limitations, as syntactically different queries can produce identical results. (Cohen and Kim, 2013) propose evaluation frameworks that combine syntactic metrics with semantic correctness assessments to capture the practical utility of generated queries.



```
SELECT DISTINCT ?lexeme ?qitemPos ?pos ?
lemma ?word
WHERE
{
 VALUES ?lang { 'V_WORD'@en }.
 ?qitemLang rdfs:label ?lang .
 VALUES ?pos { 'V_POS'@en }.
 ?qitemPos rdfs:label ?pos .
 ?lexeme a ontolex:LexicalEntry ;
          dct:language ?qitemLang ;
          wikibase:lexicalCategory ?qitemPos ;
          wikibase:lemma ?lemma ;
          ontolex:lexicalForm [
ontolex:representation ?word ] .
 FILTER (regex(?word, 'n$')) .
}
```

```
SELECT DISTINCT ?lexeme ?qitemPos ?pos ?
lemma ?word
WHERE
{
 VALUES ?lang { 'Danish'@en }.
 ?qitemLang rdfs:label ?lang .
 VALUES ?pos { 'noun'@en }.
 ?qitemPos rdfs:label ?pos .
 ?lexeme a ontolex:LexicalEntry ;
          dct:language ?qitemLang ;
          wikibase:lexicalCategory ?qitemPos ;
          wikibase:lemma ?lemma ;
          ontolex:lexicalForm [
ontolex:representation ?word ] .
 FILTER (regex(?word, 'n$')) .
}
```

Figure 2: Our approach to creating SPARQL templates based on a four-dimension taxonomy followed by dataset population and model implementation. The ultimate goal is to infer the models by querying in natural language.

Recent work suggests moving beyond simple comparison with gold standards toward functional correctness testing (Chen et al., 2021), similar to general code generation evaluation approaches.

As such, several research gaps persist in this domain. First, existing datasets predominantly focus on factual knowledge, leaving lexicographical queries underexplored. Second, the optimal approach to handling incomplete vocabulary and generalization remains an open question. Finally, while LLMs show promise for SPARQL generation, their potential specifically for lexicographic data queries remains uncertain.

3 Methodology

We develop a systematic methodology to map natural language queries to SPARQL for lexicographic data in Wikidata, illustrated in Figure 2. This relies on a taxonomy to generate query templates which are then populated with data instances to create a comprehensive dataset. The dataset is subsequently used to train and fine-tune LLMs for the SPARQL generation task. We provide background information about Wikidata in Appendix B.

3.1 Taxonomy for the Lexicographic Data

To systematically approach template creation for lexicographic data, we develop a taxonomy that defines the relevant aspects of translating natural language to SPARQL queries in Wikidata's lexicographic domain. Our taxonomy is based on three criteria:

> **Criterion 1**: It should encompass the full range of SPARQL syntax constructs and operators
> **Criterion 2**: It should cover the variety of use cases for lexicographic data
> **Criterion 3**: It should be particularly detailed in frequently queried areas

These criteria guided the identification of four feature dimensions (D) that capture the heterogeneity of lexicographic queries:

D1: Lexical Properties This dimension addresses Criterion 2 by covering the range of lexicographic properties in Wikidata. These properties serve as fundamental building blocks for SPARQL queries using the lexicographic data ontology module. We classify these properties into the following seven categories, summarized in Table C.1 in the appendix:

- *Linguistic Properties*: Grammatical and morphological features, e.g., grammatical gender, conjugation class
- *Historical References*: Temporal aspects of lexemes, e.g., first attestation
- *Syntactic Functions*: Roles of lexemes within sentences, e.g., auxiliary verb, examples
- *Semantic Relations*: Meaning relationships between lexemes, e.g., synonyms, antonyms
- *Orthographic and Phonetic Features*: Written and spoken forms, e.g., IPA transcription
- *Translation and Lexical Variety*: Cross-linguistic information and variants, e.g., borrowed forms, regional variants
- *Stylistic Attributes*: Context-dependent characteristics, e.g., language register, tone

D2: Single vs. Multi Lexeme Output This dimension focuses on whether the natural language query targets a single lexeme or multiple lexemes. This classification is based on the semantics of the utterance rather than the actual number of lexemes in the output. For example, the question *"What is the grammatical gender of the French word 'livre'?"* is classified as Single-Lexeme Output despite potentially returning multiple homograph lexemes (masculine '*livre*' meaning 'book' and feminine '*livre*' meaning 'pound' as unit of weight).

This dimension is particularly important for addressing Criterion 1, as certain SPARQL keywords and structures are associated with either Single- or Multi-Lexeme queries. Conversely, some utterances inherently imply a Multi-Lexeme Output. An example is the utterance *"Create a French-German-Basque lexicon"*.

D3: Mono- vs. Multilinguality This dimension distinguishes between queries that involve one language versus those that involve multiple languages. Classification is based on the languages of all lexemes that would appear in the output if all variables were included. For instance, the query *"What is the French word for 'fish'?"*, is classified as multilingual because lexemes from multiple languages appear in the result. This dimension addresses Criterion 3.

