**Karl Oskar Magnus Holm**

# LLMs - The Death of GIS Analysis?

## An Investigation into Using Large Language Models for GIS Data Analysis

Specialization Project in Computer Science and Geomatics, June 2024
Supervisor: Hongchao Fan
External supervisors from Norkart: Alexander Salveson Nossum, Arild Nomeland and Rune Aasgaard

Department of Geomatics
Faculty of Engineering
Norwegian University of Science and Technology

# Abstract

To be written

# Contents

**Abstract** **i**

**List of Figures** **v**

**List of Tables** **vi**

**1. Introduction** **1**
1.1. Background and Motivation . . . 1
1.2. Goals and Research Questions . . . 1
1.3. Contributions . . . 2
1.4. Thesis Structure . . . 2

**2. Theory** **3**
2.1. Large Language Models . . . 3
2.1.1. Attention and The Transformer Architecture . . . 5
2.1.2. The GPT Family . . . 6
2.1.3. The BERT Family . . . 6
2.1.4. Gemini . . . 7
2.1.5. Open-Source Alternatives . . . 8
LLaMA 2 . . . 8
Mistral . . . 8
Orca 2 . . . 8
2.2. LLM providers . . . 9
2.3. Geospatial Standards . . . 10
2.3.1. International Standardization Work . . . 10
OGC Standards . . . 10
STAC Api Standard . . . 10
2.3.2. Norwegian Standardization Work . . . 11
SOSI . . . 11
Geovekst . . . 11
Norge digitalt . . . 12
2.4. User Groups . . . 12
2.5. Ethical and Privacy Concerns . . . 14

**3. Related Work** **15**
3.1. GIS with LLMs . . . 15
3.1.1. Taking the Temperature on GIS with LLMs on Social Media . . . 15

3.1.2. Geospatial Context in LLMs . . . 17
3.1.3. Autonomous GIS . . . 18
3.2. Prompt Engineering and Planning Strategies . . . 19
3.3. Retrieval Augmented Generation and Frameworks . . . 20
3.3.1. LangChain . . . 20
3.3.2. AutoGPT . . . 21
3.3.3. AutoGen and Microsoft Semantic Kernel . . . 21
3.3.4. OpenAI Assistants API . . . 22
3.4. Benchmarking and Evaluation of LLMs . . . 22
3.4.1. Evaluation Metrics . . . 22
Human Evaluation . . . 22
Perplexity . . . 22
BiLingual Evaluation Understudy . . . 23
Recall-Oriented Understudy (ROUGE) . . . 23
Diversity . . . 23
3.4.2. Benchmarks . . . 24

**4. Experiments and Results** **25**
4.1. Method: Experimental Plan and Setup . . . 25
4.1.1. Experiment 1: Testing ChatGPT's Ability to Perform Geospatial Analysis . . . 25
4.1.2. Experiment 2: Comparing File Upload and API Calling in ChatGPT-4 . . . 26
4.1.3. Experiment 3: Using ChatGPT's and LangChain to perform API Call . . . 27
4.2. Experimental Results . . . 27
4.2.1. Results for Experiment 1 . . . 27
4.2.2. Results for Experiment 2 . . . 28
4.2.3. Results for Experiment 3 . . . 31

**5. Discussion and Conclusion** **32**
5.1. Using the In-Built ChatGPT Code Interpreter for Geospatial Analysis . . . 32
5.2. Mitigating ChatGPT's Inability to Access Web APIs . . . 33
5.3. Conclusion and Future Work . . . 33
5.3.1. Balancing Accuracy Against Performance and Costs . . . 33
5.3.2. Testing regime . . . 34
5.3.3. Memory and Embeddings . . . 34

**Bibliography** **36**

**Appendices** **42**

**A. Task Description from Norkart** **43**

**Acronyms** **45**

# List of Figures

2.1. Actor map for stakeholders, providers, and other groups and organizations that could have some relevance to an autonomous LLM-based GIS-agent. 4

4.1. The result of ChatGPT when asked to "Find the area best suited for expansion to accommodate residential buildings", using provided GeoJSON datasets. Potentially suitable areas for residential expansion are depicted in blue. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

4.2. Comparison of the resulting outline of Drammen when using file upload (a) and providing an URL to an API endpoint (b) with ChatGPT-4 . . . 30

4.3. Outline of Drammen using LangChain and OpenAI Functions . . . . . . . 31

# List of Tables

2.1. Categories and frequency of ChapGPT usage (Skjuve et al., 2023, pp. 16–17). 13

4.1. Overview of the ability of ChatGPT's Code Interpreter to handle various geospatial data formats . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

# 1. Introduction

## 1.1. Background and Motivation

The field of Large Language Models (LLMs) is an emerging one. Fan et al. (2023, p. 2) found that the proportion of papers about LLMs[1] to arXiv has skyrocketed since 2020, with a six-times increase in percent points from 2022 to 2023. They write that prompt engineering has been extensively used as a way to improve code generation (Fan et al., 2023, p. 7)

## 1.2. Goals and Research Questions

The overarching goal of this specialization project is to investigate how Large Language Model (LLM) can be utilized to make GIS analysis simpler, faster, and more accessible. As exemplified in the task description provided by Norkart (see appendix A), such a system should be able to create a meaningful response to a query such as:

> "Find all buildings within a 100-meter belt that are above 100 meters above sea level and have docks."

The task then is to investigate how modern language models can be used to perform classical GIS analyses using standard GIS technologies like PostGIS and geospatial data catalogues adhering to OGC or STAC standards. Based on the task description I have constructed three research questions that I will attempt to answer in this specialization project report:

1. What is the potential of LLM-based GIS analysis?
2. How can OGC API Features be used in an overlay analysis using ChatGPT-4?
3. How can we give ChatGPT-4 access to external tools?

Researching the potential of using LLMs in GIS could uncover new methodologies for spatial analysis, predictive modeling, and decision-making. The aim with RQ1 would be to assess the capabilities and limitations of integrating machine learning algorithms with Geographic information systems (GISs), and also touch on how a mature LLM-based GIS could impact the daily work of (human) GIS professionals.

[1]Including articles whose title or abstract includes "LLM", "Large Language Model", or "GPT".

RQ2 focuses on the feasibility of using the OGC API - Features Standads in a typical overlay analysis within a conversational AI context like ChatGPT-4, having the users be able to express themselves using natural language queries. An answer to RQ2 would describe how OGC APIs can be called and manipulated in a flexible manner during a conversation to perform spatial queries or analyses, like intersecting layers or filtering features based on certain criteria.

RQ3 delves into the technical and ethical considerations of expanding ChatGPT-4's capabilities through integration with external tools, such as GIS software or data analytics platforms. It would explore options for secure and efficient data exchange, and assess the implications of such access in terms of data privacy and user consent.

## 1.3. Contributions

The main contributions of this specialization project are listed below:

1. Gather relevant literature to facilitate development of a state-of-the-art LLM-based GIS agent.

2. Perform experiments on ChatGPT-4, investigating its abilities in manipulating geospatial data of different formats and access channels.

## 1.4. Thesis Structure

The thesis is divided into four main sections:

- Chapter 2 introduces the theory, tools, and methods necessary to understand the work.

- Chapter 3 discusses previous work related to this thesis.

- Chapter 4 explains the experimental plan and setup, and lists the results of these.

- Chapter 5 discusses some of the finding of the experiments, provide considerations when developing an LLM-based GIS agent, and concludes the project report.

# 2. Theory

*Disclaimer: Parts from a paper written in the theory module "TDT13 - Advanced Text Analytics and Language Understanding" will be reused in the Theory chapter. This includes subsection 2.1.1 and subsection 2.1.3.*

Chapter 2 of this specialization project will talk about the leading technologies in the field of Large Language Model (LLM), which in itself is a subfield of Natural Language Processing (NLP). section 2.1 will go over the leading LLM models, their strengths and weaknesses, and briefly mention differences in model architecture. Section 2.2 will name the most prominent providers of LLM services.

Figure 2.1 shows an actor map which includes potential user groups, regulatory bodies that could affect adoption of AI-based technologies, different types of Large Language Model and their providers, and some relevant geospatial standards for an LLM-based GIS agent. Subsections 2.1 through 2.5 will elaborate on each of these, starting with LLMs.

## 2.1. Large Language Models

Figure 2.1 includes six different LLMs that are all explained in the subsections below. All but the BERT model are made for text generation, with BERT being more adapted for predicting masked tokens. BERT is very efficient at Natural Language Understanding (NLU), and is applicable for a range of downstream tasks, one of which is discussed in section 3.1. The other LLMs are geared towards text generation, with certain members of the GPT family and Gemini having multimodal capabilities as well. As subsection 3.1.1 illustrates, these generative models, known for their great conversational abilities and extensive knowledge from pre-training on large corpora, have been found useful in Geographic information system (GIS) analysis. The combination of conversational and visual skills, particularly in multimodal models, holds significant promise for GIS analysis, where visual interpretation of maps is crucial.

This section will open with an explanation of the building block of modern Large Language Models, namely the Transformer, which employs self-attention. Subsection 2.1.2 and subsection 2.1.3 will then discuss the two most famous families of LLMs, namely the GPT's and the BERT's. Subsection 2.1.4 will present the brand new multimodal Gemini model from Google, and subsection 2.1.5 will list various open-source alternatives.


Figure 2.1.: Actor map for stakeholders, providers, and other groups and organizations that could have some relevance to an autonomous LLM-based GIS-agent.

### 2.1.1. Attention and The Transformer Architecture

Vaswani et al. (2017) managed to achieve new state-of-the-art results for machine translation tasks with their introduction of the Transformer architecture. The Transformer has later been proved effective for numerous downstream tasks, and for a variety of modalities. Titling their paper *Attention Is All You Need*, Vaswani et al. suggest that their attention-based architecture renders Recurrent Neural Networks (RNNs) redundant, due to its superior parallelization abilities and the shorter path between combinations of position input and output sequences, making it easier to learn long-range dependencies (Vaswani et al., 2017, p. 6).

The Transformer employs self-attention, which enables the model to draw connections between arbitrary parts of a given sequence, bypassing the long-range dependency issue commonly found with RNNs. An attention function maps a query and a set of key-value pairs to an output, calculating the compatibility between a query and a corresponding key (Vaswani et al., 2017, p. 3). Looking at Vaswani et al.'s proposed attention function (2.1), we observe that we take the dot product between the query $Q$ and the keys $K$, where $Q$ is the token that we want to compare all the keys to. Keys similar to $Q$ will get a higher score, e.g., be *more attended to.* These differences in attention are further emphasized by applying the softmax function. The final matrix multiplication with the values $V$, being the initial embeddings of all input tokens, will give us a new embedding in which all tokens have some context from all other words. We improve the attention mechanism by multiplying queries, keys, and values with weight matrices learned through backpropagation. Self-attention is a special kind of attention in which queries, keys, and values are all the sequence.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2.1}$$

Attention blocks can be found in three places in the Transformer architecture (Vaswani et al., 2017, p. 5) (I will use machine translation from Norwegian to German as an example):

1. In the encoder block to perform self-attention on the input sequence (which is in Norwegian)

2. In the decoder block to perform self-attention on the output sequence (which is in German)

3. In the decoder block to perform cross-attention (or encoder-decoder attention) where each position in the decoder attends to all positions in the encoder

The Transformer represented a breakthrough in the field of NLP, and is the fundamental building block of LLMs like BERT.

### 2.1.2. The GPT Family

Generative Pre-trained Transformers (GPTs) are a type of LLM first introduced by OpenAI in 2018 (Radford and Narasimhan, 2018). Specifically designed to perform text generation, it is essentially a stack of Transformer *decoders*, and demonstrates through its vast training on unlabelled data help a language model learn good representations, providing a significant performance boost, while alleviating the dependence on supervised learning. While the original Transformer architecture as described by Vaswani et al. (2017) was intended for machine translation, thus having encoder to learn the representation of the origin language representation of a given input sequence and an encoder to learn the representation in the target language and perform cross-attention between the two, the GPT is designed only to *imitate* language. By applying masked multi-head attention, the model is restricted to only see past $k$ (with $k$ being the size of the context window) tokens and is tasked to predict the next one.

