SPOT: Bridging Natural Language and Geospatial Search for Investigative Journalists

Lynn Khellaf* **Ipek Baris Schlicht*** **Tilman Miraß**
Julia Bayer **Tilman Wagner** **Ruben Bouwmeester**
Deutsche Welle Innovation, Bonn/Berlin
https://innovation.dw.com/
hey@findthatspot.io

Abstract

OpenStreetMap (OSM) is a vital resource for investigative journalists doing geolocation verification. However, existing tools to query OSM data such as Overpass Turbo require familiarity with complex query languages, creating barriers for non-technical users. We present SPOT, an open source natural language interface that makes OSM's rich, tag-based geographic data more accessible through intuitive scene descriptions. SPOT interprets user inputs as structured representations of geospatial object configurations using fine-tuned Large Language Models (LLMs), with results being displayed in an interactive map interface. While more general geospatial search tasks are conceivable, SPOT is specifically designed for use in investigative journalism, addressing real-world challenges such as hallucinations in model output, inconsistencies in OSM tagging, and the noisy nature of user input. It combines a novel synthetic data pipeline with a semantic bundling system to enable robust, accurate query generation. To our knowledge, SPOT is the first system to achieve reliable natural language access to OSM data at this level of accuracy. By lowering the technical barrier to geolocation verification, SPOT contributes a practical tool to the broader efforts to support fact-checking and combat disinformation.

1 Introduction

Investigative journalists frequently rely on OpenStreetMap (OSM) (OSM contributors, 2017) as a vital tool for geolocation verification or research because of its detailed and comprehensive coverage of various locations. However, non-technical users face challenges due to required knowledge of query languages (such as OverpassQL[1]) for data retrieval.

Although language models have been applied to relational database interactions, their use in OSM-based applications is still limited and not tailored to the needs of investigative journalists. Lawrence and Riezler (2016) and Will (2021) for instance introduced datasets and applications that employ neural-network-based semantic parsers to transform natural language into intermediate query formats. Similarly, Staniek et al. (2024) introduced the OverpassT5 model along with benchmarking data for directly querying OSM. However, prior datasets are not directly applicable to the current use case, as they assume prerequisite knowledge of OSM functionalities. While there are AI-powered geolocation tools available to support investigative journalists, they either don't or fail to work effectively with unstructured text inputs (Chen, 2025; Graylark, 2025), or are based on source code that is not publicly available or utilize closed Large Language Models (LLMs) (Meixner, 2025).

To this extent, we present SPOT, an AI-powered, fully open source and open weight geospatial tool designed for investigative journalism, although other potential applications are conceivable. As illustrated in Figure 1, SPOT includes a pipeline for generating artificial training data tailored to user requirements and the OSM tagging system. Its backbone model leverages LLaMA 3 (Touvron et al., 2023), which is fine-tuned on the generated data. During inference, SPOT transforms user input into YAML-based queries which are enriched with predefined OSM tag bundles by using a semantic search engine. Additionally, SPOT provides a user-friendly graphical interface that enables users to seamlessly enter their unstructured search requests, with results displayed interactively on a map. Places of interest can be further explored in detail via integrated external tools such as GoogleStreetView. SPOT is publicly accessible at https://www.findthatspot.io/, with its