D4: Simple vs. Complex Queries This dimension analyzes query complexity based on the number of lexical properties involved. While "complex" in literature often refers to queries requiring multiple reasoning steps (Wang et al., 2024), we define simple queries as those containing only one lexical property, e.g., *"From what word is the French word 'cigare' derived?"*, and complex queries as those containing multiple properties. This definition better suits lexicographic data, where users target properties of a single lemma rather than performing multi-step reasoning.

3.2 Implementation

We implement two distinct approaches to fine-tune and train models for natural language to SPARQL:

- First, we fine-tune a pre-trained Phi-1.5 model (Li et al., 2023) using the Low-Rank Adaptation (LoRA) framework. Phi-1.5 is a small language model with 1.3B parameters that demonstrates strong capabilities in both natural language and code generation. For fine-tuning, we use the following hyperparameters: learning rate of 0.0002, train batch size of 4, Adam optimizer, cosine learning rate scheduler, and mixed precision training. Following Schimanski et al. (2024), we limited training to a single epoch to avoid overfitting. The LoRA approach allowed us to fine-tune 0.44% of the model's parameters.

- Second, we train a GPT-2 architecture with 124M parameters (Radford et al., 2019) from scratch using the Hugging Face library. For this model, we use a learning rate of 5e-05, train batch size of 16, Adam optimizer, linear learning rate scheduler, and trained for three epochs.

Both models are trained on data formatted by concatenating natural language utterances prefixed with "`question:`", and corresponding SPARQL queries prefixed with "`answer: <code>`" and "`suffixed with "</code>"`". This format simplifies the parsing of SPARQL code from the output. The training utilized Phi-1.5's tokenizer, which extends GPT-2's BPE vocabulary with special tokens for code representation. We employ two NVIDIA GeForce RTX 3090 GPUs with CUDA 12.4 for training.

3.3 Evaluation

Inspired by Cohen and Kim (2013), we deploy an evaluation framework structured around the following four key principles:

A. **Automatic evaluation** of the text-to-SPARQL model rather than manual;

B. **Functionality** prioritizing functional correctness over exact match, i.e., character-by-character comparison of the generated SPARQL query with a gold standard reference query. In our evaluation setup, we use Chen et al. (2021)'s $pass@k$ metric which generates k responses for a given prompt containing few-shot examples. Each of the generated responses is then run against the KG.[2] If the triples retrieved by the generated query match or include the expected answer triples from the gold standard query, the generated response is deemed correct. The $pass@k$ metric is then calculated as the ratio of all the correctly generated responses (k_{correct}) within the k trials and all generated responses:

$$pass@k = \frac{k_{correct}}{k} \tag{1}$$

C. **Granularity** employing unit test-like checks to evaluate specific aspects of the generated SPARQL queries, including syntax correctness and appropriate variable usage rather than just overall correctness. As such, we define a granularity ratio to assess the fine-grained quality of generated queries as follows:

$$R_{\text{granularity}} = \frac{c_{\text{pass}}}{c_{\text{all}}} \tag{2}$$

[2] Wikidata Query Service: https://query.wikidata.org

where c_{pass} is the number of passed checks and c_{all} is the total number of checks performed. A list of the tests is provided in Appendix C.

D. **Generalization** assessing the model's ability to generalize by altering input questions to trigger different query types. To do so, we transform a training question like *"What is the gender of 'Apfel' in German?"* (requiring a `SELECT` query) into a test question like *"Is the gender of 'Apfel' in German feminine?"* (requiring an `ASK` query), testing whether the model can adapt to this structural change.

Finally, for string-based matching, we report performance using BLEU as implemented in SacreBLEU (Post, 2018).[3]

4 Dataset

To develop a comprehensive dataset mapping natural language utterances to SPARQL queries targeting lexicographic data in Wikidata, we adopt a template-based approach similar to Soru et al. (2017) based on the taxonomies defined in Section 3.1. Each data point in our templates consists of three elements:

1. **utterance**: natural language input reflecting a user's question;
2. **template_name**: identifier for the template in SPARQL containing tags that are later populated with actual words;
3. **query**: the populated SPARQL template aligned with the utterance.

All utterances are in English, though they may reference terms in other languages, e.g., *"What is the grammatical gender of 'livre' in French?"*. The following is an instance in our populated dataset:

utterance: where does the word color come from?
template_name: q20
query:

```
SELECT ?etonymLexeme ?qitemLanguageOfOrigin
       ?etonym ?qitemLanguageOfOriginLabel
WHERE {
  VALUES ?lemma {'color'@en} .
  ?lexeme wikibase:lemma ?lemma ;
          wdt:P5191 ?etonymLexeme.
  ?etonymLexeme dct:language ?qitemOrigin;
                wikibase:lemma ?etonym .
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language 'en'
  }
}
```

To address the limited diversity inherent in template-based approaches, we decouple semantics from syntax by generating multiple variations of utterance templates while preserving their meaning. This is accomplished by using GPT-4 to generate alternative phrasings with random selection during template population with an example provided in Appendix A.