Training consists of two stages: unsupervised pre-training and supervised fine-tuning. The former is used to find a good initialization point, essentially teaching the model to imitate the corpora it is trained upon. This results in a model that will ramble on uncontrollably, just trying to complete the input sequence it's given to the best of its knowledge, however this will produce undefined behaviour. It is therefore necessary to fine-tune the model on a target task with supervision. Radford and Narasimhan (2018, p. 4) explain how the model can fine-tune directly on tasks like text classification, but how one for other tasks needs to convert structured inputs into ordered sequences because the pre-trained model was trained on contiguous sequences of text. In the case of ChatGPT, OpenAI used Reinforcement Learning from Human Feedback (RLHF) by employing a three-step strategy: first training using a supervised policy, then using trained reward models to rank alternative completions produced by ChatGPT models, before fine-tuning the model using Proximal Policy Optimization (PPO). This pipeline is then performed for several iterations until the model produces the desired behaviour (OpenAI, 2022).

### 2.1.3. The BERT Family

Bidirectional Encoder Representation from Transformers (BERT), introduced about four months after release of paper on the GPT architecture (Radford and Narasimhan, 2018), is a family of language models which was first introduced in 2018 and is designed to facilitate a wide range of downstream tasks (Devlin et al., 2019, p. 5). The BERT architecture consists of stacked bidirectional Transformer *encoders*. This makes BERT unsuitable for text generation, unlike the *decoder*-based GPT architecture. However, the self-attention in the encoder, in which tokens can *see* both past and future tokens, mechanism allows for training of deep bidirectional representations, facilitating a wide range of NLP tasks. The input sequence is transformed into embeddings (vector representations). These per-token embeddings include information about the meaning of the word itself, the meaning of the sentence/segment it belongs to, and the token's position in the full input. These embeddings then pass through a stack of Transformer encoders (12 and 24 for $\mathbf{BERT_{BASE}}$ and $\mathbf{BERT_{LARGE}}$, respectively), allowing the model to learn more complex patterns and of different granularities (token, sentence, document) (Devlin et al., 2019, p. 5).

In the BERT framework, there are two training steps, namely the pre-training and fine-tuning procedures. BERT is pre-trained on two NLP tasks. One is Masked Language Modelling (MLM), in which 15% of words are masked with the special `[MASK]` token and are left for the model to predict (Devlin et al., 2019, p. 4). The MLM task helps the model learn bidirectional representations. The second of the two unsupervised tasks used during pre-training is Next Sentence Prediction (NSP), where the special `[CLS]` token (found at the start of each tokenized sequence) is used to predict if a sentence `B` follows `A`. During this pre-training step, the input sequence looks like this:

`[CLS]` this is sentence A `[SEP]` and this is sentence B `[SEP]`

The `[CLS]` token is used to label sentence B as either `IsNext` or `NotNext`.

BERT is normally fine-tuned to specific downstream tasks by using the `[CLS]` token, which captures an aggregated representation of the input sequence. This vector representation can then be used as input to a classification layer for tasks like multi-label classification and regression.

### 2.1.4. Gemini

As of writing, Google's Gemini (Gemini Team and Google, 2023) (introduced on December 6th 2023) is the latest addition to the field of LLMs. Being fundamentally designed for multimodality, it is able to reason between text, images, video, audio, and code. Is released in different sizes: the Gemini Ultra, the Gemini Pro, and the Gemini Nano. The Gemini Ultra produces state-of-the-art performance on 30 of 32 widely-used academic benchmarks, and performs worse than OpenAI's GPT-4 on only one benchmark, the HellaSwag benchmark for common-sense reasoning for everyday tasks. Gemini Ultra outperforms GPT-4 on the MMLU benchmark, various reasoning benchmarks, and shows significant improvements in maths and code related benchmarks (Gemini Team and Google, 2023, p. 7). Gemini performs better than OpenAI's multimodal equivalent of GPT-4V in all (Gemini Team and Google, 2023, p. 12).

Like the GPT architecture, Gemini is built on top of Transformer decoders, and trained to accommodate various modalities, supporting interleaved sequences of text, image, audio, and video as inputs (Gemini Team and Google, 2023, pp. 3–4). This highlights one of the greatest strengths of the Transformer architecture, namely that it can be adapted for multiple modalities. Commonly, multimodal AI architectures consist of an ensemble of models, one for each given modality, with different representations that can be difficult to combine. The Transformer solves this issue and provides a common architecture that can be trained end-to-end.

### 2.1.5. Open-Source Alternatives

Seeing as the state-of-the-art models of today (GPT-4 and now, Google's Gemini) are all closed-source, it is important to note that there are viable open-source alternatives out there that. This section lists the most prominent ones.

#### LLaMA 2

Perhaps the most famous open-source LLM, Meta AI's LLaMA 2 is a powerful family of pre-trained and fine-tuned LLMs that outperformed open-source chat models on most benchmarks Touvron et al. (2023). It also shows great results in terms of safety, even outperforming the closed-source ChatGPT-0301 model. The training process of LLaMA is similar to that of the GPT model (see subsection 2.1.2). Pre-training is performed using an optimized auto-regressive transformer which is trained from a large corpus of unstructured data. It is the fine-tuned using various alignment techniques, and the authors also share anew technique they call Ghost Attentions, which aims to control dialogue flow over multiple turns. RLHF and Proximal Policy Optimization (PPO) are important techniques used to get the desired behaviour out of the model.

Many flavours of LLaMA have been trained. Most notably is, Code LLaMA (Rozière et al., 2023), a family of LLMs fine-tuned for code, scoring at 53% and 55% on HumanEval and MBPP, respectively. Vicuna-13B is another example, which is a LLaMA model fine-tuned on user-shared conversations collected from ShareGPT (a website where users can upload ChatGPT conversations). At its release in March 2023, it achives more than 90% quality of OpenAI ChatGPT and Google Bard, while also outperforming the original LLaMA model.

#### Mistral

Mistral 7B (AI, 2023) is another open-source LLM, claiming to be "the most powerful language model for its size to date". Being a 7.3B parameter model, it is quite small compared to other models, such as the 1.76T parameter GPT-4 model. Mistral 7B outperforms the LLaMA 2 13B on all benchmarks, and approaches CodeLlama 7B performance on code.

#### Orca 2

Orca 2, developed at Microsoft, is an open-source LLM with exceptional step-by-step reasoning capabilities. It is based upon LLaMA 2 is fine-tuned on curated training data from a more capable teacher model like GPT-4. Evaluation results shows that the Orca-2-13B model surpasses models of the same size, and that it is competitive with models 5-10x larger, exceeding the performance of LLaMA-2-Chat70B and performing comparably to ChatGPT on reasoning tasks (Mitra et al., 2023, pp. 11–12). It does have some limitations, most notably in terms of potential of misuse due to the lack of suitable safeguards like RLHF training (Mitra et al., 2023, p. 21).

## 2.2. Large Language Model Providers

The AI community called Hugging Face[1] is the most widely used hub for open-source and open-science machine learning. It is used by community members and also large commercial actors like Google Cloud and Meta AI to contribute the world open-source AI. Hugging Face's `transformers` Python library[2] provides an interface with their hub, which at the time of writing has 177k stars on GitHub.

OpenAI is one of the leading providers of LLMs, most prominently through their GPT series. While originally formed to be an open-source company, their models are now closed-source and their GPT models are only available through their own API, which are paid services. It should be noted, however, that they still are great contributors to open-source having uploaded a range of speech detection and diffusion models to Hugging Face. Microsoft Azure, the main partner to OpenAI, provides APIs for OpenAI's language models including the GPT-4, GPT-3.5-Turbo and Embeddings model series. Unlike OpenAI's own APIs, Microsoft Azure also offers private networking, regional availability, and responsible AI content filtering[3]. Also, by using Microsoft Azure's API, LLM inputs and outputs do not become available to OpenAI, which is not the case otherwise[4].

Aforementioned Google Cloud is another one of the largest LLM providers. They provide a wide range of services, among these the Vertex AI platform[5]. Vertex AI provides access to foundational models through APIs through their Model Garden platform which supports first-party, open-source, and third-party models. Vertex AI also provides experimental prompt design and fine-tuning services. Like the OpenAI APIs, Vertex AI is a paid service.

Anthropic[6] is a closed-source LLM provider that has grown rapidly in size since their launch in 2021, and focus on creating safer, steerable, and more reliable models. Their flagship model is Claude 2.1, which has a large context window of 200,000 tokens which enables users to upload large technical documentation that the model can keep in its memory when creating responses. Anthropic lists complex reasoning, creativity, and coding as Claude's major strengths. Perhaps their most famous model is GPT-J, which is an auto-regressive model in the style of GPT-3, aimed at the English language.

There are numerous open-source alternatives to the closed-source OpenAI and Google resources. Meta AI[7] is a big contributor to open-source AI, most famously through Pytorch—a popular deep learning framework—and their LLaMA models (see subsection 2.1.5). Eleuther AI[8] is an open-source centred company with aim to "increase transparency and reduce potential harms from emerging AI technologies". As such, they

[1] https://huggingface.co/
[2] https://github.com/huggingface/transformers
[3] https://learn.microsoft.com/en-us/azure/ai-services/openai/overview?source=recommendations
[4] https://learn.microsoft.com/en-us/legal/cognitive-services/openai/data-privacy
[5] https://cloud.google.com/vertex-ai/
[6] https://www.anthropic.com/
[7] https://ai.meta.com/
[8] https://www.eleuther.ai/

have released a range of trained LLMs along with the codebases used to train these.

## 2.3. Geospatial Standards

It is important for GIS-centred applications of any kind to support modern geospaial standards. Subsections 2.3.1 and 2.3.2 will provide a brief overview of the standardization work that has been and is being done on the international scene and in Norway, respectively.

### 2.3.1. International Standardization Work

International standardization work aims to unify the various geospatial standards found in different countries. The OGC standards and the STAC API standard are examples of such work.

#### OGC Api Standard

The Open Geospatial Consortium (OGC) API Standards serve as the glue in the field of GIS, paving the way for interoperability and data exchange between diverse systems. Leveraging common web protocols like HTML and supporting multiple data formats including JSON, GML, and HTML. The OGC API standard provides a modular architecture consisting of a core specification and various extensions. This modularity allows for flexibility, enabling users to customize their services according to specific needs. According to their webpages, they provide 80 different standards, each for a specific geospatial purpose. Notable examples are 3D Tiles, CityGML, GeoTiff, and OGC API - Features (OGC, 2023).

OGC API Standards function as modern replacements to older standards like WMS and WFS, and presents an evolved and more adaptable framework for spatial data operations, setting the stage for future innovations in the GIS domain.

#### STAC Api Standard

The SpatioTemporal Asset Catalog (STAC) API is a standardized way to expose collections of spatial temporal data for online search and discovery. Built upon a JSON core, it aims to be a uniform and flexible environment from which developers can customize the API infrastructure to their domain. STAC API provides a powerful query language that allows users to search by various parameters like time, location, and keywords, making widely applicable. The STAC community has also defined specification in order remove the complexity associated with having to create unique pipelines when consuming different spatial-temporary collection. The significance of the STAC API lies in its ability to democratize access to large volumes of geospatial data. By offering a common standard for data cataloguing and discovery, it reduces the barriers that often exist due to incompatible data formats. Developers or GIS professionals can take advantage of this through built-in tooling in QGIS, a desktop GIS for viewing, editing, and analysing spatial data, or through third-party packages in the Python and R programming languages. The API is also accessible through the command line interface when using GDAL (*STAC Tutorials* n.d.).

As OGC board member Chris Holmes puts it: "The STAC API implements and extends the OGC API — Features standard, and our shared goal is for STAC API to become a full OGC standard." (Holmes, 2021).

### 2.3.2. Norwegian Standardization Work

Geospatial standardization work has been on the agenda of Norwegian governing powers for decades and have materialized in frameworks/collaborations like Geovekst and Norge digitalt, as wells as the SOSI file format. Subsection 2.3.2 will delve into the work that has been done and what is expected for the future. The reasoning for the conclusion of this section is that the constraints of this specialization project is set by the Norwegian borders, and thus it is important to be aware of the standards that apply.