*Equal Contribution

[1] https://wiki.openstreetmap.org/wiki/Overpass_API/Overpass_QL

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations), pages 71–81

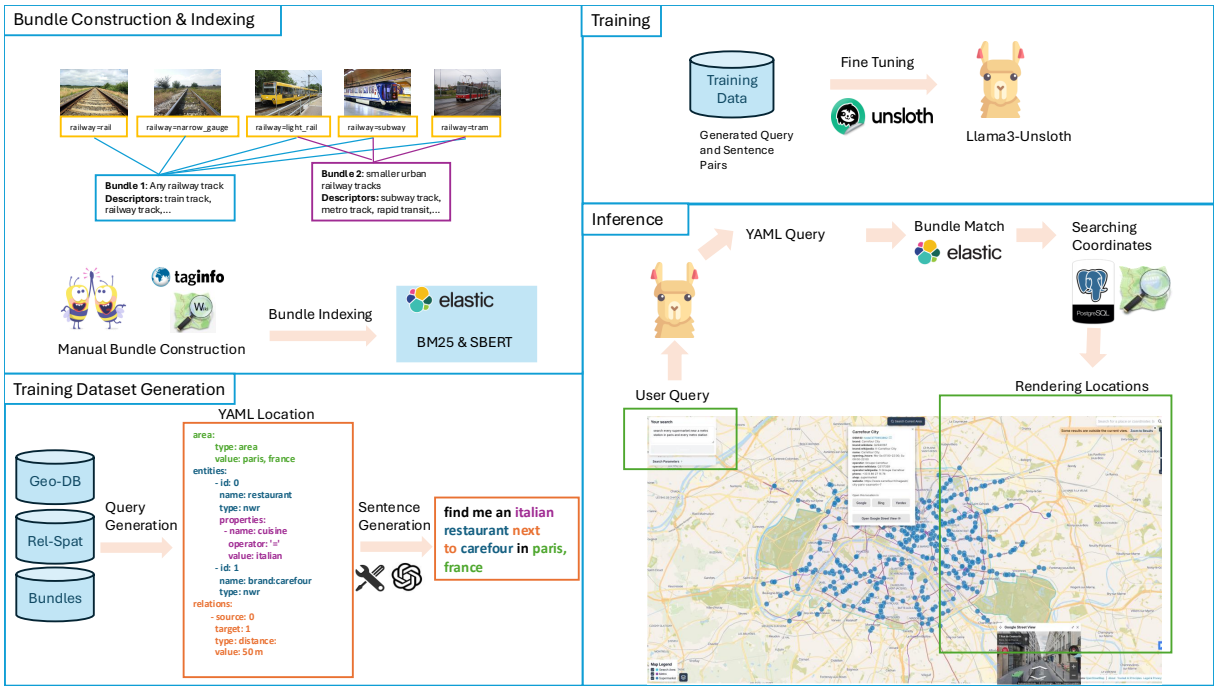

Figure 1: Overview of SPOT's OSM-based pipeline, from tag bundle indexing and semantic search, through artificial sentence and YAML pair generation, to model fine-tuning and interactive inference.

source code hosted on GitHub[2]. Moreover, the fine-tuned LLaMA 3 model, along with other benchmarked LLMs (detailed in Section 4), is available on HuggingFace[3].

2 Related Work

2.1 Text-to-Structured Language

Several research studies have explored ways for users to interact with databases without requiring technical knowledge of structured query languages. The most common approach is to transform natural language questions into SQL queries (text-to-SQL) to facilitate interaction with relational databases, which is closely related to the current use case. Recent advances in this area have explored both prompt-based methods and parameter-efficient tuning of LLMs (Zhu et al., 2024; Shi et al., 2024). For example, Jang et al. (2023) applied adapter tuning to T5 (Raffel et al., 2020), while Zhang et al. (2024) used adapter tuning and merging on LLaMA. Other work has focused on prompt engineering: Gao et al. (2024) proposed DAIL-SQL to improve example selection in in-context learning, and Lee et al. (2025) introduced MCS-SQL, which uses multi-prompting for text-to-SQL generation.

Despite the growing importance of OSM for applications such as geo-verification in journalism, natural language interaction with OSM has been relatively under-researched compared to text-to-SQL. Some research (Lawrence and Riezler, 2016; Will, 2021) proposes the use of semantic parsers to convert natural language queries into intermediate representations that include elements from OSM tags, which can be used to create downstream OSM queries. In contrast, Staniek et al. (2024) tackled the direct text-to-OverpassQL task, creating a dataset of natural language inputs paired with their corresponding OverpassQL queries. They also introduced a task-specific evaluation metric that considers surface string similarity, semantics, and syntax. Their evaluation indicated that explicit pre-training of sequence-to-sequence models like OverpassT5 was not beneficial, while few-shot prompting with GPT-4 performed the best.

Unlike previous approaches, the intermediate representation step in SPOT is multi-layered. To handle variations in query styles (e.g., typos or different terms for the same object) and to allow for updates to OSM tags without needing to retrain the language model, we employ multiple processing steps. SPOT queries are structured in YAML and initially do not contain any OSM tag elements. In a second step, object and property names are passed through a semantic search engine and replaced with

[2]Source code: https://github.com/dw-innovation/kid2-spot

[3]Model weights: https://huggingface.co/DW-ReCo

Tool	Input	Customization	External Data Integration	Open Source
Overpass Turbo (Turbo, 2025)	OT Query	via Query	✓	✓
GeoGuessr GPT (Meixner, 2025)	Unstructured Text	via Chat	X	X
GeoSpy (Graylark, 2025)	Image	NA	✓	X
EarthKit (Chen, 2025)	Semi-structured Text	via Query	✓	✓
SPOT	Unstructured Text	User Guided Search	✓	✓

Table 1: Comparison of OSM-based, AI-supported geolocation verification tools.

the best-fitting OSM tag bundles required for the final OSM database request. We fine-tuned an instance of LLaMA 3 to generate the initial YAML. This state-of-the-art LLM is vastly more performant than our earlier T5-based approach (Khellaf et al., 2023), in which we encountered limitations addressing several key requirements.

2.2 OSM Datasets

The datasets (Lawrence and Riezler, 2016; Will, 2021; Staniek et al., 2024) are currently the only publicly available resource designed for natural language interaction with OSM. They allow users to query OSM using its tagging system, based on coordinates, specific tag types or meta-information such as changes made by particular users. These datasets, however, are primarily intended for users who are familiar with OSM's tagging logic, making them difficult to use for those without prior experience.

In contrast, our tool is designed for visual location verification, allowing users to perform the search using natural language descriptions without requiring OSM expertise. Our approach focuses on visual features such as objects, their properties and the spatial relationships between them, while excluding meta-information irrelevant to the task. For this purpose, we have developed a pipeline for artificial data generation tailored to these specific needs.

