Artificial Intelligent Agent in Gene editing: Designing a Smart Chatbot for Geneticists

Universitat Oberta de Catalunya

Ana Aragón Pérez

Master Thesis Machine Learning (15 etcs)

Master Bioinformatics & Biostatistics

Name of the tutor:
Alfredo Madrid García

Name of the SRP:
Agnès Pérez Millan

Deadline: 4 June 2025

UNIVERSITAT DE BARCELONA

Final Work Card

Title of the work:	Artificial Intelligent Agent in Gene editing: Designing a Smart Chatbot for Geneticists
Name of the author:	Ana Aragón Pérez
Name of the tutor:	Alfredo Madrid García
Name of the SRP:	Agnès Pérez Millan
Date of delivery:	Deadline: 4 June 2025
Studies or Prgram:	Master Bioinformatics & Biostatistics
Area or the Final Work:	Master Thesis Machine Learning (15 etcs)
Language of the work:	English
Keywords:	Artificial Intelligence, Agents, Large Language Models, CRISPR, Bioinformatics, Genomics

Abstract

Gene editing relies on the use of several bioinformatic tools and requires highly specific domain knowledge. This master thesis develops an Artificial intelligent-driven agent enhancing workflow and information retrieval of gene editing experiments. The agent through the implementation of the Large Language Model 'gpt-4o-2024-08-06' and LangGraph framework interprets user queries, automates tool selection, and facilitates the search of information. The developed system integrates Tavily Web Search alongside five specialized bioinformatics tools: FORECasT, FORECasT-Repair, WGE CRISPR Search, WGE Off-Targets, and NCBI Libraries. Together, these components enable the agent to effectively support CRISPR-Cas9 knockout applications, with a particular focus on sgRNA design, mutation outcome prediction, and off-target site identification. Additionally, Tavily and the NCBI Libraries extend the agent's capabilities by facilitating broader information retrieval in response to user queries. However, the system does not currently support other gene editing methodologies beyond CRISPR-Cas9 knockout. Future development will focus on expanding tool integration, improving performance through retrieval-augmented generation (RAG), and enhancing the system's ability to address a broader range of gene editing needs. All information regarding this project can be found in the following link https://github.com/A-Aragon/TFM

Abstract

La edición genética depende del uso de diversas herramientas bioinformáticas y requiere un conocimiento altamente especializado del dominio. Esta tesis de máster desarrolla un agente impulsado por inteligencia artificial que mejora los flujos de trabajo y la recuperación de información en experimentos de edición genética. A través de la implementación del modelo de lenguaje grande 'gpt-4o-2024-08-06' y del marco LangGraph, el agente interpreta consultas del usuario, automatiza la selección de herramientas y facilita la búsqueda de información. El sistema desarrollado integra Tavily Web Search junto con cinco herramientas bioinformáticas especializadas: FORECasT, FORECasT-Repair, WGE CRISPR Search, WGE Off-Targets y NCBI Libraries. En conjunto, estos componentes permiten al agente brindar soporte eficaz en aplicaciones de knockout con CRISPR-Cas9, con un enfoque particular en el diseño de sgRNA, la predicción de resultados mutacionales y la identificación de sitios fuera del objetivo. Además, Tavily y las bibliotecas de NCBI amplían las capacidades del agente al facilitar la recuperación de información general según las necesidades del usuario. El desarrollo futuro se centrará en ampliar la integración de herramientas, mejorar el rendimiento mediante técnicas de generación aumentada por recuperación (RAG) y fortalecer la capacidad del sistema para abordar un espectro más amplio de necesidades en edición genética.