#### SOSI

Samordnet Opplegg for Stedfestet Informasjon (SOSI) is a Norwegian file format for storing and exchanging geospatial data. It was first introduced in 1987 and has since approached international standards, the most important arenas currently being ISO/TC 211 and OGC (Mardal et al., 2015). SOSI is the adopted Norwegian standard for creating and delivering digital geographic data, administered by the Norwegian Mapping Authority (Statens kartverk) (Mæhlum and Rød, 2023).

In a SOSI dataset, terrain points, lines, and polygons are represented by their coordinates and classified into various object types according to the SOSI object catalog standard. However, there are few GIS systems that can read SOSI data directly, so data in SOSI format usually needs to be converted to another GIS-readable data format (Mæhlum and Rød, 2023).

#### Geovekst

Geovekst is a Norwegian initiative that aims for collaborative collection, distribution, management, and maintenance of geospatial information. It was established in 1992, and is a partnership between national, regional, and local government bodies, as well as several private companies[9]. The primary goal associated with Geovekst is to “collaborate to secure updated Geovekst data to help solve parts of the parties’ societal missions” (The Norwegian Mapping Authority, 2023, p. 5). The collaboration is to ensure that geographical data is collected *once*, conforming to *one* standard, maintained in *one* place and used by *many*. Responsibilities and costs are shared among the parties of the collaboration.

The most important contribution of Geovekst is Felles KartdataBase (FKB), a series of very detailed Norwegian mapping datasets , serving as rich resources for both public and

[9] https://www.kartverket.no/geodataarbeid/geovekst

private sectors. The datasets are obtained through a variety of data sources, including aerial photographs, laser scans, and manual mapping. Geovekst also provides airborne surveys in emergency situations, when the speed of surveying is important[10].

**Norge digitalt**

Established in 2005, Norge Digitalt is a more recent framework compared to Geovekst and is often known as the National Spatial Data Infrastructure (NSDI). The NSDI involves governmental bodies (national, regional, and municipal), but also educational and research institutions and companies with responsibilities on a nation-wide scale; examples include Telenor and local and regional energy companies (Norge Digitalt, 2023, p. 6). The NSDI infrastructure is the sum of common standards and rules, geographical data and services related to these, in addition to tools and deals. It aims to coordinate geospatial activities in Norway, making it easier to discover, access, and use spatial data. The framework is coordinated by The Norwegian Mapping Society (Norge Digitalt, 2023).

## 2.4. User Groups

There are several user groups that could take advantage of an AI-based agent with general geographic and GIS knowledge. Questions can span from simple retrieval questions such as "How many people live in Trondheim" and "How long is the drive from Oslo to Bergen?", to more complicated questions that require problem-solving abilities and reasoning. While it is difficult to obtain dataset over common queries, Kumar (2023), creater of the chatbot app Pocket AI[11], shared a dataset of ~13k user queries from his app along with classifications of these. Salient categories were:

- "task oriented" (23.1%)
- "informational" (20.2%)
- "social" (16.2%)
- "personal advice and self-improvement" (13.1%)

The main takeaway from these numbers is that the main motivation for use is productivity.

This aligns with the results of Skjuve et al. (2023) from their questionnaire-based study performed in late January 2023, about three months after its release (OpenAI, 2023). The goal with the study was to find out *why* people use ChatGPT. They found that most participants (55%) are motivated by productivity, and specifically applying it for routine tasks, information retrieval, text generation and writing support, and software development (Skjuve et al., 2023, pp. 17–21). Table 2.1 shows all categories and their frequencies. There were 197 samples in total, and more than one category could be

[10] https://www.kartverket.no/geodataarbeid/geovekst/datafangst-i-krise
[11] https://github.com/varunon9/pocket-ai

assigned to each sample. It is worth noting that the study is likely to have included early adopters, and might therefore make the results less representative for the time at which this report is written (13th December 2023), now that use patterns have become more established (Skjuve et al., 2023, p. 37).

Table 2.1.: Categories and frequency of ChapGPT usage (Skjuve et al., 2023, pp. 16–17).

| **Category** | % **(n)** |
|---|---:|
| Productivity | 55% (109) |
| Novelty | 51% (101) |
| Fun and amusement | 20% (41) |
| Creative work | 18% (34) |
| Social interaction and support | 9% (18) |
| Other | 7% (15) |

Given that the main reason people use conversational AI is for productivity, whether in a professional, academic, or personal context, such technology could be highly beneficial in a geospatial setting. 67 out of the 197 participants in Skjuve et al. (2023, p. 18) highlighted "ChatGPT's ability to understand complex queries" and that it is "efficient in alleviating the need to experiment with different phrasings of the query", as is often needed when 'Googling' for an answer to a specific question. This ease of information retrieval, along with it's problem-solving abilities (Skjuve et al., 2023, p. 20), could also make conversational AIs highly relevant for geospatial purposes, and in the field of GIS.

One could imagine a range of potential user groups that could benefit from such an artificial, and spatially aware, companion. A couple of suggestions user groups are presented in Figure 2.1. Perhaps the most obvious one is that of the GIS professionals. While a AI-based GIS agent more capable than the average GIS professional currently far from becoming a reality, such an agent could help suggest strategies of solving a particular problem using the input data available and the end goal, or it could help solve mundane tasks in an automated way in order to allow the GIS professional to allocate more time to creative and complex tasks.

Closely related is city planners. Though often less knowledgeable in the field of GIS, they are increasingly dependent upon geospatial analysis in order to make informed decisions. Having an easy-to-use GIS ready at any moment could prove both time- and -cost-saving. The same goes for business analysts and people involved in academia. The *time* variable is especially important to the emergency services. At the impact of a natural disaster like a flood or forest fire, geospatial analysis could prove lifesaving. Having powerful geospatial knowledge even in the absence of a GIS professional is therefore important in order to focus resources to areas where the situation is most pressing.

An AI agent with geospatial awareness could also be useful to the general public. One may want to find suitable biking routes or good hills to do interval running in, or to know what areas are prone to flooding, when buying a house. Most people do not have the knowledge or time to perform such analyses themselves, so an automated AI-based agent could prove useful here.

## 2.5. Ethical and Privacy Concerns

Although an AI-based GIS agent could prove powerful, there are some ehtical pitfalls in terms of regulations and privacy concerns. If such an agent is to make decision on behalf of humans it is important that one can hold someone accountable in the case where something goes wrong. If an AI-based agent is tasked to lead firefighters to the tactically optimal locations in order to quench the fire, but misleads them and traps them inside the fire, who is then to blame? Sparrow (2007) provides an interesting angle on this issue, though related the role of artificially intelligent robots in modern warfare.

Furthermore, it is important that it is impossible to use the AI for immoral purposes, such as finding the optimal location in which to dissolve poison into people's drinking water. Work has been done with newer LLMs to prevent them from producing dangerous, misinformed, or toxic text—it is widely discussed in the technical report of the latest GPT model (OpenAI, 2023, pp. 11–14)—but there have been cases where these issues haven't been considered well enough[12].

The General Data Protection Regulation (GDPR) entered into applicability in the European Union in 25th of May 2018, and although not a member of the European Union, Norway incorporated the GDPR in July the same year due to its membership of the European Economic Area. The GDPR imposes stricter regulations about data privacy, meaning people have more control over their personal data, and that businesses get a level playing field in terms of what customer information is available (Datatilsynet, n.d.). With data privacy being more relevant than ever in this information age we find ourselves in, we must make certain that AI-based GIS agents are unable to access and spread information of private or sensitive character, even if such information has become publicly available by accident—as was the case when the Norwegian Broadcasting Corporation (NRK) was able to (legally) obtain accuracy geolocations of central people in the Norwegian army from a commercial London-based company[13].

[12] The LaMA-based Alpaca model, developed at Stanford University, was taken offline after being shown to produce misinformation, toxic text: https://www.theregister.com/2023/03/21/stanford_ai_alpaca_taken_offline/

[13] Link to news article in Norwegian: https://www.nrk.no/norge/xl/norske-offiserer-og-soldater-avslort-av-mobilen-1.14890424

# 3. Related Work

The related works are divided into three sections, each being relevant to the project in different ways. Section 3.1 is the most obviously relevant section, discussing works in which LLMs were employed to perform tasks in the geospatial realm. Section 3.2 delves into different prompt engineering and planning strategies that could be useful to make an autonomous GIS agent perform better and more reliably. Section 3.3 discusses Retrieval Augmented Generation (RAG), that is, how one can provide an autonomous LLM-based agent with externaFcontributionsl tooling and up-to-date information. Weng (2023) provides a good summary of techniques relevant to section 3.2 and section 3.3.

## 3.1. GIS with LLMs

A substantial body of work have been done in recent years to assess the geospatial knowledge of LLMs, and how they can be fine-tuned or embedded into frameworks to serve downstream tasks.

### 3.1.1. Taking the Temperature on GIS with LLMs on Social Media

The search term "LLM GIS" on Twitter/X shows various ways that people are using LLMs for GIS-related tasks. One user praises the use of ChatGPT to "extract and categorize data from unstructured text", sharing a video from an ESRI conference[1]. Twitter user Zeke Hausfather shares the discovery that "GPT4 now supports processing netCDF files and other geospatial data, as well as some pretty amazing visualization"[2]. Arpit Gupta shares a summary of a paper on generative regulatory measurement on Twitter/X, where he explains how they have utilized LLMs to decode and interpret status updates and administrative documents, including, for instance, mapping zoning and housing regulations for the suburbs of Chicago[3]. Yu Zhao speculate in the effectiveness of smaller LLMs fine-tuned on domain-specific knowledge for GIS or remote sensing[4], an interest other users share[5,6].

Swapping out "LLM" with "ChatGPT" gave more results. One user shows you using ChatGPT with tabular geographical data can increase productivity[7]. Other users

[1] https://twitter.com/mildthing99/status/1658507921234296833
[2] https://twitter.com/mildthing99/status/1658507921234296833
[3] https://twitter.com/arpitrage/status/1723033894801309893
[4] https://twitter.com/zhaoyutim/status/1651233975946321920
[5] https://twitter.com/zhaoyutim/status/1651233975946321920
[6] https://twitter.com/DougButdorf/status/1670938318979121152
[7] https://twitter.com/BooneLovesVideo/status/1617479222724857856

show how they use ChatGPT for entertainment or as an educational tool in a GIS context[8,9,10,11,12]. This appears to be the primary method by which people utilize ChatGPT, and it seems to offer mostly adequate responses. Another user highlights ChatGPT's built-in geographical context, using it to get GeoJSON polygons for a specified are directly[13].

On YouTube, the search term "ChatGPT GIS" yield a range of relevant responses. Several videos display how ChatGPT can be used to create Python code for GIS-related purposes. Examples were found of users highlighting ChatGPT's abilities to generate Python code to manipulate geospatial files, perform analysis, and visualize, using Python libraries like GeoPandas and Folium[14,15,16]. Other users show how uploading geospatial files into ChatGPT using Code Interpreter can be an efficient workflow[17]. Some user demonstrate the QChatGPT[18] plugin to QGIS, which is a plugin integration between QGIS and the OpenAI API. QChatGPT does not seem to have any context of the current QGIS project the user is working on, but appears to have sparked some excitement among certain users, seeing how LLMs can assist them in their daily work as GIS professionals[19,20,21]. One user shows an application with ChatGPT integration that can generate SQL code and visualize geospatial data[22]. A recurring user shows how one can use LangChain and its SQL database plugins to "unlock ChatGPT's potential"[23].

Lastly, the "FME Channel" released a recording of a webinar where they show how GPT-3 is being used in FME Data Integration Workflows[24]. They highlight how the OpenAI API allows for easy and automated no-code API to API workflows. On the FME Community website, an article writes about the `OpenAICompletionsConnector` and `OpenAIImageGenerator` transformers in FME[25]. They list use cases such as, running data through the AI for analysis, generation of reports and summaries, generating scripts or SQL for use in a data integration workflow, and automatic generation of images based on a dynamic input.