2.3 Comparison of Geolocation Tools

There are numerous geolocation tools that have a similar target audience, with and without AI support. Among the most popular for investigative journalists are the original Overpass Turbo (Turbo, 2025) (not using AI), GeoGuessr GPT (Meixner, 2025), GeoSpy (Graylark, 2025) and EarthKit (Chen, 2025). Table 1 contains a design comparison of the aforementioned tools with SPOT. Both SPOT and GeoGuessr GPT (which uses ChatGPT with a custom prompt) accept unstructured text as input, while the other tools rely on structured queries, images, or semi-structured text. In the case of EarthKit, users are presented with OSM tags and must manually select the relevant ones to complete their query.

Of these tools, only SPOT and EarthKit offer full stack open source software and AI models, allowing anyone to host them on their own infrastructure. In terms of integration, GeoGuessr GPT does not connect to any external tools or OSM other than GPT, while EarthKit only integrates with OSM. The remaining tools offer integration with Google Maps or Google Street View. In addition to linking to the location on Google, Bing and Yandex, SPOT also features an OpenStreetView.com integration for a detailed view of identified locations, increasing its utility for investigative work.

3 Overview of SPOT

As shown in Figure 1, SPOT has four main components: bundle construction and indexing, training data generation, training and inference. Each component is briefly described in the following subsections.

3.1 Bundle Construction and Indexing

To bridge the gap between natural language and the OSM tagging system, we developed a static *bundle list* that groups visually similar (individual or combinations of) OSM tags. This list maps natural language descriptors to relevant OSM tags, taking into account the ambiguity and variability of everyday language. For example, terms such as *light rail*, *subway* and *tram* are all mapped to the same bundle representing "smaller urban railway tracks". This approach helps to mitigate inconsistencies in OSM tagging, where multiple tags or tag combinations can refer to objects that are frequently referred to by the same terms.

To make them searchable, the bundle lists are indexed via Elasticsearch[4]. We index both the raw text and its semantic embeddings to deal with typos and paraphrases. The semantic embeddings are

[4] https://www.elastic.co/elasticsearch

vectorized using the `all-MiniLM-L6-v2` version of the SBERT sentence transformer (Reimers and Gurevych, 2019). This setup allows for a hybrid search approach that combines BM25 with SBERT-based retrieval.

3.2 Training Dataset Generation

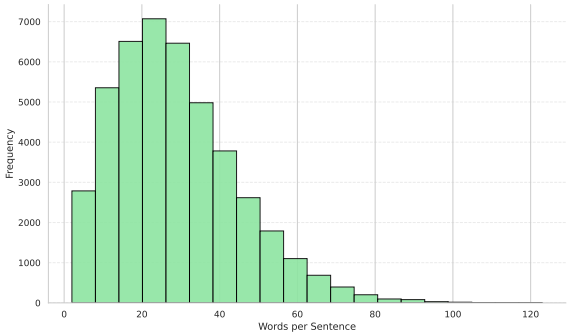

Figure 2: Sentence length distribution of the generated sentences

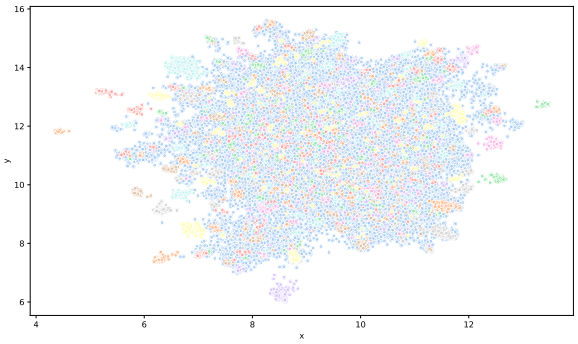

Figure 3: Semantic Diversity Visualization of sentence embeddings of the generated sentences using UMAP and HBSCAN. Blue dots indicate the noisy points that do not belong to any clusters (16,416 points in total).

Prior to development, we conducted a user study with the in-house SPOT development team and our expert OSINT community to collect descriptions of scenes based on images. From this study, we derived a list of user requirements (Appendix A.1) to guide system development. Key findings included the high prevalence of generic terms for objects and spatial relations, as well as frequent typos and grammar errors.

As illustrated in Figure 1, we designed a novel YAML-based structure to simplify data handling, overcoming the challenges associated with JSON's strict syntax (Tam et al., 2024). The structure contains all relevant information, namely search area, entities, properties, and spatial relations. We implemented a framework that creates any number of YAML combinations via random draft of values for the semantic fields. Relation types distinguish between distance and contains relations, as inspired by the user requirements. In addition to specific distance values (such as *within 100 meters*), the model is trained to translate vague relative spatial terms (such as *nearby*, *next to*) into concrete values (*next to* for instance is defined as 50 meters, the full list in Appendix A.2). The multi-lingual area names used in the artificial data are extracted from the public map database NaturalEarthData[5]. The information from the YAML queries with additional text style (e.g. typos) and persona (e.g. fact-checker) specifications is then used to dynamically generate prompts, which is in turn used to turn the YAML into a synthetic natural query sentences using GPT-4o (OpenAI, 2023).