4.1 Template Sources

Our dataset comprises five specialized modules following different paradigms:

Google Templates Following Hazoom et al. (2021), who advocate deriving data from naturalistic environments, we extract questions related to lexicographic data from Google's Natural Questions dataset. We identify relevant lexicographic terms and extract 3,296 user questions containing these terms. To do so, we cluster questions using k-means and `FlagEmbeddings` embedding model (Chen et al., 2024)[4]. We then manually review clusters to identify 639 genuinely relevant questions. The selected questions yield 21 unique SPARQL templates that closely align with typical user questions (see Appendix C.2 for sample cluster). Analysis of the Natural Questions dataset showed 35% multilingual vs. 65% monolingual and 52% complex vs. 48% simple queries, informing our template distribution to meet Criterion 3.

Property Templates To enable efficient Wikidata usage through natural language interfaces, we also create templates covering properties specific to the WikibaseLexemes extension. We manually select 36 relevant properties from lexicographical properties, categorizing them based on their domain (lexeme, sense, or form) and range data type (string, Q-item, etc.). This dual classification resulted in nine archetypal SPARQL templates, which are further adapted to handle multi-lexeme outputs and ASK statements.

Multi-Property Templates These templates address queries requiring multiple pieces of information for a given lexeme. All multi-property queries derive from a single adjustable base template modified to handle both single-result and multiple-result queries. The templates use the `OPTIONAL` keyword to handle cases where properties are unavailable for certain lexemes. Properties are randomly selected from a pool of 211 options (not

[3] nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.4.2

[4] BAAI's BGE-Large variant

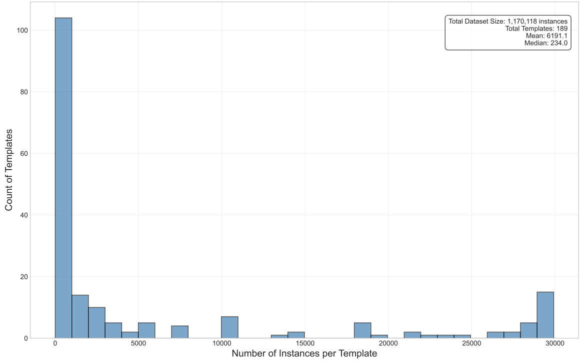


Figure 3: Distribution of the number of populated data tuples per template

restricted to WikibaseLexemes) to prevent overfitting. Two versions of utterance templates were used: single-lexeme and multi-lexeme.

Language-Independent Templates These templates function without specifying the lexeme's language, enabling cross-language lookups. They use string matching (`FILTER(STR(?lemma) = "word")`) rather than language-specific `VALUES` clauses, trading computational efficiency for flexibility. Since these queries can return numerous lexemes, we introduced templates restricting output based on lexical category and grammatical features. This resulted in eight templates covering both language-dependent and language-independent queries.

Rule-Based Templates This paradigm incorporates existing work in lexicographic data querying. We adapted seven templates from SPARQLify[5], a simple form-based query generator. These templates cover advanced use cases employing multiple properties and SPARQL functions as in `regex()`) not represented in other paradigms, such as *"Find at most 50 longest words in {language}"* and *"List at most 50 onomatopoeia in {language}"*.

4.2 Dataset Population

We populate templates by replacing tags with actual lemmas from Wikidata, ensuring that lexemes had relevant properties whenever possible. The data used represents a snapshot from April-May 2024, constrained by Wikidata's query limits (30,000 data points maximum, one-minute computation time). A custom Python program replaced template tags with corresponding population data.

4.3 Dataset Statistics

Our dataset contains 1,270,113 data tuples derived from 189 templates with an average of 6,191 instances per template. Templates populated between 1 (for *limit_t9_P2859* and *order_t9_P2859*) and 29,922 (for *ask_t9_P7243* and *t9_P7243*) data tuples each. Approximately half of the templates populated over 1,000 data tuples. The distribution of the number of populated tuples per template is illustrated in Figure 3. Following Soru et al. (2017), we define the train-test split such that the evaluation dataset contains at most 10% of data points per template, with a maximum of 20 data points. This ensures a balanced evaluation set while maintaining a substantial training set. From our dataset, we include at least one instance of each template in the test set to ensure comprehensive evaluation.

5 Experiments and Results

In order to evaluate the effectiveness of various language models in generating SPARQL queries for lexicographic data on Wikidata, we conduct experiments with three strategically selected models: GPT-3.5-Turbo as a **baseline**, and our **fine-tuned** Phi-1.5 and **trained** GPT-2 models. When evaluated in a zero-shot setting without fine-tuning or training, both Phi-1.5 and GPT-2 failed completely, scoring 0 across all metrics, demonstrating that task-specific adaptation is essential for SPARQL generation with these models.