[8] https://twitter.com/briankingery87/status/1631365717269307394
[9] https://twitter.com/burdGIS/status/1614630141858316288
[10] https://twitter.com/__jsolly/status/1652867118797590528
[11] https://twitter.com/wanjohikibui/status/1628282272548806657
[12] https://twitter.com/GeoWithJustin/status/1641155652759199744
[13] https://twitter.com/at_dot_Py/status/1649985754800730112/
[14] https://www.youtube.com/watch?v=QDf-zc81NSE&t=1707s&ab_channel=GeoDeltaLabs
[15] https://www.youtube.com/watch?v=iNHQgLw7qZc&ab_channel=GeoDeltaLabs
[16] https://www.youtube.com/watch?v=BK2IzZZZC-k&ab_channel=MattForrest
[17] https://www.youtube.com/watch?v=dgzWLBYswh0&ab_channel=MiningGeologist
[18] https://plugins.qgis.org/plugins/QChatGPT/
[19] https://www.youtube.com/watch?v=zUZs4GsDk6I&ab_channel=GISWorld
[20] https://www.youtube.com/watch?v=eEkVTUS8Qtc&ab_channel=HansvanderKwast
[21] https://www.youtube.com/watch?v=Tc-hHaDqoxY&ab_channel=DEVICKSGEOSPATIALCO.
[22] https://www.youtube.com/watch?v=gaA46aaWDuc&ab_channel=GeospatialWorld
[23] https://www.youtube.com/watch?v=FoGm7d0paIo&t=1190s&ab_channel=MattForrest
[24] https://www.youtube.com/watch?v=94ZDhgW8yMY&ab_channel=FMEChannel
[25] https://community.safe.com/s/article/Tutorial-Getting-Started-with-OpenAI-in-FME

### 3.1.2. Geospatial Context in LLMs

Scherrer and Ljubešić (2020) were able to show that BERT can be fine-tuned to accurately predict geolocations from textual input, by winning a shared task on predicting geolocations from Twitter/Jodel messages in a workshop in 2020 (Gaman et al., 2020). By converting the task into a double regression problem, where they predicted latitude/-longitude pairs from the output `[CLS]` representation of BERT models. For a subtask on a Swiss Jodel dataset, they were able to achieve a median distance of 15.72 km from the ground truth, showing that LLMs can be trained correlate lingual features and geolocations.

Roberts et al. (2023) investigated extent of GPT-4's geospatial awareness through a set of case studies with increasing difficulties, starting with general factual tasks and finishing with complex questions such as generating country outlines and travel networks. The authors find that GPT-4 is "skilful at solving a variety of application-centric tasks", almost having the ability to "see", despite being a language model and thus only being able to interface with the world through sequenced, textual input. Examples include its ability to perform as a travel assistant in providing itinerary suggestions for a trip when provided with requirements, and its ability to provide start and end locations bird migrations generally correct, and in some cases highly accurate. While it becomes obvious that a lot of geospatial context have been embedded within the model during the vast pre-training, the question whether this is memorization or reasoning is a central one. The authors suggest that variability of tasks in their experiments deems it unlikely that it is all memorization, but they say that some things appear to be memorized.

Mooney et al. (2023) examined the performance of ChatGPT in a Geographic information system (GIS) exam, aiming to assess its ability to grasp various geospatial concepts, highlighting its capabilities and limitations. Experiments were conducted on GPT-3.5 and GPT-4, which delivered performances equivalent to grades of D and B+, respectively. Additional experiments were conducted for more specialized areas of GIS, including True/False questions about spatial analysis, and simple tasks in applied GIS workflow. Experiments on the latter showed that ChatGPT-4 was able to correctly answer a relatively complex GIS tasks involving seven different datasets, requiring seven steps in order to obtain a perfect score. Generally, ChatGPT-4 outperformed ChatGPT-3.5 in all tasks. While clearly powerful, the authors highlight a range of limitations, among which the multimodal nature of GIS, which would hinder a straightforward application of existing models.

Unlu (2023) discussed importance of enabling LLMs to recognize and interpret geospatial data, and how OpenStreetMap (OSM) can play an important role in offering LLMs linguistic access to vast cartographic datasets. He exemplifies this claim through a proof of concept in which he performs small-scale fine-tuning on an LLM with 1B parameters, using an artificial supervised datasets curated by the more capable ChatGPT 3.5-turbo, which functions as a teacher model, generating prompt-answer pairs for given preprompts. The fine-tuned model displays promising ability of answer questions about a location's attributes, allowing the user to inquire about things like tourist appeal and potential profitability of businesses in the vicinity of the given location. Unlu emphasizes the method's strengths for small datasets and minimal computational settings. The study also investigated the idea of using embeddings of the curated prepromts. Experimenting with average GLOVE embeddings, he showed that the latent structure of verbal descriptions of OSM data can yield insightful patters. This, he argues, can prove useful when creating Retrieval Augmented Generation (RAG) applications aimed at allowing users to retrieve geospatial information in a prompt-based manner.

### 3.1.3. Autonomous GIS

Z. Li and Ning (2023) states that "autonomous GIS will need to achieve five autonomous goals: self-generating, self-organizing, self-verifying, self-executing, and self-growing.", and provide a "divide-and-conquer"-based method to address some of these goals. Furthermore, they propose a simple trial-and-error approach to addressing the self-verifying goal. They also highlight need of a memory system in a mature Large Language Model (LLM)-based Geographic information system (GIS) system, referring to the use of vector databases in autonomous agents like AutoGPT (Richard, 2023). Even with its shortages, the solution that (Z. Li and Ning, 2023) provide, called LLM-Geo, is able to solve provide good solutions in various case studies by providing executable assemblies in a Python environment when provided with URLs to relevant data sets, along with a user-specified query.

Zhang et al. (2023) use the LangChain framework (Chase, 2022) in order to combine different GIS tools in a sequence in order to solve different sub-goals, and focuses on using the semantic understanding and reasoning abilities of LLMs like (e.g., ChatGPT) to call externally defined tools, employing the LLM as an agent or controller. The authors take great inspiration from the AutoGPT framework (Richard, 2023). The externally defined tools are described (manually) by its name and description. Said description contains information about the input parameters and output types of the tools/functions. Tools are defined for geospatial data collection, data processing and analysis, and data visualization. The effectiveness of the system is showcased in four case studies.

Qi et al. (2023) discuss how LLMs can be used in spatial data management, facilitating a system that can learn from both structured and unstructured data, the latter of which is possibly the greatest strength of modern LLMs. They highlight the opportunity that LLMs provide in reducing the barrier to information retrieval for the general public, and discuss how these strengths can be used in spatial data management by leveraging a spatial database system trained from both structured and unstructured data, allowing for seamless access to spatial knowledge, also for those with little or no expertise in querying a spatial database. With that in mind, they envisage to use *machine learning models as a spatial database* (MaaSDB), which when trained on structured and unstructured spatial data can *generate query answers directly* instead of retrieving data from tables, the latter of which has been the most common way of using machine learning in database query processing. From conducting preliminary studies they present a system of LLM-based system of query analysers, query plan generators, and a query result generators to handle natural language user queries. They propose a Generative Adverserial Network (GAN)-based model to generate tabular data, seeing if such a model can remember the key characteristics of the data. Such a model will have a *generator G* that will produce a record a *discriminator D* that will classify whether the generated record resembles a real record. The results of this approach is promising, and further prompt-based test performed on ChatGPT demonstrates its potential to learn spatial knowledge and answer queries. While potentially powerful, they highlight a range of challenges of implementing their proposed system, such as hallucination, the limited availability of structured spatial data, generalizability issues, and the problem of updating the trained models when the underlying data changes.

## 3.2. Prompt Engineering and Planning Strategies

Large Language Models have shown great abilities in problem-solving and decision-making tasks, but generally struggle as they are presented with larger and more complex tasks. Also, seeing as they are pure stochastic machines, the output is seldom reproducible. While the temperature parameter of the GPT models help serve as a control mechanism for this randomness, it does not guarantee fully predictable text generation. These issues have lead people into investigation *prompt engineering* and various techniques for helping the models form plans when faced with large and complex tasks, both of which aim to guide the model into producing the desired response.

The *Chain of Thought* strategy (Wei et al., 2023) aimed at complex reasoning in Large Language Models showed that reasoning can emerge naturally from sufficiently large LLMs. *Chain-of-Though prompting* entails the inclusion of examples of chain of thought sequences, that is, examples of how one might reason about a given problem in order to get to the answer, into the prompt. The exemplars are categorized into the types of tasks they aim to solve. This, along with instructing the model to think "step by step", achieved a new state-of-the-art accuracy on the GSM8K benchmark of math word problems in early 2023.

The *Tree of Thoughts* strategy (Yao et al., 2023) is a more recent planning strategy aimed at problem-solving with Large Language Models, and addresses a common limitation of *vanilla* LLM problem-solving, which often lacks the ability to explore strategically. Generalizing over *Chain of Thought*, *Tree of Thoughts* allows the LLM to consider multiple different reasoning paths and to perform self-evaluation to decide the next course of action. *Tree of Thoughts* can be used with different search algorithms. The authors discuss breadth-first search and depth-first search, and leave more advanced ones for future work. Using the *Tree of Thoughs* strategy proved very effective on certain tasks that are near impossible for the state-of-the-art LLM of GPT-4, particularly in the mathematical reasoning challenge called "Game of 24".

Zhou et al. (2023) introduces a framework called Language Agent Tree Search (LATS) "that synergizes the capabilities of LLMs in planning, acting, and reasoning.". As of writing (October 30th, 2023), the LATS framework is the highest scoring model on the HumanEval benchmark (see subsection 3.4.2), demonstrating state-of-the-art performance on decision-making tasks in a range of diverse domains. LATS performs a sequence of operations in succession until the task at hand is solved. These are *selection, expansion,* *evaluation, simulation, backpropagation, and reflection.* Employing Monte Carlo Tree Search they enable the LLM-based to select the among $n$ sampled options while still exploring other promising alternatives, using a heuristic to rank alternatives. Though a shared space of thoughts and actions, the framework supports both reasoning and decision-making tasks. Observation and self-reflection abilities enables LATS to use external feedback, which proved valuable when testing the framework on different benchmarks, some of which were discussed in subsection 3.4.2.

## 3.3. Retrieval Augmented Generation and Frameworks

RAG is tightly interwoven with explainable AI, being a framework for retrieving facts from an external knowledge base to allow a LLM-based agent access to accurate up-to-date information (Martineau, 2023). A common problem when working with language models, especially those designed to be general-purpose, is hallucination; that is, when the model provides an answer that is completely wrong but in a very convincing manner. While progress is being made with newer models even the better ones, like GPT-4, gives an incorrect answer about 1 out 5 times, and even worse for certain categories of queries (for instance 'code' and 'business') (OpenAI, 2023, p. 10). Retrieval Augmented Generation can help mitigate this problem.

Lewis et al. (2020) show that RAG model with access to a non-parametric memory in the form of a dense vector index of Wikipedia, will generate more specific and factual responses compared to state-of-the-art parametric-only sequence-to-sequence models at the time of publishing their paper. Their model architecture is a combination of a pre-trained retriever and a pre-trained sequence-to-sequence generative model, which is fine-tuned end-to-end (Lewis et al., 2020, p. 2). The approach obtained state-of-the-art results on open-domain question answering (Lewis et al., 2020, pp. 5–6).

Shi et al. (2023) shows that a simple RAG architecture provide significant improvement over state-of-the-art parametric-only LLMs like GPT-3. Their RePlug framework works by retrieving documents and prepended these to a "black-box" LLM. They also propose training scheme to further improve the retrieval model with supervision signals from the black-box LLM. Training is done with an objective that prefers documents that improve the perplexity (see (3.1)) of the model. This approach shows promising results relative to the original black-box model (Shi et al., 2023, pp. 5–6).

### 3.3.1. LangChain

LangChain (Chase, 2022) is an open-source project that provides tooling that can be used to create autonomous AI agents. It is designed to help with prompt management and optimization, creating chains of calls to LLMs, data augmented generation, autonomous agent creation, and memory-related tasks.

Nascimento et al. (n.d.) experimented with using ChatGPT with LangChain for Natural Language Interfaces for Databases (NLIDBs), that is, allowing the querier of a database to use natural language queries such as "Give me locations of all churches in Trondheim along with a short description" instead of SQL queries. They saw promising results when using `SQLDatabaseChain`, which inspects database schemas, tables, and joins in the database one provides it with. Doing so also helps mitigate issues with exceeding the ChatGPT token limit, compared with passing entire schemas as prefaces to the prompt itself. However, while the method was able to answer 13/27 test queries correct, using keyword search tools along with ChatGPT proved significantly more applicable, answering 22 correct and only 5 wrong.