In total, we used 7 personas and 5 writing styles, we provide them in Appendix A.3. The number of generated samples for training is 43976, 2350 of which form the development set. An example prompt is shown in Table 9. As shown in Figure 2, the generated dataset contains different length of sentences. To evaluate the semantic diversity of the generated dataset, we first performed sentence embedding using SBERT. We then used UMAP (McInnes et al., 2018) to project these high-dimensional embeddings into 2D space for visualization, while preserving local semantic relationships. UMAP was configured with 50 nearest neighbours, a minimum distance of 0.1, a target dimensionality of two, and a fixed random seed to ensure reproducibility. We then applied HDBSCAN (Campello et al., 2015) to the resulting 2D embeddings. HDBSCAN is a density-based clustering algorithm that can detect clusters of varying shapes and identify outliers. HDBSCAN was configured with a minimum cluster size and minimum samples parameter both set to 5. The algorithm identified 1,274 distinct clusters but did not assign cluster labels to 16,416 sentences, treating them as noise. A graph of the result can be seen in Figure 3. The considerable number of clusters, along with a substantial proportion of unclustered sentences, indicates that the generated dataset exhibits significant semantic diversity.

3.3 Training and Inference

We fine-tuned an open-source LLM on the synthetic dataset (described in Section 3.2) by using

[5] https://www.naturalearthdata.com/

- Find a tattoo shop and a doityourself shop, both within 2.5 ft of each other.
- Find a restroom and an american football field in 米林根, 巴登-符堡, 国, no more than 28 meters apart.
-In the region of Ward County, North Dakota, United States, seek out a campsite alongside a production studio, specifically one that is situated on a street whose name concludes with the suffix "-der-Tann-Straße."
- Let's see. I'm looking for a 新星堂. Then there's a moving walkway. It has a traffic lane numbered 484 and a car lane numbered 581. I also need to find a monument whose name starts with ""emin du Ro"". All of these should be found within a distance of 75556 miles from one another.
- Could you kindly locate a play area within the confines of Comuna Vadu Moţilor?
- Find a bowling cemter located three hundrd kilomters away from a camera shop.

Table 2: Examples from the training dataset showing different features (e.g. long/short sentence, properties, typos, non-Latin alphabet, etc.).

the `unsloth library`[6]. The fine-tuning process employed Low-Rank Adaptation (Hu et al.) with a rank of 32 and an alpha scaling factor of 64. Training was conducted with a batch size of 8 and the learning rate was set to 1e-5 with a weight decay of 0.01. Early stopping was activated with a patience of 10 epochs and evaluation was performed every 200 steps.

We host the SPOT language model using HuggingFace Inference Endpoints[7]. A backend built with FastAPI[8] handles post-processing of the model output, such as replacing names with corresponding OSM tags. The backend forwards user queries to a PostgreSQL database with the PostGIS extension, indexed with the OSM planetary dataset[9], to retrieve spatial coordinates and details about the detected objects. The results are then finally displayed on an interactive map in the UI.

4 Experiments

Total	195 samples
Named area	143 samples
No Area (bbox)	52 samples
Properties	63 samples
Typos	36 samples
Grammar Mistakes	39 samples
Relative Spatial Terms	43 samples
Contains Relation	48 samples
Distance Relation	121 samples

Table 3: Breakdown of samples in the benchmarking dataset.

4.1 Experimental Setup

Benchmarking Dataset. We constructed a benchmarking dataset consisting of real user queries to assess the viability of several candidate LLMs as query translators. The queries were generated by a pool of investigative journalists, fact-checkers, and verification experts from Deutsche Welle while trying to geolocate sample images using an early version of SPOT. The resulting list was then filtered based on how well the queries aligned with the OSM database structure and its resulting limitations. Table 3 shows statistics on the prevalence of different requirements in the dataset. Table 4 highlights some example queries from this study. These sentences showcase some aspects of the linguistic variety the system might be faced with and needs to handle.

Evaluation Metric. As evaluation metric, we evaluated the percentage of the matches across areas, entities, properties and relations. Since the entity and property names detected by the model might be correct but not covered by the static bundle list, we employed the SBERT transformer also used for the bundle indexing. We considered a ground truth and a model prediction a match if their cosine similarity exceeded 0.8. We additionally counted the number of hallucinated/omitted entities and properties.

4.2 Results

We evaluated several LLMs as semantic parsers. As a baseline, we used the multilingual T5 variant, mT5, which has shown strong performance in past studies on the generation of structured output despite its relative small size (Khellaf et al., 2023; Staniek et al., 2024). To adapt mT5 to our task, we applied LoRa adapter learning. In addition, we obtained baseline results from GPT-4o by testing it with zero-shot and few-shot prompting (the full prompts are provided in Appendix A.4).