Contents

List of Figures

List of Tables

List of Abbreviations

AI Artificial Intelligence

API Application Programming Interface

Cas CRISPR Associated Protein

CRISPR Clustered Regularly Interspaced Short Palindromic Repeats

CRISPRa CRISPR Activation

CRISPRi CRISPR Interference

DNA Deoxyribonucleic Acid

LLM Large Language Models

NLP Natural Language Processing

RNA Ribonucleic Acid

RAG Retrieval-Augmented Generation

sgRNA Single guide RNA

SDGs Sustainable Development Goals

Chapter 1

Introduction

1.1 Context & Justification

Gene editing technology allows for precise genetic modifications in living organisms. This field mainly took off with the refinement of the gene editing system CRISPR-Cas9 back in 2012 (Jinek et al., 2012). The impact of this system in the field has been so relevant that Emmanuelle Charpentier and Jennifer A. Doudna, received the Nobel Prize in Chemistry in 2020. This technology holds a huge potential for its application in broad range of fields such as agriculture, energy and health.

By 2050, world's population is expected to reach 9.6 billion resulting in an increase demand of 60% for staple crops (Tilman et al., 2011) (crops grown in large quantities that form the basis of population's diet). Gene editing could help address food supply demands by improving crop yield, nutritional attributes and disease/herbicide resistance (Zhu et al., 2020). Biofuels are a renewable alternative to fossil fuels. Gene editing can accelerate biofuel commercialization by enhancing organism production (Ajjawi et al., 2017). Moreover, gene editing is transforming healthcare. In 2023, the first CRISPR–Cas9 therapy was approved for Sickle Cell Disease and η-thalassemia (Sheridan, 2024). For instances, this field shows great potential for treating diseases such as cancer and HIV (Henderson, 2024).

Furthermore, CRISPR-Cas9 has driven the emergence of new tools that address the potential negative effects of this mechanism such as CRISPR a/i based on enhancing or suppressing gene expression via epigenetic regulation (Qi et al., 2013), prime editing which avoids introducing double-stranded breaks (Anzalone et al., 2019) and base editing that enables conversion of one DNA base into another (Gaudelli et al., 2017). Thus, increasing the potential of gene editing application.

Nevertheless, designing gene-editing experiments demands extensive domain knowledge. The process involves numerous decision-making steps, and the growing number of available tools makes it increasingly challenging to stay up to date and make informed choices based on the latest advancements. To address this, the agent was developed to automate and simplify

the design process of gene editing experiments, supporting researchers during the design and development phase.

On the other hand, the rise on AI holds great potential to overcome this challenge. LLM are deep learning models pretrained on vast amounts of data with millions of parameters that outshine in natural language processing. Nowadays, LLM are built on the Transformer architecture (Vaswani et al., 2017). These models can play a crucial role in generative AI, enabling the creation of human-like content based on given prompts. In fact, they have significantly reshaped how people interact with search engines. Their tasks include summarizing documents, engaging in conversations, generating tailored responses based on user input, translation and classifying content

Recent research has focused on enhancing their problem-solving by integrating external tools (Xi et al., 2025) leading to the rise of revolutionary AI agents. Their access to real tools make AI agents more powerful and efficient in real-world applications. LLM could assist gene editing design experiment through suggestions on the appropriate gene editing technique, the delivery method selection and experimental protocol recommendation. Additionally, amplifying the potential of LLM as AI Agents would also allow to assist on sgRNA design and off target evaluation through external tools. Thus, augmenting the assistance by LLM in the design process. However, the available LLM general-purpose models and AI Agents have notable limitations in the gene editing field due to their lack of knowledge on it, as analyzed by (Huang et al., 2024). These limitations highlight the need for custom-tailored LLMs/ AI Agents for gene-editing experimental designs.

1.2 Research Objectives

The main purpose of this project is to develop an LLM-based AI Agent capable of assisting users in the design of gene editing experiments by interacting with various external applications and documentation and Tavily web search.

The sub-objectives of this study, aimed at achieving the main purposes, are outlined as follows:

1. Developing an AI agent with the available tools and documentation in the field of gene editing that can be utilized.
2. Equipping the AI Agent with Tavily web search.

1.3 Environmental, Ethical-Social, and Diversity Impact

UOC University is committed to the Ethical and Global Commitment Competency (CCEG) which is defined as follows:

"Acting honestly, ethically, sustainably, socially responsibly, and with respect for human rights and diversity, both in academic and professional practice, and designing solutions to improve these practices."