### 3.3.2. AutoGPT

Richard (2023) will try to split a task into subtasks and use the internet and other tools in an automatic loop to solve the task/subtasks. AutoGPT comes with ready-to-go code templates for various purposes, benchmarks for agent performance measurements, and UI and CLI tools to control and monitor agents. The AutoGPT project adopts the *Agent Protocol Agent Protocol* (n.d.), which is an OpenAPI specification v3 based protocol that provides a common interface for communicating with agents. This ensures compatibility with future applications, and is currently used for communication with the UI and CLI tools.

Firat and Kuleli (2023) performed an exploratory study to map different use cases and experiences of AutoGPT users. They found that content creation, such as making a podcast outline, is a common use case for AutoGPT-powered applications. Other applications include data summarization and information organization. The authors highlight limitations token limit and inefficiency. AutoGPT is known to have behave unreliable at times, and a common complaint is that it gets stuck in "reasoning loops"[26,27].

### 3.3.3. AutoGen and Microsoft Semantic Kernel

AutoGen and the Microsoft Semantic Kernel are both Microsoft aimed at creating autonomous AI-based agents. AutoGen Wu et al. (2023) is a generic framework that allows for multi-agent applications in which agents can converse with each other. The authors demonstrate the effectiveness of the approach in domains including mathematics, coding, and online decision-making. They highlight improved performance, reduced development code, and decreased manual burden for existing applications as the main benefits. It also allows for limiting of fixed back-and-forth interactions between the AI agent and the human user by allowing

Microsoft Semantic Kernel[28] is an SDK that functions as the brain of an autonomous agent and provides connectors to models and memory, and connects to triggers and actions. Maeda (2023) talks about how Semantic Kernel can be used to augment the abilities of AutoGen agents by providing it with hooks into the real world. These *hooks* can be native functions that written by the developer, or existing OpenAI/Semantic

[26] https://github.com/Significant-Gravitas/AutoGPT/discussions/1939
[27] https://github.com/Significant-Gravitas/AutoGPT/issues/1994
[28] https://github.com/microsoft/semantic-kernel

Kernel plugins, like the WebPages Plugin which fetches a given URL and returns the text found.

### 3.3.4. OpenAI Assistants API

## 3.4. Benchmarking and Evaluation of LLMs

Subsection 3.4.1 will address common evaluation metrics used during model development, while subsection 3.4.2 will present some common benchmarks used to compare the performance of different LLMs.

### 3.4.1. Evaluation Metrics

Having objective ways of evaluating the performance of a textual response is as important as it is difficult. Such evaluations often have subjective nature, and it is not immediately obvious how automate evaluation. This section presents common approaches for different objectives, and serves as inspiration for how an evaluation metric can be adapted for GIS-related purposes.

#### Human Evaluation

Human evaluation, though an obvious evaluation metric, can be powerful. Human evaluators can manually score generated text based on a range of criteria, including relevance, fluency, coherence, and overall impression. Human evaluation can be expensive and time-consuming, and researchers have therefore developed mathematical formulas for evaluation.

#### Perplexity

Perplexity is an evaluation metric suitable for autoregressive models, measuring the degree of uncertainty in predicting the next word in a sequence, based on the preceding words. It is essentially a way of evaluating a model's ability to predict uniformly among the tokens available in the corpus it is trained upon. This is done by calculating the negative average log-likelihood

$$\mathrm{PPL}(X) = \exp\left\{-\frac{1}{t}\sum_{i}\log p_\theta(x_i|x_{<i})\right\} \tag{3.1}$$

where $X = (x_0, x_1, \ldots, x_t)$ is the tokenized input sequence and $p_\theta(x_i|x_{<i})$ is the log-likelihood of token $x_i$ given the preceding tokens $x_( < i)$. A lower score indicates better performance. It is normally calculated using a sliding window strategy, where a fixed number $k$ preceding tokens $(x_{i-k-1}, x_{i-k}, \ldots, x_{i-1})$ are used to calculate the perplexity for token $x_i$ (Hugging Face, n.d.).

### BiLingual Evaluation Understudy (BLEU)

BiLingual Evaluation Understudy (BLEU) provides a quick, inexpensive, and language-independent method of automatic machine translation, allowing researchers to rapidly home in on effective modelling ideas (Papineni et al., 2002). BLEU shows the BLEU formula, which takes the geometric mean of the corpus' modified precision score and then multiplies it by an exponential brevity penalty factor. (3.2) shows the result when taking the log of the function, which makes the ranking behaviour more apparent (Papineni et al., 2002, p. 5).

$$\log \text{BLEU} = \min\left(1 - \frac{r}{c}, 0\right) + \sum_{n=1}^{N} w_n \log p_n \tag{3.2}$$

### Recall-Oriented Understudy (ROUGE)

The Recall-Oriented Understudy (ROUGE) metric, introduced by Lin (2004), aims to automatically determine the quality of a summary by comparing it to ground truth summaries produced by humans. (3.3) shows the ROUGE-N formula, which is the n-gram recall between a candidate summary and a set of the aforementioned ground truth summaries.

$$\text{ROUGE-N} = \frac{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{ReferenceSummaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)} \tag{3.3}$$

### Diversity

Diversity metrics aim to measure the variety and uniqueness of generated sequences. J. Li et al. (2016) proposed an objective function called Maximum Mutual Information (MMI), which seeks to guide sequence-to-sequence models into producing more diverse, interesting, and appropriate responses, as opposed to safe and commonplace ones. The parameters of MMI are chosen in order to maximize mutual information between the source sequence $S$ and the target sequence $T$

$$\hat{T} = \underset{T}{\operatorname{argmax}}\{(1-\lambda)\log p(T|S) + \lambda \log p(S|T)\} \tag{3.4}$$

where $\lambda$ serves as a weighting parameter. As the paper is from 2016, the authors only discuss the $p(Y|X)$ function in relation to the Long Short-Term Memory (LSTM) algorithm, but it can be applicable to contemporary language models as well.

Stasaski and Hearst (2022) propose three metrics which leverage the predictions of a Natural Language Inference (NLI) model, that is, a model which seeks to determine if one sentence entails, contradicts, or is neutral toward a second sentence (Stasaski and Hearst, 2022, p. 1). The *Baseline NLI Diversity* metric

$$\text{Baseline NLI Diversity} = \sum_{u_i, u_j \in u_1, \ldots, u_n} NLI_{score}(NLI_{pred}(NLI(u_i, u_j))) \tag{3.5}$$

where $NLI_{score}$ is 1, 0, or -1 if the sentence is deemed contradictory, neutral, or entails the other sentence, respectively. The two other metrics, the *Neutral NLI Diversity* and *Confidence NLI Diversity* differ only in how they define the $NLI_{score}$. Results from experiments show that using these can produce more diverse sets of responses, and that they can be used to investigate a model's ability to produce diverse responses (Stasaski and Hearst, 2022, p. 9).

### 3.4.2. Benchmarks

Benchmarks are standardized tests that aim to highlight the strengths and weaknesses of different Large Language Models, and serve as a non-biased way of comparing them. Benchmarks have been developed to assess different tasks, such as language understanding, general knowledge, arithmetic, and code generation. This section will introduce some common benchmarks.

HumanEval is a dataset of handwritten problems used to measure functional correctness for synthesizing programs for docstrings (Chen et al., 2021, pp. 2–4). Code generation is one of the most common use cases for LLMs and HumanEval is therefore arguably one of the more important benchmarks.

First introduced by Hendrycks et al. (2021) Multitask Language Understanding (MMLU) is a way of testing an Large Language Model's multitask accuracy, covering 57 tasks including mathematics, computer science, and others. MMLU a commonly used to highlight the general knowledge that is embedded within the model.

The BIG-Bench-Hard is another LLM benchmark created by Suzgun et al. (2022). It is a suite of 23 problems where prior language models were unable to exceed average human performance. The character of many of the BIG-Bench-Hard tasks, requires the rater to use multi-step reasoning, which has traditionally been hard for language models to apply.

HellaSwag (Zellers et al., 2019) is a benchmark designed to measure an LLM's ability to "finish your sentence". By developing a lengthy and complex dataset to see where the LLM starts producing "ridiculous" responses, the HellaSwag benchmark provides a way of testing a model's common-sense inference abilities.

API-Bank (M. Li et al., 2023) is a benchmark designed to evaluate an LLM's ability to use external tools (APIs). Through interview the author highlights two main requirements for a tool-augmented LLM (M. Li et al., 2023, p. 2). (1) *Few vs. Many APIs in API Pool.* With one a couple APIs in the API pool, one can possibly send the entire API schema with the prompt, simplifying request parameterization and response parsing. This becomes difficult as the number of APIs increases, and the token limit becomes a limiting factor. When this is the case, the LLM needs to reason about which APIs are relevant or not. (2) *Single vs. Several API calls per Turn.* Based on the user's preferences, one might want the LLM to perform several API requests at once, or one might want to gradually guide it through several steps.

# 4. Experiments and Results

Three experiments were conducted in this specialization project. Section 4.1 will explain how the experiments were conducted, and section 4.2.

## 4.1. Method: Experimental Plan and Setup

The experiments were conducted in order to answer the research questions listed in section 1.2. As chapter 3 shows, there have been a substantial body of work on mapping the geospatial abilities of LLMs like ChatGPT, and how these can be used to create larger frameworks for GIS purposes. However, no literature was found that specifically discussed how LLMs handle geospatial files like GML or shapefiles, or how LLMs can be used to access web APIs that conform to common geospatial API standards like the OGC API Features specification. The experiments seek to address these points, with experiment 1 focusing on RQ1 and ChatGPT's abilities of performing geospatial analysis using different file formats, while experiment 2 and 3 focus RQ2 and RQ3 and on its ability to access external web APIs.

### 4.1.1. Experiment 1: Testing ChatGPT's Ability to Perform Geospatial Analysis

Experiment 1 was performed to assess ChatGPT's ability to perform geospatial analysis, and is aimed at RQ1. The approach used is inspired by the work of Roberts et al. (2023) (see section 3.1), who did experiments with increasing difficult on GPT-4 to characterize what GPT-4 knows about the geographical world, highlighting both capabilities and limitations. Roberts et al. focused on GPT-4's general geospatial awareness, and were not concerned with GIS-related tasks. The reference to Roberts et al. (2023) will therefore be made when highlighting the somewhat surprising geospatial awareness abilities of GPT-4. The focus of this experiment will instead be on displaying its potential for use in the world of GIS. This will be achieved by constructing various tests that aim to reflect its GIS knowledge.

The experiment will use the Elveg 2.0 dataset (The Norwegian Mapping Authority, 2019), along with cadastral data. In order to assess ChatGPT's ability to read and understand different data formats, the data will be provided in both SOSI, GML, and GeoJSON format. Datasets for the first two formats were downloaded from https://geonorge.no, while the GeoJSON datasets were created using a custom Bash script which converts from GML to GeoJSON using the `ogr2ogr` program from GDAL. The Elveg 2.0 dataset contains a range of different layers for different types of geometries.

In order to simplify the experiments, only the layer named "Fartsgrense" (eng. "Speed limit") was used from Elveg 2.0.

Below are the questions that were asked, in rising order of predicted complexity:

1. “Provide a summary of the file contents, highlighting the file’s most salient features.”
2. “Provide a visual representation of the file contents.”
3. “Find the mean location of the building locations.”
4. “Extract all roads with a speed limit greater than or equal to 80 km/h.”
5. “Select all buildings located within 50 metes of a high-speed road (speed limit >= 80 km/h).”
6. “Find the area best suited for expansion to accommodate residential buildings.”

Some follow-up questions are added when needed, in order help the model understand the questions or when it stops and asks for permission to go forth with analysis. All conversations are saved in the project’s GitHub repository[1].