We then compared the baseline results with several widely used open LLMs from different companies: LLaMA 3 (Dubey et al., 2024) from Meta, Mistral (Jiang et al., 2023) from Mistral

[6] https://unsloth.ai/
[7] https://ui.endpoints.huggingface.co/
[8] https://fastapi.tiangolo.com/
[9] https://wiki.openstreetmap.org/wiki/Planet.osm

- all Don Quijote that are in a retail building with a purple roof coluor in 東京都
- Find me a bus platform next to a Cheesecake Factory restaurant and a building with a red roof in Dubrovnik.
- Focus on Arch, Switzerland. Find a restaurant within 1.5 km of a bus station. The restaurant should have a public toilet inside.
- Search for a planetarium containing a public toilet. It should be within 85,800 yards of a public clock.
- Find a speet kamera within 100 meater from antenna in Paraiba
- I'm looking for a supermarket from a brand ending in "ermarché" with a parking lot next to it and a power line running past it in less than 15 meters distance.

Table 4: Examples from the benchmarking dataset.

LLM	Company	Unsloth's Version
Mistral	Mistral	`unsloth/Mistral-Nemo-Base-2407-bnb-4bit`
LLaMA 3	Meta	`unsloth/llama-3-8b-bnb-4bit`
Phi	Microsoft	`unsloth/Phi-3-medium-4k-instruct-bnb-4bit`
Qwen2.5	Alibaba	`unsloth/Qwen2.5-14B`

Table 5: Open source LLMs that were examined as potential semantic parsers with their company name and model code from Unsloth (Han et al., 2023).

Adaptation	Model	Area	Entity	Entity*	Property	Relation
Zero-shot	**GPT-4o**	88.14	2.28	90.21	3.03	9.8
One-shot		89.18	1.13	92.03	10.96	11.11
Adapter Tuning	**mT5**	88.21	72.34	90.02	48.89	37.01
	Mistral	**93.33**	82.54	95.01	**56.58**	45.45
	Phi	92.82	79.59	94.10	53.33	**53.90**
	LLaMA 3	92.31	**81.41**	96.15	50.00	48.05
	Qwen2.5	92.31	**82.31**	**95.69**	51.95	52.60

Table 6: Accuracy of the models in identifying areas, entities, properties and relations. Entity* is the accuracy when associated properties are excluded. **Bold results** are the top results.

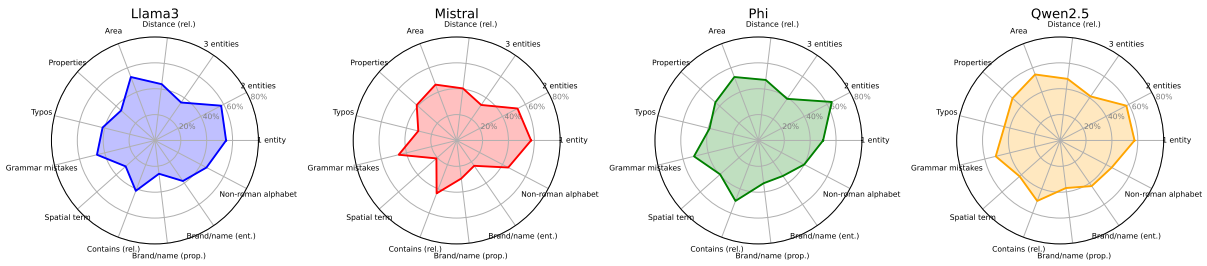

Figure 4: Analysis of LLaMA 3, Mistral, and Phi regarding the ratio of perfect YAML generations various metadata categories. It highlights inter- and intra-model differences in feature handling.

Adaptation	Model	Entity		Property	
		Missed	Hallucinated	Missed	Hallucinated
Zero-shot	**GPT-4o**	48	37	53	10
One-shot		40	34	50	11
Adapter Tuning	**mT5**	51	31	15	6
	Mistral	27	21	17	6
	Phi	30	22	18	7
	LLaMA 3	20	16	18	7
	Qwen2.5	23	17	19	6

Table 7: The number of omitted/hallucinated entities and properties of each tested model.

AI, Phi (Abdin et al., 2024) from Microsoft, and Qwen (Qwen et al., 2025) from Alibaba. We applied adapter training as detailed in Section 3.3 to the quantized versions of their (due to hardware constraints) small/medium models (as summarized in Table 5).

As shown in Figure 6, the fine-tuned LLMs outperformed both GPT-4o and mT5 in all aspects. All fine-tuned LLMs have similar scores for areas, entities, and entities without properties. Noticeably high scores were achieved by Mistral for property, and Qwen2.5 for relation prediction. Qwen2.5 having the most parameters could indicate that relation identification is a task that requires advanced reasoning skills. Furthermore, the fine-tuned LLMs generated fewer hallucinations and omissions com-

pared to the baseline models (shown in Table 7).

We performed a more nuanced analysis of the generated outputs using meta tags, indicating the use of area names, properties, typos, grammar mistakes, spatial terms, brand names (as entities or properties), non-Roman characters, the presence of distance or contains relations, and the number of entities up to three. The percentage of perfectly generated YAML queries for each category is shown in Figure 4. Faulty grammar, typos, and non-Roman characters in particular posed a challenge to the models. Despite these similarities, some model-specific differences are visible, such as Phi and Qwen2.5 performing slightly better when relations were defined using spatial terms.