Our selection of models prioritizes those with modest parameter counts (1.3B for Phi-1.5 and 124M for GPT-2) to demonstrate if effective SPARQL generation can be achieved without requiring computationally expensive models, making deployment more accessible for resource-constrained environments. Additionally, these models represent different training approaches–GPT-3.5-Turbo as a commercial API-based model, Phi-1.5 as a recent code-capable model amenable to parameter-efficient fine-tuning, and GPT-2 as a fully trainable smaller model–providing a diverse evaluation spectrum. For each model, we assess performance using the evaluation framework described in Section 3.3. The results are summarized in Table 1.

5.1 GPT-3.5-Turbo

We evaluate GPT-3.5-Turbo to establish a baseline against which our custom-trained models can be compared. Despite its extensive parameter count,

[5] https://sinaahmadi.github.io/SPARQLify

Model	Parameter	Non-Generalization			Generalization		
		$pass@k$↑	$R_{\text{granularity}}$↑	BLEU↑	$pass@k$↑	$R_{\text{granularity}}$↑	BLEU↑
Phi 1.5	k=1	0.86	0.84	92.1	0	0.7	54.4
GPT-2	k=1	0.90	0.84	94.4	0	0.41	0.3
GPT-3.5 Turbo	k=1	0.87	0.94	99.2	0.41	0.81	72.7
	k=3	0.89	0.95	99.6	0.57	0.84	67.0

Table 1: Performance of few-shot fine-tuned GPT-3.5 Turbo in comparison to our trained and fine-tuned models using $pass@k$ $[0, 1]$ for functionality, $R_{\text{granularity}}$ $[0, 1]$ for granularity and BLEU $[0, 100]$. Although GPT-3.5 Turbo as the baseline performs better than our models, our trained GPT-2 model achieves a higher $pass@k$ despite having significantly less parameters. Due to computational costs, $k = 3$ could not be included for Phi 1.5 and GPT-2.

this model performs poorly when directly asked to generate lexicographic SPARQL queries. We leverage GPT-3.5-Turbo's strong few-shot learning capabilities by employing prompt engineering, sampling two random utterances and corresponding SPARQL queries from the training dataset for each template to create the prompt, with an example in Appendix A.

In the evaluation without generalization, GPT-3.5-Turbo achieves a $pass@1$ score of 0.87 and $R_{\text{granularity}}$ of 0.94. When allowed to generate multiple responses ($k = 3$), performance improves to 0.89 and 0.95 respectively. For the evaluation with generalization, performance drops to a $pass@1$ score of 0.41 and $R_{\text{granularity}}$ of 0.81, improving to 0.57 and 0.84 with $k = 3$, highlighting the challenge of adapting to novel query structures. The same pattern is seen in BLEU scores, except in generalization where the BLEU score with $k = 3$ (67.0) is lower than $k = 1$ (72.7). This counter-intuitive result can be explained by the model's tendency to explore more diverse, but potentially less syntactically aligned, query structures when generating multiple responses. While this diversity improves functional correctness (as measured by $pass@k$), it reduces strict textual similarity to reference queries.

5.2 Phi 1.5

We evaluate Phi-1.5 fine-tuned on our dataset with $k = 1$ only, a decision driven by significant computational demands—the evaluation without generalization alone requires 23 hours to complete. The model achieves a $pass@1$ score of 0.86 and $R_{\text{granularity}}$ of 0.84 in non-generalization scenario.

Our analysis indicates that Phi-1.5 does not attempt to generalize beyond specific SPARQL structures from fine-tuning. While information from utterances is correctly mapped to appropriate positions in the code, the query structure remains closely aligned with training examples. In the generalization scenario, the model struggles significantly with a $R_{\text{granularity}}$ of 0.7, indicating that many generated queries fail to meet basic correctness criteria.

5.3 GPT-2

We evaluate GPT-2 trained from scratch on our dataset, representing a model unexposed to any data except our training examples. Similar to Phi-1.5, we compute results with $k = 1$ only due to computational constraints. In the evaluation without generalization, GPT-2 achieves the highest $pass@1$ score among all models at 0.90, with a $R_{\text{granularity}}$ of 0.84. In the generalization scenario, however, GPT-2's performance deteriorates substantially, with a $R_{\text{granularity}}$ of only 0.41 and BLEU score of 0.3, the lowest among all models. This suggests a high degree of memorization rather than a deeper understanding of the relationship between natural language and SPARQL. The model's strong performance in familiar scenarios coupled with poor generalization indicates effective pattern learning but limited transfer capability.