#### 4.1.2. Experiment 2: Comparing File Upload and API Calling in ChatGPT-4

Another important thing to test is the issue of providing ChatGPT with relevant files on which it can perform analysis. ChatGPT Plus users will have access a range of advanced features, including web browsing with Bing, Dall-E Image Generation, and Code Interpreter (or Advanced Data Analysis). The latter of these allows the user to manually upload files into the chat instance and perform advanced analyses on the contents of these, which is what was used to upload the datasets in experiment 1. While this is very powerful, having to manually upload files poses some limitations. A more flexible system should be capable of accessing web APIs in real time.

A dataset containing the border of Drammen Municipality was used to test compare ChatGPT’s ability to perform analyses on manually uploaded data, versus data handed through it from passing a URL address. The data conforms to the GeoJSON standard and contains a FeatureCollection object with a single Feature, namely the border. When file/URL has been provided, ChatGPT is simply asked to present a visual presentation of its contents.

The dataset is located under this web API: https://alenos-tester001.azurewebsites.net/. This example OGC API was created by Norkart’s Alexander Salveson Nossum for with the purpose of testing OGC API Features on Norwegian data. It was created using `pygeoapi`[2], which is a Python server implementation of the OGC API suite of standards. It allows for deployment of a RESTful OGC API endpoint using OpenAPI, GeoJSON, and HTML.

[1] https://github.com/oskarhlm/prosjektoppgave/tree/main/documents/ChatGPT_conversations
[2] https://pygeoapi.io/

| | SOSI | GML | GeoJSON | Shapefile |
|---|---|---|---|---|
| Summary | No | Yes | Partially | Partially |
| Plotting | - | When guided | When guided | Yes |
| Mean location | - | When guided | Yes | No |
| Filtering | - | No | Yes | Yes |
| Buffer + Intersect | - | No | No | No |
| Planning for expansion | - | No | Partially | No |

Table 4.1.: Overview of the ability of ChatGPT's Code Interpreter to handle various geospatial data formats

### 4.1.3. Experiment 3: Using ChatGPT's and LangChain to perform API Call

A final and third experiment was conducted to see if there are other, more programmatic ways of performing API calls using LLMs. The LangChain framework (Chase, 2022) was used to create OpenAI functions that can be called using Function Calling from OpenAI. The goal of the experiment was to produce the same plot as requested in experiment 2 using the same API endpoint.

One function was made to fetch the data from the API and save it to a temporary file on the machine from which the code is executed. Another function was made to load a GeoJSON file and plot the contents using the Matplotlib library. The hope is that, using LangChain's `AgentExecutor`, which allows reasoning and chaining responses from an LLM, along with the Function Calling abilities of the OpenAI GPT APIs, should make it possible for the agent to call these functions in the right order and with the correct arguments. The `gpt-4-1106-preview` model was used for this experiment.

## 4.2. Experimental Results

### 4.2.1. Results for Experiment 1

As Table 4.1 shows, ChatGPT's Code Interpreter is unable to read and write SOSI files. It was unable to manipulate the data directly and was also unable to convert the file into a more suitable format, failing to convert it to GeoJSON using GDAL's `ogr2ogr`. SOSI is therefore excluded from Table 4.1, which shows the test results on the different file formats.

Furthermore, ChatGPT's Code Interpreter did not manage to properly analyze the GML data without guidance. It created a parser that was difficult to use for further analysis. When guided into using the GeoPandas library, which can handle GML data, it managed to plot the contents and calculate a centroid. The buffering and intersection task "was interrupted due to its time-consuming nature", and it did not make an attempt at solving the planning task due to inability to analyze the GML files.

With the GeoJSON data, ChatGPT had difficulties reading the files and could not provide a good summary consistently. It *was* however able to plot the data, but that had to be done in separate responses for each of the two datasets. It was able to find the high-speed roads, but could not figure out which buildings were within a 50-meter buffer of these. However, when asked to plan for expansion to accommodate residential buildings, it managed to achieve a result close to what was expected in the "Buffer + Intersect" task. It accomplished this by creating a grid and figuring out which grid cells were within 50 meters of a high-speed road. While this did not extract a subset of the building points—which would have been the desired output—it had some minor value in terms of visualization (see Figure 4.1). Though the result is hardly useful, it is closer than in the GML attempt to what would be an acceptable response.

Using the shapefile formatted data, ChatGPT was able to produce a decent summary of the data, but the attribute names were cut off after about 10 characters. It was, however, able to produce quite good visual representations of both dataset, colouring the roads differently by their speed limits and the buildings by their building type. While it did manage to filter roads on speed limits, it could not calculate the mean location of the buildings. The buffer + intersection and planning tasks again proved too complex.

### 4.2.2. Results for Experiment 2

Figure 4.2a and Figure 4.2b shows the results when providing ChatGPT-4 with the GeoJSON for the outline of Drammen municipality using the file upload functionality and providing it and URL to an API endpoint, respectively. While the Code Interpreter handled the direct file upload with ease, it struggled when provided with the URL to the corresponding web API. When provided with the URL, its first response was that "there was a problem with establishing a connection to the website", after trying to process the request using Code Interpreter. When guided (twice) to try accessing the URL using its web browsing abilities, it was eventually able to read the data. In the subsequent prompt it was asked to provide a visual representation, but failed to do so successfully as Figure 4.2b shows. The reason for this was its decision to "truncate the dataset for brevity, using a subset of the full coordinate list" (see Listing 4.1).

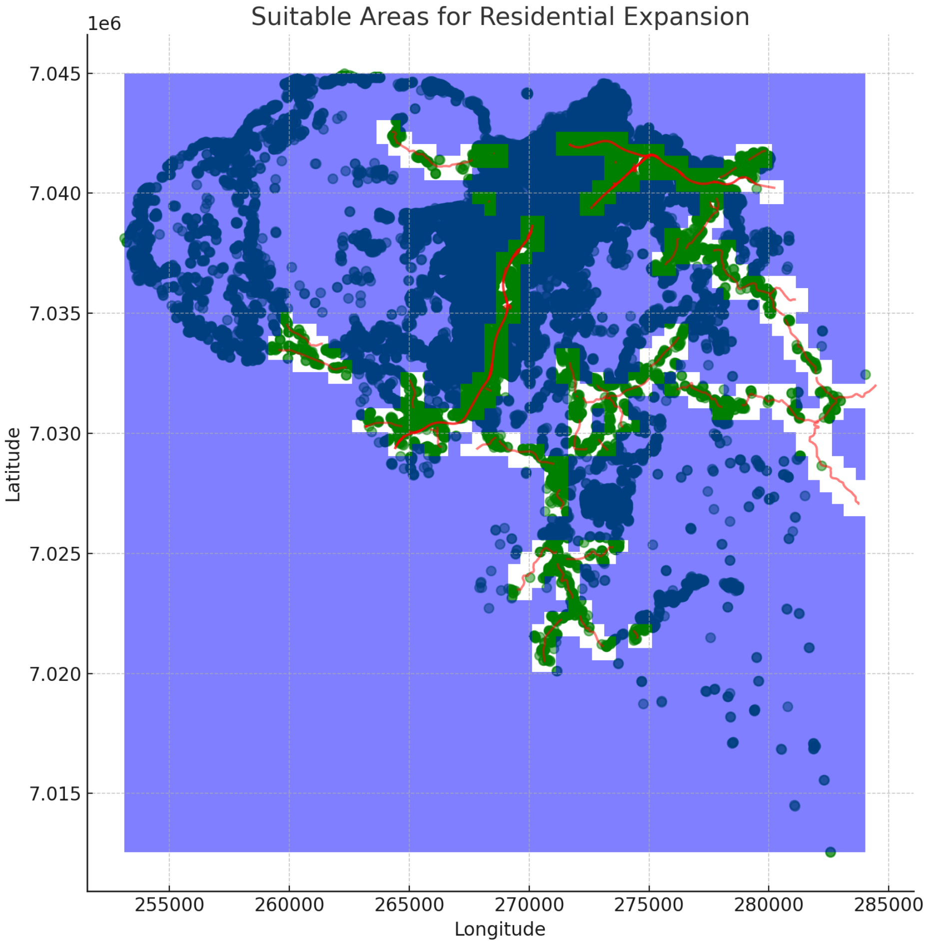
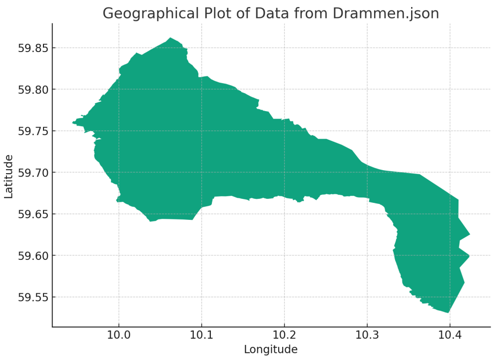
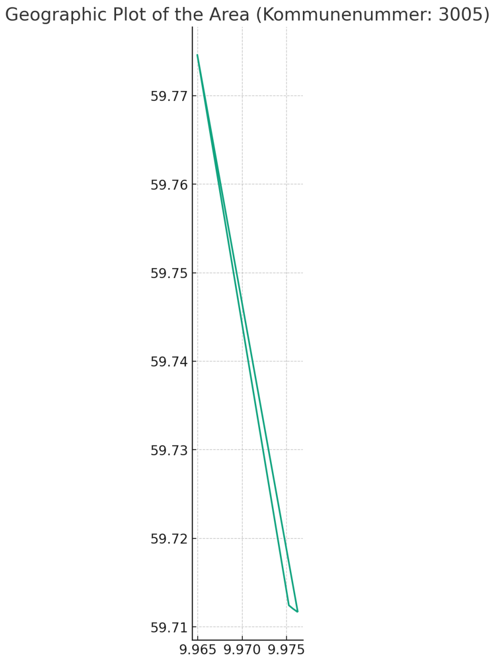



Figure 4.1.: The result of ChatGPT when asked to “Find the area best suited for expansion to accommodate residential buildings”, using provided GeoJSON datasets. Potentially suitable areas for residential expansion are depicted in blue.




(a) File upload




(b) Providing URL

Figure 4.2.: Comparison of the resulting outline of Drammen when using file upload (a) and providing an URL to an API endpoint (b) with ChatGPT-4

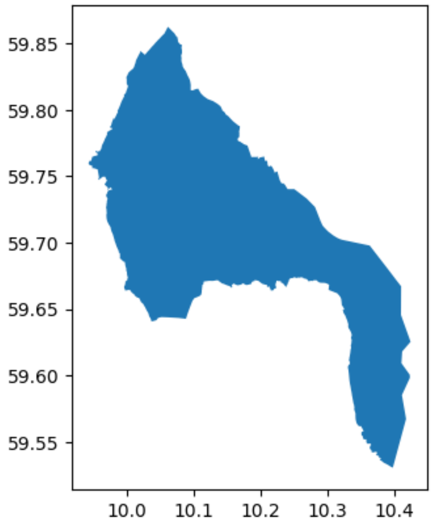



Figure 4.3.: Outline of Drammen using LangChain and OpenAI Functions

Listing 4.1: ChatGPT code that truncates coordinates

```
# ...

# Extracted coordinates from the JSON data
coordinates = [
    [[9.976278541184014, 59.71166107645171],
        [9.975715936016496, 59.71206324390201],
        [9.975270250797282, 59.71243147807226],
    # ... Truncated for brevity, using a subset of the full
        coordinate list
    [9.964929990238785, 59.774609947672644]]
]

# ...
```

### 4.2.3. Results for Experiment 3

The `AgentExecutor` with the `gpt-4-1106-preview` was able to call the two functions in the correct order and with the correct arguments. Figure 4.3 show the resulting plot. The notebook used for the experiment is available on the project GitHub repository[3].

[3]https://github.com/oskarhlm/prosjektoppgave/blob/main/src/python/examples/drammen_ogc_test/drammen_ogc_test.ipynb

# 5. Discussion and Conclusion

Sections section 5.1 and section 5.2 will discuss the experimental results (see section 4.2), and providing suggestions as to how the limitations highlighted by the experiments can be mitigated. Section 5.3 will conclude this specialization project report, and provide directives for future work on the subject of LLM-power GIS.