Finally, we assessed whether the generated output was parsable, as a well-formatted output is essential for the rest of the query pipeline. Based on our benchmark data set, only LLaMA 3 and GPT-4o consistently produced parsable output, leading to the selection of LLaMA 3 as the primary parser for SPOT. A custom parser was deemed too unreliable and potentially detrimental to the inference speed. Although not specifically fine-tuned in languages other than English, the model appears to be able to interpret queries in a variety of languages, although this was not further tested.

5 Conclusion

SPOT represents a significant step forward in making OSM more accessible to non-technical users, particularly investigative journalists, through an easy-to-use natural language interface. By addressing the complexity of OSM query languages with a data pipeline that generates any amount of synthetic data, a static list of descriptors, and tag bundles that allow users to perform geospatial searches using their natural language, SPOT improves the usability of OSM data. Our evaluations demonstrate its ability to handle different linguistic styles, grammatical errors and different types of object relationships, achieving state-of-the-art performance in query interpretation with fine-tuned LLaMA 3 and other LLMs. This work bridges the gap between complex geospatial query languages and practical, intuitive interfaces.

Despite its strengths, SPOT's reliance on synthetic data, limits in hardware and a small benchmark dataset highlight potential avenues for future improvement. We further aim to expand language support, add multimodal features such as image queries, and explore an alternative chat interface to further improve usability. Lastly, we plan to conduct comprehensive end-to-end evaluations with SPOT users to assess all components of the system, including the overall user experience.

Acknowledgments

This project is led by the Deutsche Welle Research and Cooperation Projects team and was co-funded by BKM ("Beauftragte der Bundesregierung für Kultur und Medien," the German Government's Commissioner for Culture and Media).

Limitations

While our approach performs well in several cases, it does not fully capture the complexity of real-world user queries. Users may phrase their queries ambiguously or use implicit descriptions rather than naming entities directly ('somewhere to eat' instead of 'restaurant', for example). In addition, references to entities with multiple interpretations, such as ambiguous landmarks, can introduce challenges that our current setup does not explicitly address. Another limitation is our reliance on OSM as the primary knowledge source. While OSM provides broad coverage, its data may be incomplete or inconsistent in certain regions. Addressing more diverse data sources and improving the handling of ambiguous or underspecified queries are important areas for future work.

Ethics Statement

SPOT democratizes access to geospatial data, but there are several ethical considerations. First, the underlying LLMs may contain inherent biases that could influence query interpretation and results. In addition, the OSM data itself has uneven coverage across regions, potentially limiting the utility of SPOT in under-represented areas.

Regional differences in tagging conventions also present challenges. Although our bundling approach mitigates some inconsistencies, cultural and regional idiosyncrasies in describing places may not be fully captured in our current implementation, reflecting potential limitations in the geographic perspective of the development team.

The most important ethical consideration is privacy. By lowering the technical barriers to geolocation identification, SPOT could potentially facilitate invasions of privacy through the analysis of

images or videos shared on for example social media. While these capabilities already exist through tools such as Overpass Turbo, SPOT's accessibility heightens concerns. We believe that the benefits for legitimate fact-checking and investigative journalism outweigh these risks, but emphasize that users should only use SPOT for ethical purposes, such as verifying public information rather than tracking individuals. Ongoing work includes exploring additional safeguards to prevent misuse while preserving functionality for legitimate uses.

The broader impact of the tool lies in its potential to empower journalists around the world to verify information more efficiently, potentially countering misinformation and strengthening factual reporting in an era of increasing manipulation of digital information.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, et al. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Ricardo J. G. B. Campello, Davoud Moulavi, Arthur Zimek, and Jörg Sander. 2015. Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Trans. Knowl. Discov. Data*, 10(1).

Jett Chen. 2025. Earthkit. Accessed: 2025-02-10.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Dawei Gao, Haibin Wang, Yaliang Li, Xiuyu Sun, Yichen Qian, Bolin Ding, and Jingren Zhou. 2024. Text-to-sql empowered by large language models: A benchmark evaluation. *Proc. VLDB Endow.*, 17(5):1132–1145.

Graylark. 2025. Geospy. Accessed: 2025-02-10.

Daniel Han, Michael Han, and Unsloth team. 2023. Unsloth. `http://github.com/unslothai/unsloth`.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Joel Jang, Seungone Kim, Seonghyeon Ye, Doyoung Kim, Lajanugen Logeswaran, Moontae Lee, Kyungjae Lee, and Minjoon Seo. 2023. Exploring the benefits of training expert language models over instruction tuning. In *International Conference on Machine Learning*, pages 14702–14729. PMLR.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Lynn Khellaf, Ipek Baris Schlicht, Julia Bayer, Ruben Bouwmeester, Tilman Miraß, and Tilman Wagner. 2023. Spot: A natural language interface for geospatial searches in osm. *Proceedings of OSM Science 2023*, page 49.