5.4 Qualitative Analysis

Our qualitative analysis reveals distinct patterns across models. Phi-1.5 demonstrates limited semantic understanding, surprising knowledge of less-resourced language tags, and accurate syntactic mapping, but struggles with generalization, often generating syntactically correct but semantically nonsensical SPARQL code. GPT-2 exhibits similar semantic limitations (interpreting "lengthy words" as words with specific prefixes) and contextual failures, but handles special characters well; in generalization, it produces random word sequences and incomplete syntax. GPT-3.5-Turbo occasionally uses incorrect language tags and struggles with special characters, but shows better understanding of complex utterances and develops creative adaptation strategies like nesting `SELECT` statements within `ASK` blocks. Overall, few-shot GPT-3.5-Turbo achieves superior performance across most metrics, though trained GPT-2 excels in $pass@1$ for familiar queries despite having significantly fewer parameters. These findings suggest that while smaller models can be effectively trained for domain-specific SPARQL generation within familiar patterns, robust generalization to novel query structures may require larger models with diverse pre-training or more sophisticated fine-tuning approaches.

6 Conclusion and Discussion

This paper addresses the challenge of creating natural language interfaces for lexicographic data in KGs. We develop a multidimensional taxonomy capturing the complexity of Wikidata's lexicographic data representation based on which we create a template-based dataset with over 1.2 million mappings from natural language utterances to SPARQL queries. Our experiments with GPT-2, Phi-1.5, and GPT-3.5-Turbo reveal significant differences in model capabilities. While all models perform well on familiar query patterns ($pass@1$ scores ranging from 0.86 to 0.90), only GPT-3.5-Turbo demonstrates meaningful generalization capabilities ($pass@3$ of 0.57 in the generalization scenario). This suggests that model size and diverse pre-training are crucial for adaptability in this domain. This work is timely and important as KGs continue to grow in complexity, creating an urgent need for accessible interfaces.

Limitations and Future Work While our experiments demonstrate promising results with models of modest size, future work should explore more recent reasoning-focused models such as DeepSeek, QwQ, and Llama-3, which may offer improved performance for complex SPARQL generation tasks. Additionally, scaling experiments with larger model variants could help establish the relationship between model size and SPARQL generation capabilities, potentially identifying optimal efficiency-performance tradeoffs for this specific task. As such, future work should focus on improving model generalization through more diverse training data, expanding this approach to other KGs, particularly Dbnary (Sérasset, 2012), and conducting user studies to evaluate practical utility for different stakeholder groups in lexicography and linguistics.

Acknowledgments

This work was supported by the Swiss National Science Foundation (MUTAMUR; no. 213976). The authors are grateful to the constructive feedback of the anonymous reviewers.

References

Sina Ahmadi. 2022. Monolingual alignment of word senses and definitions in lexicographical resources. *arXiv preprint arXiv:2209.02465*.

Caio Viktor S Avila, Vânia MP Vidal, Wellington Franco, and Marco A Casanova. 2024. Experiments with text-to-SPARQL based on ChatGPT. In *2024 IEEE 18th International Conference on Semantic Computing (ICSC)*, pages 277–284. IEEE.

Debayan Banerjee, Pranav Ajit Nair, Jivat Neet Kaur, Ricardo Usbeck, and Chris Biemann. 2022. Modern baselines for SPARQL Semantic Parsing. In *SIGIR*.

Julia Bosque-Gil, Jorge Gracia, and Elena Montiel-Ponsoda. 2017. Towards a module for lexicography in ontolex. In *Proceedings of the LDK 2017 Workshops: 1st Workshop on the OntoLex Model*, volume 1899 of *CEUR Workshop Proceedings*, pages 74–84.

Felix Brei, Johannes Frey, and Lars-Peter Meyer. 2024. Leveraging small language models for Text2SPARQL tasks to improve the resilience of AI assistance. *arXiv preprint arXiv:2405.17076*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. *arXiv:2402.03216*.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde, Jared Kaplan, Harrison Edwards, Yura Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, David W. Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William H. Guss, Alex Nichol, Igor Babuschkin, Suchir Balaji, Shantanu Jain, Andrew Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew M. Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. *ArXiv*, abs/2107.03374.

Philipp Cimiano, Paul Buitelaar, John McCrae, and Michael Sintek. 2011. LexInfo: A declarative model for the lexicon-ontology interface. *Web Semantics: Science, Services and Agents on the World Wide Web*, 9(1):29–51.

K. Bretonnel Cohen and Jin-Dong Kim. 2013. Evaluation of SPARQL query generation from natural language questions. In *Proceedings of the Joint Workshop on NLP&LOD and SWAIE: Semantic Web, Linked Open Data and Information Extraction*, pages 3–7, Hissar, Bulgaria.

Jacopo D'Abramo, Andrea Zugarini, and Paolo Torroni. 2025. Investigating large language models for text-to-SPARQL generation. In *Proceedings of the 4th International Workshop on Knowledge-Augmented Methods for Natural Language Processing*, pages 66–80, Albuquerque, New Mexico, USA. Association for Computational Linguistics.

Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. LC-QuAD 2.0: a large dataset for complex question answering over Wikidata and DBpedia. In *The Semantic Web – ISWC 2019*, pages 69–78, Cham. Springer International Publishing.