## 5.1. Using the In-Built ChatGPT Code Interpreter for Geospatial Analysis

When using ChatGPT's Code Interpreter with file uploads, it became apparent that it runs in a Linux environment, and that it uses a mounted drive in the `/mnt` director, which is used for temporarily mounted filesystems. From the initial experiments where the same data in different file formats was tested it tried to a GDAL command (`ogr2ogr -f "GeoJSON" {converted_geojson_path} {sosi_file_path}`) to perform a conversion from SOSI to GeoJSON, the latter of which is far easier to manipulate in a Python environment. This test failed, and the system's response was that "the `ogr2ogr` tool is not available in this environment".

This result was not very surprising, especially since the driver needed to read and write SOSI files—which is called *fyba* and is developed by The Norwegian Mapping Authority[1]—is almost certaintly not available in the standard Linux environment for ChatGPT's Code Interpreter. Seeing as the SOSI standard still is widely used for Norwegian geospatial purposes (though expected to be exchanged with the GML format in the future), it is important for an LLM-based GIS agent focused on the Norwegian market to be able to handle this file type.

The inability to manipulate the Linux environment using by the Code Interpreter clearly poses some limitations on the systems. A solution to the problem is to create a custom environment on a server and implement agent-like capabilities by other means (LangChain, AutoGPT, AutoGen, etc.). Having the agent run on an environment that we control ourselves gives us greater flexibility, and we can then allow the agent to access powerful GIS tooling, such as the GDAL library. This also allows us to avoid having to perform I/O on a mounted directory (in the `/mnt` directory), which can increase the speed of reads and writes.

[1] https://github.com/kartverket/fyba

## 5.2. Mitigating ChatGPT's Inability to Access Web APIs

As results of experiment 2 (see subsection 4.2.2) showed, ChatGPT-4 struggles when provided with URLs to external Web APIs, even when prompted to use its web browsing abilities and pairing them with its Code Interpreter. These issues are not present when using direct file upload, in which case the model appears to save the uploaded file in a temporary file directory in on a mounted drive in its Linux environment. While it can be hard to interpret the inner workings only from the code samples, it appears that it does not do this by default after fetching data from an external API. It appears from the results, in which it *truncates* the file and "store's" the content directly in code, thus attempting to store the entire file in the context window of the LLM. The context window of the ChatGPT-4 model is currently at 32,000 tokens (the new GPT-4 Turbo has a context length of 128,000 tokens). While these are of significant size, they are not meant to (or able to) store large file, and thus it becomes a limiting factor when the file contents grow large, which is not uncommon for geospatial files.

This is a significant limitation of using ChatGPT-4 out of the box, and one should therefore look into other ways of handling web requests and subsequent storing of the received data. Techniques within Retrieval Augmented Generation (RAG) can help here, and tools like LangChain or OpenAI's own "GPTs"[2] can help solve this issue.

## 5.3. Conclusion and Future Work

This specialization project has displayed an extensive literature research in the fields of Natural Language Processing, Large Language Models, and Geographic information system, along with two experiments trying to learn display strengths and limitations of LLMs when dealing with geospatial data in different file formats. The literature and experiments serve as a preliminary study to provide a better starting point when attempting to develop an LLM-based GIS agent.

With this being a specialization project that will transition into a larger master thesis, some points of discussion have been reserved for future work. Additionally, the task of developing a proof of concept has been assigned to the master thesis due to time constraints and the intention to acquire more knowledge before proceeding with development. The following sections will elaborate on potentially important issues that needs to be addressed when developing an LLM-based GIS agent.

### 5.3.1. Balancing Accuracy Against Performance and Costs

The ecosystem of agent frameworks and planning strategies to improve agent performance on complex tasks (discussed in Related Work), is a growing one. Different agents frameworks and planning strategies should be compared to see which are most viable for GIS work. Important considerations are the ability to destructure complex problems, the ability to take advantage of external tooling, and computational time. More complex

[2] https://openai.com/blog/introducing-gpts

planning strategies typically demand more interactions with LLM APIs, which can be expensive both in terms of computational time and cost (when using a monetized API like OpenAI's for GPT-4 and GPT-4).

Cleary (2023) did benchmarking of different LLMs on different providers. Important takeaways were that GPT-4 is about 6.3 times slower than GPT-3.5-Instruct, and that Azure has far lower latency in most cases for inference on GPT models. Such considerations are important when addressing usability of LLM-based applications, balancing accuracy against speed and costs. Using free open-source alternatives where possible is a good option to reduce costs.

### 5.3.2. Testing regime

In order to test the feasibility of different language models to serve as the brain of an autonomous GIS agent, a testing regime should be developed. In the examples of autonomous GIS agents described in the literature study of this report (see section 3.1), results have generally been presented in the form of case studies (Z. Li and Ning, 2023; Zhang et al., 2023). This type of qualitative testing is entirely appropriate to showcase the possibilities of the technologies but may be insufficient when comparing performance of different systems. In the latter case a quantitative approach would probably be preferable.

One idea is to create a test dataset which consists of inputs and corresponding desired outputs of typical GIS tasks. Inputs would in this case be natural language queries inputted by a mock user, and the output would be what you would expect a GIS professional to return when given the same tasks/queries. Inputs should reflect the varying level of GIS knowledge in the different user groups (see section 2.4). Outputs could be files with typical geospatial extensions (.shp, .geojson, .sosi, etc.), or they could adhere to API schemas specified by geospatial standards (see section 2.3).

While the inputs should be fairly simple to construct there are several questions to be answered in regard to the outputs:

- How does one evaluate the accuracy of the output?
- How should the AI agent respond when the user does not specify an output file format?
- How does one evaluate the usefulness of outputs to questions that should not return geospatial files, e.g. answers to general questions about geo-related subjects?

These are questions outside the scope of this specialization project. They will, however, be pursued in my master thesis.

### 5.3.3. Memory and Embeddings

Storing information for future use is important when developing LLM-based agents in order for it to produce consistent responses. Weng (2023) three different types of memory in human brains: (1) *Sensory Memory*, (2) *Short-Term Memory*, and (3)

*Long-Term Memory*. When translated to LLM we can think of *Sensory Memory* as learning embedding representation, *Short-Term Memory* as the memory contained within the limits of the context window of the Transformer, and *Long-Term Memory* as an external vector store that can be attended to by the agent at query time. Such a vector store/database would store the vector embeddings of the data contained within it, and allows for fast and accurate similarity search and retrieval based on the vector distance or similarity between the vector representations (evchaki, 2023). Weng (2023) lists some common approximate nearest neighbours algorithms for fast retrieval speeds, including Locality-Sensitive Hashing (LSH) and Facebook AI Similarity Search (FAISS).

Future work should expand on the work of Unlu (2023) (see section 3.1) and investigate if vector embeddings can be used for long-term storage of geographical data with textual description, or if a vector database can be used to efficiently retrieve relevant resources like APIs or other external tools based on their documentation/specification. Additionally, this documentation and the API specifications can be large is size, and the context length can become a limiting issue. Vector embeddings can help mitigate such issues. By splitting the documents into chunks and indexing them using vector embeddings, one can extract only the relevant parts and pass these to the LLM with the prompt.

# Bibliography

Agent Protocol. (n.d.). Retrieved December 11, 2023, from https://agentprotocol.ai/

AI, M. (2023). Mistral 7B. Retrieved December 7, 2023, from https://mistral.ai/news/announcing-mistral-7b/

Chase, H. (2022). LangChain. Retrieved October 5, 2023, from https://github.com/langchain-ai/langchain

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., Ray, A., Puri, R., Krueger, G., Petrov, M., Khlaaf, H., Sastry, G., Mishkin, P., Chan, B., Gray, S., Ryder, N., Pavlov, M., Power, A., Kaiser, L., Bavarian, M., Winter, C., Tillet, P., Such, F. P., Cummings, D., Plappert, M., Chantzis, F., Barnes, E., Herbert-Voss, A., Guss, W. H., Nichol, A., Paino, A., Tezak, N., Tang, J., Babuschkin, I., Balaji, S., Jain, S., Saunders, W., Hesse, C., Carr, A. N., Leike, J., Achiam, J., Misra, V., Morikawa, E., Radford, A., Knight, M., Brundage, M., Murati, M., Mayer, K., Welinder, P., McGrew, B., Amodei, D., McCandlish, S., Sutskever, I., & Zaremba, W. (2021). Evaluating Large Language Models Trained on Code. https://doi.org/10.48550/arXiv.2107.03374

Cleary, D. (2023). Latency Benchmarks and Comparisons for OpenAI, Azure, and Anthropic. Retrieved December 11, 2023, from https://medium.com/@dan_43009/latency-benchmarks-and-comparisons-for-openai-azure-and-anthropic-6f035f1acab6

Datatilsynet. (n.d.). General Data Protection Regulation. Retrieved December 5, 2023, from https://www.datatilsynet.no/en/regulations-and-tools/regulations/

Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. https://doi.org/10.48550/arXiv.1810.04805

evchaki. (2023). Vector Database. Retrieved December 11, 2023, from https://learn.microsoft.com/en-us/semantic-kernel/memories/vector-db

Fan, A., Gokkaya, B., Harman, M., Lyubarskiy, M., Sengupta, S., Yoo, S., & Zhang, J. M. (2023). Large Language Models for Software Engineering: Survey and Open Problems. https://doi.org/10.48550/arXiv.2310.03533

Firat, M., & Kuleli, S. (2023). What if GPT4 Became Autonomous: The Auto-GPT Project and Use Cases. *Journal of Emerging Computer Technologies*, *3*(1), 1–6. https://doi.org/10.57020/ject.1297961

Gaman, M., Hovy, D., Ionescu, R. T., Jauhiainen, H., Jauhiainen, T., Lindén, K., Ljubešić, N., Partanen, N., Purschke, C., Scherrer, Y., & Zampieri, M. (2020). A Report on the VarDial Evaluation Campaign 2020. *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, 1–14. Retrieved October 16, 2023, from https://aclanthology.org/2020.vardial-1.1

Gemini Team & Google. (2023). *Gemini: A Family of Highly Capable Multimodal Models* (tech. rep.). Retrieved December 7, 2023, from https://storage.googleapis.com/deepmind-media/gemini/gemini_1_report.pdf

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., & Steinhardt, J. (2021). Measuring Massive Multitask Language Understanding. https://doi.org/10.48550/arXiv.2009.03300

Holmes, C. (2021). SpatioTemporal Asset Catalogs and the Open Geospatial Consortium. Retrieved October 23, 2023, from https://medium.com/radiant-earth-insights/spatiotemporal-asset-catalogs-and-the-open-geospatial-consortium-659538dce5c7

Hugging Face. (n.d.). Perplexity of fixed-length models. Retrieved December 9, 2023, from https://huggingface.co/docs/transformers/perplexity

Kumar, V. (2023). What are people asking to ChatGPT? Retrieved October 25, 2023, from https://varunon9.medium.com/what-are-people-asking-to-chatgpt-f5f324a6cc27

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.-t., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *Advances in Neural Information Processing Systems*, *33*, 9459–9474. Retrieved December 6, 2023, from https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html

Li, J., Galley, M., Brockett, C., Gao, J., & Dolan, B. (2016). A Diversity-Promoting Objective Function for Neural Conversation Models. In K. Knight, A. Nenkova & O. Rambow (Eds.), *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 110–119). Association for Computational Linguistics. https://doi.org/10.18653/v1/N16-1014

Li, M., Zhao, Y., Yu, B., Song, F., Li, H., Yu, H., Li, Z., Huang, F., & Li, Y. (2023). API-Bank: A Comprehensive Benchmark for Tool-Augmented LLMs. https://doi.org/10.48550/arXiv.2304.08244

Li, Z., & Ning, H. (2023). Autonomous GIS: The next-generation AI-powered GIS. https://doi.org/10.48550/arXiv.2305.06453

Lin, C.-Y. (2004). ROUGE: A Package for Automatic Evaluation of Summaries. *Text Summarization Branches Out*, 74–81. Retrieved December 7, 2023, from https://aclanthology.org/W04-1013

Maeda, J. (2023). AutoGen Agents Meet Semantic Kernel. Retrieved December 11, 2023, from https://devblogs.microsoft.com/semantic-kernel/autogen-agents-meet-semantic-kernel/

Mæhlum, L., & Rød, J. K. (2023). SOSI. *Store norske leksikon*. Retrieved October 23, 2023, from https://snl.no/SOSI

Mardal, G., Borreb, M., Christensen, L., Jetlund, K., Ryghaug, P., & Hokstad, I. (2015). Nasjonal strategi for videreutvikling av SOSI.