Carolin Lawrence and Stefan Riezler. 2016. Nlmaps: A natural language interface to query openstreetmap. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*, pages 6–10.

Dongjun Lee, Choongwon Park, Jaehyuk Kim, and Heesoo Park. 2025. MCS-SQL: Leveraging multiple prompts and multiple-choice selection for text-to-SQL generation. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 337–353, Abu Dhabi, UAE. Association for Computational Linguistics.

Leland McInnes, John Healy, Nathaniel Saul, and Lukas Grossberger. 2018. Umap: Uniform manifold approximation and projection. *The Journal of Open Source Software*, 3(29):861.

Bill Meixner. 2025. Geoguessr gpt. Accessed: 2025-02-10.

OpenAI. 2023. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

OSM contributors. 2017. Planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. *Preprint*, arXiv:2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Liang Shi, Zhengju Tang, and Zhi Yang. 2024. A survey on employing large language models for text-to-sql tasks. *arXiv preprint arXiv:2407.15186*.

Michael Staniek, Raphael Schumann, Maike Züfle, and Stefan Riezler. 2024. Text-to-OverpassQL: A natural language interface for complex geodata querying of OpenStreetMap. *Transactions of the Association for Computational Linguistics*, 12:562–575.

Zhi Rui Tam, Cheng-Kuang Wu, Yi-Lin Tsai, Chieh-Yen Lin, Hung-yi Lee, and Yun-Nung Chen. 2024. Let me speak freely? a study on the impact of format restrictions on large language model performance. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pages 1218–1236, Miami, Florida, US. Association for Computational Linguistics.

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. 2023. Llama: Open and efficient foundation language models. *Preprint*, arXiv:2302.13971.

Overpass Turbo. 2025. Overpass turbo website. Accessed: 2025-02-10.

Simon Will. 2021. Nlmaps web: A natural language interface to openstreetmap. In *Proceedings of the Academic Track, State of the Map 2021*, pages 13–15.

Chao Zhang, Yuren Mao, Yijiang Fan, Yu Mi, Yunjun Gao, Lu Chen, Dongfang Lou, and Jinshu Lin. 2024. Finsql: Model-agnostic llms-based text-to-sql framework for financial analysis. In *Companion of the 2024 International Conference on Management of Data, SIGMOD/PODS 2024, Santiago AA, Chile, June 9-15, 2024*, pages 93–105. ACM.

Xiaohu Zhu, Qian Li, Lizhen Cui, and Yongkang Liu. 2024. Large language model enhanced text-to-sql generation: A survey. *arXiv preprint arXiv:2410.06011*.

A Appendix

A.1 Requirements for SPOT

We list the requirements of SPOT that serve as also function for the data generation pipeline.

Entity Recognition

- SPOT identifies general categories like restaurant, train station which allows recognition of places based on category type.
- SPOT detects specific brand names, including ‘McDonald’s’, ‘KFC’, ‘Tchibo’ and compound names such as Thalia bookstore.

Entity Properties

- SPOT identifies properties such as ‘organic (food shop)’, ‘Italian (restaurant)’ or colors such as ‘brown (bench)’ for refined queries.
- SPOT interprets quantitative descriptors such as height, floors and house numbers.

Area Recognition

- SPOT supports cities, districts, and regions, including multi-word areas (e.g., "New York") and states such as "Nordrhein-Westfalen."
- SPOT introduces bounding box support for identifying entities within a broader, undefined area.

Relation Recognition

- SPOT interprets both numeric distances (e.g., ‘100 meters’) and written forms (e.g., ‘one hundred meters’).
- SPOT supports terms like ‘next to’, ‘opposite from’ and ‘beside’ to improve natural understanding of spatial relationships.
- SPOT supports distance-based relations 1) radius constraints (e.g. entity A to entity B and entity C) and entity chains (e.g. entity A to B and entity B to entity C).
- SPOT recognizes relationship such as ‘a fountain within a park’ and ‘a shop inside a mall’, ‘a park with a fountain’, ‘hotel with a parking lot’.

Robustness to Different Styles

- SPOT can match descriptors with slight variations such as plurals ("bookshops" vs. "bookshop") and minor differences (‘bookstore’ vs. ‘book shop’).
- SPOT is robust to typos in names and common words (e.g., ‘MacDonalds’ for ‘McDonald’s’)
- SPOT is robust to styles that presents different user profiles such as an experienced fact-checker, beginner, etc. Additionally, it is robust to formal and casual query styles.
- SPOT recognizes area names and locations in code-switching texts (mixture of texts in different languages). For example, area and brand names in non-Roman alphabets such as Cyrillic and Greek.
- SPOT supports both single and multi-sentence structures in user queries.

A.2 Relative Spatial Terms

A list of relative spatial terms and their interpretation can be found in Table 8.

A.3 Styles and Personas

Writing styles randomly selected in each prompt: “in perfect grammar and clear wording”, “in simple language”, “with very precise wording, short, to the point”, “with very elaborate wording”,“as a chain of thoughts split into multiple sentences’.