Gil Francopoulo, Monte George, Nicoletta Calzolari, Monica Monachini, Nuria Bel, Mandy Pet, and Claudia Soria. 2006. Lexical markup framework (LMF). In *International Conference on Language Resources and Evaluation-LREC 2006*, page 5.

J. Gregson, J.M. Brownlee, R. Playforth, and N. Bimbe. 2015. *The Future of Knowledge Sharing in a Digital Age: Exploring Impacts and Policy Implications for Development*. Number 125 in IDS Evidence Report. Brighton.

Ann-Kathrin Hartmann, Edgard Marx, and Tommaso Soru. 2018. Generating a large dataset for neural question answering over the DBpedia knowledge base. In *Workshop on Linked Data Management, co-located with the W3C WEBBR*, volume 2018.

Moshe Hazoom, Vibhor Malik, and Ben Bogin. 2021. Text-to-SQL in the wild: A naturally-occurring dataset based on stack exchange data. In *Proceedings of the 1st Workshop on Natural Language Processing for Programming (NLP4Prog 2021)*, pages 77–87, Online. Association for Computational Linguistics.

Sen Hu, Lei Zou, Jeffrey Xu Yu, Haixun Wang, and Dongyan Zhao. 2018. Answering natural language questions by subgraph matching over knowledge graphs. *IEEE Trans. Knowl. Data Eng.*, 30(5):824–837.

Catherine Kosten, Philippe Cudré-Mauroux, and Kurt Stockinger. 2023. Spider4SPARQL: a complex benchmark for evaluating knowledge graph question answering systems. In *2023 IEEE International Conference on Big Data*, pages 5272–5281. IEEE.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need II: Phi-1.5 technical report. *arXiv preprint arXiv:2309.05463*.

Jia-Huei Lin and Eric Jui-Lin Lu. 2022. SPARQL generation with an NMT-based approach. *J. Web Eng.*, 21(5).

John P. McCrae, Julio Bosque-Gil, Jorge Gracia, Paul Buitelaar, and Philipp Cimiano. 2017. The Ontolex-Lemon model: Development and applications. In *Proceedings of eLex 2017 Conference*, pages 19–21, Leiden, Netherlands. Lexical Computing.

Elena Montiel-Ponsoda, Guadalupe Aguado De Cea, Asunción Gómez-Pérez, and Wim Peters. 2008. Modelling multilinguality in ontologies. *COLING 2008: Companion volume: Posters*, pages 67–70.

Axel-Cyrille Ngonga Ngomo, Lorenz Bühmann, Christina Unger, Jens Lehmann, and Daniel Gerber. 2013. Sorry, I don't speak SPARQL: translating SPARQL queries into natural language. In *Proceedings of the 22nd international conference on World Wide Web*, pages 977–988.

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.

Aleksandr Perevalov and Andreas Both. 2024. Towards LLM-driven natural language generation based on SPARQL queries and RDF knowledge graphs. 3rd international workshop on knowledge graph generation from text (Text2KG) at ESWC.

Dmitrii Pliukhin, Daniil Radyush, Liubov Kovriguina, and Dmitry Mouromtsev. 2023. Improving subgraph extraction algorihtms for one-shot SPARQL query generation with large language models. In *QALD/SemREC@ ISWC*.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Jiexing Qi, Chang Su, Zhixin Guo, Lyuwen Wu, Zanwei Shen, Luoyi Fu, Xinbing Wang, and Chenghu Zhou. 2024. Enhancing SPARQL query generation for knowledge base question answering systems by learning to correct triplets. *Applied Sciences*, 14(4).

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Md Rashad Al Hasan Rony, Uttam Kumar, Roman Teucher, Liubov Kovriguina, and Jens Lehmann. 2022. SGPT: A generative approach for SPARQL query generation from natural language questions. *IEEE access*, 10:70712–70723.

Tobias Schimanski, Jingwei Ni, Mathias Kraus, Elliott Ash, and Markus Leippold. 2024. Towards faithful and robust LLM specialists for evidence-based question-answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1913–1931, Bangkok, Thailand.

Gilles Sérasset. 2012. Dbnary: Wiktionary as a LMF based multilingual RDF network. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation, LREC 2012, Istanbul, Turkey, May 23-25, 2012*, pages 2466–2472. European Language Resources Association (ELRA).

Tommaso Soru, Edgard Marx, Diego Moussallem, Gustavo Publio, Andre Valdestilhas, Diego Esteves, and Ciro Baron Neto. 2017. SPARQL as a foreign language. In *Proceedings of the Posters and Demos Track of the 13th International Conference on Semantic Systems*, volume 2044 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Tommaso Soru, Edgard Marx, André Valdestilhas, Diego Esteves, Diego Moussallem, and Gustavo Publio. 2018. Neural machine translation for query construction and composition. *CoRR*, abs/1806.10478.