Martineau, K. (2023). What is retrieval-augmented generation? Retrieved October 6, 2023, from https://research.ibm.com/blog/retrieval-augmented-generation-RAG

Mitra, A., Del Corro, L., Mahajan, S., Codas, A., Simoes, C., Agarwal, S., Chen, X., Razdaibiedina, A., Jones, E., Aggarwal, K., Palangi, H., Zheng, G., Rosset, C., Khanpour, H., & Awadallah, A. (2023). Orca 2: Teaching Small Language Models How to Reason. Retrieved December 12, 2023, from http://arxiv.org/abs/2311.11045

Mooney, P., Cui, W., Guan, B., & Juhász, L. (2023). *Towards Understanding the Geospatial Skills of ChatGPT: Taking a Geographic Information Systems (GIS) Exam*. https://doi.org/10.1145/3615886.3627745

Nascimento, E., García, G., Victorio, W., Lemos, M., Izquierdo, Y., Garcia, R., Leme, L., & Casanova, M. (n.d.). A Family of Natural Language Interfaces for Databases based on ChatGPT and LangChain.

Norge Digitalt. (2023). *Generelle vilkår for Norge Digitalt-samarbeidet* (tech. rep.). Retrieved October 23, 2023, from https://www.geonorge.no/globalassets/geonorge2/avtaler-og-bilag-norge-digitalt/generelle-vilkar.pdf

OGC. (2023). OGC Standards. Retrieved October 23, 2023, from https://www.ogc.org/standards/

OpenAI. (2022). Introducing ChatGPT. Retrieved October 26, 2023, from https://openai.com/blog/chatgpt

OpenAI. (2023). GPT-4 Technical Report. https://doi.org/10.48550/arXiv.2303.08774

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A Method for Automatic Evaluation of Machine Translation. In P. Isabelle, E. Charniak & D. Lin (Eds.), *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics* (pp. 311–318). Association for Computational Linguistics. https://doi.org/10.3115/1073083.1073135

Qi, J., Li, Z., & Tanin, E. (2023). MaaSDB: Spatial Databases in the Era of Large Language Models (Vision Paper). https://doi.org/10.1145/3589132.3625597

Radford, A., & Narasimhan, K. (2018). Improving Language Understanding by Generative Pre-Training. Retrieved October 9, 2023, from https://www.semanticscholar.org/paper/Improving-Language-Understanding-by-Generative-Radford-Narasimhan/cd18800a0fe0b668a1cc19f2ec95b5003d0a5035

Richard, T. B. (2023). AutoGPT: The heart of the open-source agent ecosystem. Retrieved October 5, 2023, from https://github.com/Significant-Gravitas/AutoGPT

Roberts, J., Lüddecke, T., Das, S., Han, K., & Albanie, S. (2023). GPT4GEO: How a Language Model Sees the World's Geography. https://doi.org/10.48550/arXiv.2306.00020

Rozière, B., Gehring, J., Gloeckle, F., Sootla, S., Gat, I., Tan, X. E., Adi, Y., Liu, J., Remez, T., Rapin, J., Kozhevnikov, A., Evtimov, I., Bitton, J., Bhatt, M., Ferrer, C. C., Grattafiori, A., Xiong, W., Défossez, A., Copet, J., Azhar, F., Touvron, H., Martin, L., Usunier, N., Scialom, T., & Synnaeve, G. (2023). Code Llama: Open Foundation Models for Code. Retrieved October 30, 2023, from http://arxiv.org/abs/2308.12950

Scherrer, Y., & Ljubešić, N. (2020). HeLju@VarDial 2020: Social Media Variety Geolocation with BERT Models. *Proceedings of the 7th Workshop on NLP for Similar Languages, Varieties and Dialects*, 202–211. Retrieved October 16, 2023, from https://aclanthology.org/2020.vardial-1.19

Shi, W., Min, S., Yasunaga, M., Seo, M., James, R., Lewis, M., Zettlemoyer, L., & Yih, W.-t. (2023). REPLUG: Retrieval-Augmented Black-Box Language Models. https://doi.org/10.48550/arXiv.2301.12652

Skjuve, M., Bae Brandtzaeg, P., & Følstad, A. (2023). Why People Use ChatGPT. https://doi.org/10.2139/ssrn.4376834

Sparrow, R. (2007). Killer Robots. *Journal of Applied Philosophy*, *24*(1), 62–77. https://doi.org/10.1111/j.1468-5930.2007.00346.x

STAC Tutorials. (n.d.). Retrieved October 23, 2023, from https://stacspec.org/en/tutorials/

Stasaski, K., & Hearst, M. (2022). Semantic Diversity in Dialogue with Natural Language Inference. In M. Carpuat, M.-C. de Marneffe & I. V. Meza Ruiz (Eds.), *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 85–98). Association for Computational Linguistics. https://doi.org/10.18653/v1/2022.naacl-main.6

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q. V., Chi, E. H., Zhou, D., & Wei, J. (2022). Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them. Retrieved December 9, 2023, from http://arxiv.org/abs/2210.09261

The Norwegian Mapping Authority. (2019). *Elveg 2.0* (tech. rep.). Retrieved December 2, 2023, from https://register.geonorge.no/data/documents/Produktspesifikasjoner_elveg-2-0_v2_sosi-standardisert-produktspesifikasjon-elveg-2_0_1_.pdf

The Norwegian Mapping Authority. (2023). *Håndbok for Geovekst-samarbeidet* (tech. rep.). Kartverket. Retrieved October 10, 2023, from https://kartverket.no/geodataarbeid/geovekst/veiledningsmateriell-geovekst/

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., & Scialom, T. (2023). Llama 2: Open Foundation and Fine-Tuned Chat Models. https://doi.org/10.48550/arXiv.2307.09288

Unlu, E. (2023). Chatmap : Large Language Model Interaction with Cartographic Data. https://doi.org/10.48550/arXiv.2310.01429

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. Retrieved October 10, 2023, from https://arxiv.org/abs/1706.03762v7

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. https://doi.org/10.48550/arXiv.2201.11903

Weng, L. (2023). LLM Powered Autonomous Agents. *lilianweng.github.io*. Retrieved December 10, 2023, from https://lilianweng.github.io/posts/2023-06-23-agent/

Wu, Q., Bansal, G., Zhang, J., Wu, Y., Li, B., Zhu, E., Jiang, L., Zhang, X., Zhang, S., Liu, J., Awadallah, A. H., White, R. W., Burger, D., & Wang, C. (2023). AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation. https://doi.org/10.48550/arXiv.2308.08155

Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T. L., Cao, Y., & Narasimhan, K. (2023). Tree of Thoughts: Deliberate Problem Solving with Large Language Models. https://doi.org/10.48550/arXiv.2305.10601

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., & Choi, Y. (2019). HellaSwag: Can a Machine Really Finish Your Sentence? https://doi.org/10.48550/arXiv.1905.07830

Zhang, Y., Wei, C., Wu, S., He, Z., & Yu, W. (2023). GeoGPT: Understanding and Processing Geospatial Tasks through An Autonomous GPT. https://doi.org/10.48550/arXiv.2307.07930

Zhou, A., Yan, K., Shlapentokh-Rothman, M., Wang, H., & Wang, Y.-X. (2023). Language Agent Tree Search Unifies Reasoning Acting and Planning in Language Models. Retrieved October 30, 2023, from http://arxiv.org/abs/2310.04406

# Appendices

# A. Task Description from Norkart



**Oppgåve med omfang som kan tilpassast både prosjekt og masteroppgåve**

## LLMs - GIS-analysens død

(kan justerast seinare)

**BAKGRUNN**
Nyere modeller for kunstig intelligens har demonstrert spesielt gode evner til å kunne lære av store mengder ustrukturert og semi-strukturert informasjon. ChatGPT fra OpenAi tok verden med storm – og chat-baserte systemer florerer. Kan chat-baserte modeller skapes for å hente ut GIS-data effektivt? Norkart har en stor dataplattform hvor brukere utvikler mot API'er som i stor grad har GIS/Geografiske data i bunn. GeoNorge er en stor datakatalog hvor brukere slår opp, eller søker kategorisert for å finne data. QGIS, Python, PostGIS, FME og andre verktøy brukes ofte til å gjennomføre GIS-analyser – hvor en GIS-analytiker/data-scientist gjennomfører dette.

«*Finn alle bygninger innenfor 100-meters-belte som er over 100 kvm og har brygger*»

Er dette mulig å få til med dagens tilgjengelige chat-modeller?

**OPPGAVEBESKRIVELSE**
Oppgaven har som hovedmål å undersøke hvordan nyere språkmodeller kan benyttes for å gjennomføre klassiske GIS-analyser ved å bruke standard GIS-teknologi som PostGIS/SQL og datakataloger (OGC API Records fks). Hva finnes av tilgjengelig chat-løsninger? Hvordan spesialtilpasse til GIS-anvendelser? Hvor presise kan en GIS-Chat bli?

Relevante delmål for oppgaven:
1. Kartlegge state-of-the-art
2. Utvikle proof-of-concepts
3. Analysere begrensninger og kvalitet

Oppgaven vil med fordel deles i prosjektoppgave og masteroppgave
- Prosjektoppgave
  - State-of-the-art: Ai-modeller og multi-modal maskinlæring
  - Innhente og utvikle datagrunnlag og API-tilgjengelighet
- Masteroppgave
  - Utvikle proof-of-concepts med tilgjengelige åpne modeller/teknologi
  - Gjennomføre eksperimenter for analyse av kvalitet

Detaljert oppgavebeskrivelse utvikles i samarbeid med studenten.

**ADMINISTRATIVT/VEILEDNING**

Ekstern veileder: (en eller flere)
Mathilde Ørstavik, Norkart
Rune Aasgaard, Norkart
Alexander Nossum, Norkart

Aktuelle vegleiarar og ansvarleg professor ve NTNU (den som har fagansvar nærast oppgåva):

Terje Midtbø (GIS, kartografi, visualisering)
Hongchao Fan (3D modellering, fotogrammetri, laser)

# Acronyms

**AI** Artificial Intelligence.

**API** Application Programming Interface.

**AWS** Amazon Web Services.

**BERT** Bidirectional Encoder Representation from Transformers.

**BLEU** BiLingual Evaluation Understudy.

**CLI** Command Line Interface.

**EU** European Union.

**FAISS** Facebook AI Similarity Search.

**FKB** Felles KartdataBase.

**GAN** Generative Adverserial Network.

**GDAL** Geospatial Data Abstraction Library.

**GDPR** General Data Protection Regulation.

**GIS** Geographic information system.

**GML** Geography Markup Language.

**GPT** Generative Pre-trained Transformer.

**HTML** HyperText Markup Language.

**HTTP** Hypertext Transfer Protocol.

**ISO** International Organization for Standardization.

**JSON** JavaScript Object Notation.

**LATS** Language Agent Tree Search.

**LLM** Large Language Model.

**LSH** Locality-Sensitive Hashing.

**LSTM** Long Short-Term Memory.

**MLM** Masked Language Modelling.

**MMI** Maximum Mutual Information.

**MMLU** Multitask Language Understanding.

**NLI** Natural Language Inference.

**NLIDB** Natural Language Interfaces for Database.

**NLP** Natural Language Processing.

**NSDI** National Spatial Data Infrastructure.

**NSP** Next Sentence Prediction.

**OGC** Open Geospatial Consortium.

**OSM** OpenStreetMap.

**PPO** Proximal Policy Optimization.

**RAG** Retrieval Augmented Generation.

**RLHF** Reinforcement Learning from Human Feedback.

**RNN** Recurrent Neural Network.

**ROUGE** Recall-Oriented Understudy.

**SDK** Software Development Kit.

**SOSI** Samordnet Opplegg for Stedfestet Informasjon.

**SQL** Structured Query Language.

**STAC** SpatioTemporal Asset Catalog.

**TC** Technical committee.

**UI** User Interface.

**WFS** Web Feature Service.

**WMS** Web Map Service.

**XML** Extensible Markup Language.