Personas randomly selected in each prompt: “political journalist”, “investigative journalist”, “expert fact checker”, “hobby fact checker”, “human rights abuse monitoring OSINT Expert”, “OSINT beginner”, “legal professional”.

A.4 Prompts

A.4.1 Dataset Generation

We designed a dynamic prompt with some randomly selected parameters.

An example of the generated sample is shown in Table 9.

A.4.2 Inferencing Prompt

For the one-shot prompt, we appended one sample from the training data to the zero-shot prompt. The matching of each benchmarking samples to one training sample is based on the cosine similarity of their SBERT embeddings.

Index	Distance	Terms
0	25 m	not far away, enclosed by
1	50 m	next to, among, adjacent, beside, side by side, at, next door
2	100 m	near, around it, in close distance to, surrounded from
3	150 m	in front of, close to, opposite from, in surroundings
4	250 m	on the opposite side
5	1000 m	on the edge
6	2000 m	nearby

Table 8: List of relative spatial terms and distance values used during data generation.

Tag Combination	**Prompt**	**Generated Sentence**

`area:`
`  type: bbox`
`entities:`
`- id: 0`
`  name: church`
`  properties:`
`  - name: levels`
`  operator: '>'`
`  value: '56'`
`  type: nwr`
`- id: 1`
`  name: bridge`
`  properties:`
`  - name: name`
`  operator: '~'`
`  value: MK6`
`  type: nwr`
`relations:`
`- source: 0`
`  target: 1`
`  type: distance`
`  value: 16460 m`
	Generate one or more sentences simulating a user using a natural language interface for an AI geolocation search tool that finds locations based on descriptions of objects and their spatial relations. Each object has one main descriptor and optionally additional properties. All properties must be put in a logical connection to the object. Objects can either be single instances, or clusters of multiple of one object which are located in a specific distance radius (e.g. "three houses next to/within 10m of each other"). Mention the area, cover all entities and their respective properties, and describe the respective relations. Stick to the descriptions of entities and relations provided and don't add anything. When describing names or brand (names), be creative in your phrasing (examples being a "book store of brand Thalia" vs. "a Thalia book store", or simply e.g. "a Thalia" if the type of object is not given). Stick to the values of each relation. Distances always refer to a maximum distance. If no distance is given, do not use any terms such as close, near, create sentences such as "find a house and a restaurant". Vary your phrasing. Do not affirm this request and return nothing but the answer. ==Persona== hobby fact checker ==Style== as a chain of thoughts split into multiple sentences ==Input== Objects: - Obj. 0: church | Properties -> levels: above 56 - Obj. 1: bridge | Properties -> name: "MK6" Distances: - All objects are no more than 16460 meters from another. Please take your time and make sure that all the provided information is contained in the sentence.	Looking around an area, I'm trying to find a church that has more than 56 levels. In the same vicinity, not exceeding a distance of 16,460 meters, there should also be a bridge called "MK6".

Table 9: An example parametric prompt used for data generation. Due to space limitations, the prompt formatting was altered. The original prompts can be found in the source code.

Inferencing Prompt

You are a joint entity and relation extractor. Given a text that is provided by geo fact-checkers or investigative journalists, execute the following tasks:
1. Identify the area mentioned in the text. If no area is found, designate its type as 'bbox' and assign its name as 'bbox'. If area is found, designate its type as 'area'.
2. Detect and extract the geographical entities present in the text. Areas are not part of these entities. Entities are always present in a sentence. There are two type of entities: cluster and nwr. The 'cluster' type is clusters of entities, allowing queries like "3 Italian restaurants next to each other" or "at least 5 wind generators nearby." The other entity types belongs to nwr.
3. Extract properties associated with each identified entity, if available. The properties must be related to their types, colors, heights, etc.
4. Identify and extract any relations between the entities if mentioned in the text. We define two relation types: contains and dist. Assign one of them as the relation type. In contains relations, you can recognize relationships such as "a fountain within a park" and "a shop inside a mall.". In contain relation, there is no distance. In dist relation, you interpret both numeric distances (e.g., "100 meters") and written forms (e.g., "one hundred meters"), support terms like "next to," "opposite from," and "beside" to improve natural understanding of spatial relationships, and recognize Multiple distance-based relations are supported, including radius constraints (e "A to B and C") and entity chains (e.g., "A to B and B to C").
Let's think step by step.
Please provide the output as the following YAML format and don't provide any explanation nor note:

```
area:
   type: area type
   value: area name
entities:
 - name: [entity name 1]
   id: [entity id 1]
   type: [entity type 1]
   properties:
    - name: [property name 1]
      operator: [operator 1]
      value: [property value 1]
    - name: [property name 2]
      operator: [operator 2]
      value: [property value 2]
    - ...
 - name: entity name 2
   id: entity id 2
   type: entity type 2
 - ...
relations:
 - source: entity id 1
   target: entity id 2
   type: relation between entity 1 and entity 2
   value: relation distance if the type of relation is dist
 - ...
```

Figure 5: Zero-shot prompt used to query the LLMs, containing instructions and the YAML layout. The prompt includes support for cluster-type entities, which were not available in the deployed system at the time of writing.