Ricardo Usbeck, Xi Yan, Aleksandr Perevalov, Longquan Jiang, Julius Schulz, Angelie Kraft, Cedric Möller, Junbo Huang, Jan Reineke, Axel-Cyrille Ngonga Ngomo, Muhammad Saleem, and Andreas Both. 2023. Qald-10 – the 10th challenge on question answering over linked data. *Semantic Web*, Preprint:1–15. Preprint.

Ruijie Wang, Luca Rossetto, Michael Cochez, and Abraham Bernstein. 2024. QAGCN: Answering multi-relation questions via single-step implicit reasoning over knowledge graphs. In *The Semantic Web*, pages 41–58, Cham. Springer Nature Switzerland.

Paul Warren and Paul Mulholland. 2020. A comparison of the cognitive difficulties posed by SPARQL query constructs. In *International Conference on Knowledge Engineering and Knowledge Management*, pages 3–19. Springer.

Silei Xu, Shicheng Liu, Theo Culhane, Elizaveta Pertseva, Meng-Hsi Wu, Sina Semnani, and Monica Lam. 2023. Fine-tuned LLMs know more, hallucinate less with few-shot sequence-to-sequence semantic parsing over Wikidata. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5778–5791, Singapore. Association for Computational Linguistics.

Jianyun Zou, Min Yang, Lichao Zhang, Yechen Xu, Qifan Pan, Fengqing Jiang, Ran Qin, Shushu Wang, Yifan He, Songfang Huang, et al. 2021. A Chinese multi-type complex questions answering dataset over Wikidata. *arXiv preprint arXiv:2111.06086*.

A Prompts

ChatGPT prompt for utterance diversification

Input: Rewrite the following sentence 10 times in a way someone may ask it on Google. Keep it short: Look up {*word*} in any language in {*form_word*} form
Output:

- Translate {*word*} to any language in {*form_word*}?
- How to say {*word*} in {*form_word*} across languages?
- Find {*word*} in any language's {*form_word*}?
- Discover {*word*} in different languages, {*form_word*}?
- Search for {*word*} in {*form_word*} form in all languages?
- Get {*word*} translations in {*form_word*} in any language?
- Lookup {*word*} in various languages, {*form_word*}?
- How to express {*word*} in {*form_word*} across languages?
- Find {*word*} equivalents in {*form_word*} across languages?
- Translate {*word*} into {*form_word*} in any language?

Few-Shot Prompt Example for GPT-3.5 Turbo

```
Utterance 1:
Apfel gender in German
SPARQL 1:
SELECT ?lexeme ?qitem ?lemma ?qitemLabel
WHERE
{
  VALUES ?lemma {'Apfel'@de} .
  ?lexeme wikibase:lemma ?lemma ;
          wdt:P5185 ?qitem.
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language 'en'
  }
}

Utterance 2:
medailon gender Czech
SPARQL 2:
SELECT ?lexeme ?qitem ?lemma ?qitemLabel
WHERE
{
  VALUES ?lemma {'medailon'@cs} .
  ?lexeme wikibase:lemma ?lemma ;
          wdt:P5185 ?qitem.
  SERVICE wikibase:label {
    bd:serviceParam wikibase:language 'en'
  }
}

Utterance:
What is Probekörpers gender in German?
```

B Lexicographical Data on Wikidata

This section provides essential background on lexicographic data and its representation on Wikidata.

B.1 Lexicographic Data

Lexicography is the field concerned with dictionaries and reference works. Lexicographic data encompasses all information contained within dictionaries or reference works, which may range from traditional print dictionaries to digital databases and KGs. The ontology for lexicographic data on the Semantic Web is primarily supported by OntoLex-Lemon (McCrae et al., 2017), which is based on the Lexicon Model for Ontologies (lemon). This model relies on LexInfo (Cimiano et al., 2011), LMF (Francopoulo et al., 2006), and LIR (Montiel-Ponsoda et al., 2008). The OntoLex lexicography module, known as *lexicog* (Bosque-Gil et al., 2017), provides key concepts like *LexicalEntry* and *LexicalSense* that were influential in Wikidata's development. Wikidata has expanded beyond representing concepts to include structured descriptions of words through lexemes, forms, and senses. The lexicographic data module follows the *Wikibase* data model, extended with the Wikibase-Lexemes ontology module that introduces the data types *Lexemes*, *Forms*, and *Senses*.

Lexemes A lexeme is a fundamental vocabulary unit that can take various forms including simple words, complex words, phrasal words, and multi-word expressions. In Wikidata, lexemes have:

- Unique IDs starting with 'L', e.g., `L870817` for '*Steilkurve*' in German
- Lemmas providing human-readable representations, e.g., 'book'
- Language specification using Q-items, e.g., `Q1860` for English
- Lexical category indicated by Q-items, e.g., `Q34698` for adjective
- Statements describing properties not specific to forms or senses
- Forms for each combination of grammatical features
- Senses describing different meanings

Lemmas A lemma serves as a location pointer for information within a reference work. In Wikidata, lemmas are implemented as `MultilingualTextValues`[6] to accommodate languages with active diagraphia such as Serbian which uses both Cyrillic and Latin alphabets. The canonical form of the lexeme, typically the infinitive form of verbs, is used as the lemma. For example, the lemma for the English noun '*color*' would include both '*colour*' for British English and '*color*' for American English. Further, lemmas are not unique, and the combination of lemma, language, and lexical category is not unique either. For instance, there are two German nouns with the lemma '*See*' that differ only in gender, with '*der See*' meaning '*the lake*' and '*die See*' meaning '*the sea*'. These two meanings cannot be understood as a single lexeme, as they have different forms based on their gender. In RDF, Wikidata lexemes are represented as `ontolex:LexicalEntry`, connected to their senses with the `ontolex:sense` property and to their forms with the `ontolex:lexicalForm` property. Each lexeme has an associated lemma (`wikibase:lemma`) and language (`dct:language`).

Senses A sense represents one of the multiple meanings a word can have, arising from polysemy or homonymy. In Wikidata, senses are attributed to lexemes and identified by unique IDs (lexeme ID + `-S` + decimal number as in `L16168-S1` for the act of booking in the "book" lexeme `L16168`). Each sense typically includes a gloss providing a natural language definition and may have statements describing relationships with other senses and items (synonyms, antonyms, etc.).

Forms A form refers to the specific manifestation of a lexeme in a grammatical context. In Wikidata, forms have unique identifiers (lexeme ID + `-F` + decimal number as in `L16168-F1` for the simple past of 'book') and are characterized by grammatical features and statements providing information about usage, pronunciation, etc.

Properties Properties model relationships between subjects and objects in KGs. In Wikidata, properties describe the data value of a statement and have labels, descriptions, and aliases in multiple languages. Each property has a specific data type and a unique identifier with a `P` prefix. Lexicographic properties are a subset used with the WikibaseLexeme data model.

[6] `https://www.mediawiki.org/wiki/Wikibase/DataModel#MultilingualTextValues`

C Evaluation

Category	Property
Linguistic Properties	- grammatical gender (P5185) - conjugation class (P5186) - word stem (P5187) - derived from lexeme (P5191) - combines lexemes (P5238) - homograph lexeme (P5402) - valency (P5526) - requires grammatical feature (P5713) - paradigm class (P5911) - grammatical aspect (P7486) - predicate for (P9970)
Historical References	- attested in (P5323) - first attested from (P6684)
Syntactic Functions	- auxiliary verb (P5401) - classifier (P5978) - location of sense usage (P6084) - usage example (P5831) - creates lexeme type (P5923) - false friend (P5976)
Semantic Relations	- synonym (P5973) - antonym (P5974) - troponym of (P5975) - said to be the same as lexeme (P11577) - pertainym of (P8471)
Orthographic / Phonetic Features	- Han character in this lexeme (P5425) - IPA transcription (P898) - X-SAMPA code (P2859) - Slavistic Phonetic (P5276) - pronunciation (P7243)
Translation	- translation (P5972) - variety of lexeme, form or sense (P7481)
Stylistic and Phonological Attributes	- language style (P6191) - collective noun for animals (P6571) - tone or pitch accent class (P5426)

Table C.1: A taxonomic classification of Wikidata Lexicographic Properties organized by categories

For the granularity test, the following checks are performed:

- The response must start with either `SELECT` or `ASK`
- If it starts with `SELECT`, there must be at least one variable starting with ? before the `WHERE` clause
- If it starts with `ASK`, there must be a `WHERE` clause following directly after
- Every { must have a corresponding }
- The response must not contain the keyword `VALUES`
- The response must contain at least one of the following variables: *?lexeme, ?lemma, ?form, ?sense, ?qitem, ?qitemlabel*
- The response must not contain any Q-items that are not in the known Q-items

Index	Utterance
1	what is the definition of low birth weight
2	what does the prefix re mean in medical terminology
3	what does e/m stand for in medical terms
4	what does ncd stand for in medical terms
5	what does acs stand for in medical terms
6	in military terms what does gi stand for
7	what does pvc stand for in medical terms
8	what does mi stand for in medical terms
9	what is a pa c in medical terms
10	what does la stand for in medical terms
11	what does ts stand for in medical terms
12	how do you write twice a day in medical terms
13	what does dc stand for in medical terms
14	what does ta stand for in medical terms
15	what does ibm stand for in medical terms
16	what is the definition of an asthma attack
17	what is the full meaning of cpr in first aid
18	what is the meaning of rx in medical line
19	meaning of od and bd in medical term
20	medical term meaning condition of stones in the ureters

Table C.2: Utterances potentially targeting lexicographic information in one of the clusters of the Google Templates. This cluster is dominated by utterances about medical abbreviations. However, the presence of an utterance discussing military abbreviations (index 6), suggests that the clustering considers not only the topic of the utterance, but also its lexicographical category.