This competency should be addressed during the design, developmental and final stages of the project and encompasses three main dimensions:

I. Sustainability

II. Ethical behavior and social responsibility (SR)

III. Diversity (including gender, among others) and human rights

These three dimensions align with the UN's 2030 Sustainable Development Goals (SDGs) to which the UOC is publicly committed see figure 1.1.

Figure 1.1: The SDGs by United Nations

The SDGs might also be classified in three main sections: biosphere, society and economy. According to this classification, economies and societies should be seen as embedded parts of the biosphere, not as separate parts. In this project, the previous classification proposed by UOC will be followed. However, connections among all the SDGs will be emphasized.

Gene editing can be applied in multiple fields such as health, agriculture ans energy industry. Thus, this project indirect positive impacts heavily rely on the research aim of the user. However, this project prioritizes the direct impacts to facilitate comprehension.

Regarding negative impacts, it should be noticed that this project might have a light environmental footprint due to computational requirements. This footprint might scale in a near future if the project continues to be developed. However, this impact would likely be outweighed by the significant indirect positive effects of facilitating gene editing.

1.3.1 Environmental Impact

This project could have an impact on the following SGDs related to sustainability:

- 13. Climate Action (+/-)
  - (-) High Energy Consumption: Running AI models requires significant computational power, potentially increasing carbon emissions if powered by non-renewable energy sources.

1.3.2 Ethical-Social Impact

This project may have an impact on the following SGDs related to ethical-social and economy impact:

- 7. Affordable and Clean Energy (+/-)
  - (-) Potential Increase in Energy Costs: the computational resources demanded by this project would likely lead to elevated energy consumption, thereby contributing to higher energy costs.

1.3.3 Diversity Impact

This project could have an impact on the following SGDs related to diversity impact through the implementation of data sources that include balanced data on gender and races:

- 10. Reduced Inequalities (+/-)
  - (+) Democratizing knowledge: This model would facilitate the access to highly technical knowledge to new actors. Thus, reducing inequalities.

– (-) Disparities in Access to Technology: While AI-driven gene editing can reduce inequalities, the high cost and technical expertise required may limit access to wealthier nations or large corporations, widening the gap between privileged and vulnerable communities.

A further discussion about impacts on the mentioned SDGs is presented at 9.

1.4 Project Strategy

The aim of this project is to develop an AI Agent to facilitate the labour of researchers in the field of gene editing. However, the selected strategy does not involve building the AI agent from scratch. This project will focus on the search and aggregation of already built tools involved in the design of gene editing experiments and documentation related to the field. Additionally, Bing web search will be implemented and in case of time availability, Retrieval-Augmented Generation (RAG) will also be implemented to enhance the agent's accuracy by retrieving relevant, domain-specific information. This external database would include up-to-date scientific literature and curated datasets to minimize hallucinations and ensure more reliable responses.

Firstly, a literature review will be conducted to gain a comprehensive understanding of the current state-of-the-art. Secondly, a thorough search will be performed to identify available tools and libraries for the AI Agent. The search will start with the following list (Liwei, 2020) and it will be complemented with different libraries. Subsequently, the expected input prompts for the AI Agent will be defined in parallel to the creation of agent functions to call the multiple APIs with an LLM and a framework.

The LLM will be selected from among the most prominent models contributing to the advancement of AI agents, including ChatGPT by OpenAI, Claude by Anthropic, Gemini by Google, Qwen by Alibaba, Llama by Meta, and the R1 and R3 models developed by DeepSeek. In terms of orchestration frameworks, LangGraph, AutoGen, and CrewAI are among the most notable and are being considered as potential candidates for integration.

Finally, the agent will be evaluated, and a web demo with Gradio will be developed for demostration purposes.

1.5 Planning

1.5.1 Milestones

1. Define the agent's purpose and environment
2. Gather and prepare relevant tools

3. Select the right technology for the AI Agents components.
4. Design the AI Agent
5. Test the AI Agent
6. Deploy and monitor the AI agent

1.5.2 Time Planning - Gantt Chart

The following Gantt chart presents the tasks and their time associated up to June 1st, the expected submission date for this project. This date includes a three-day margin before the final deadline to handle any unexpected submission issues.

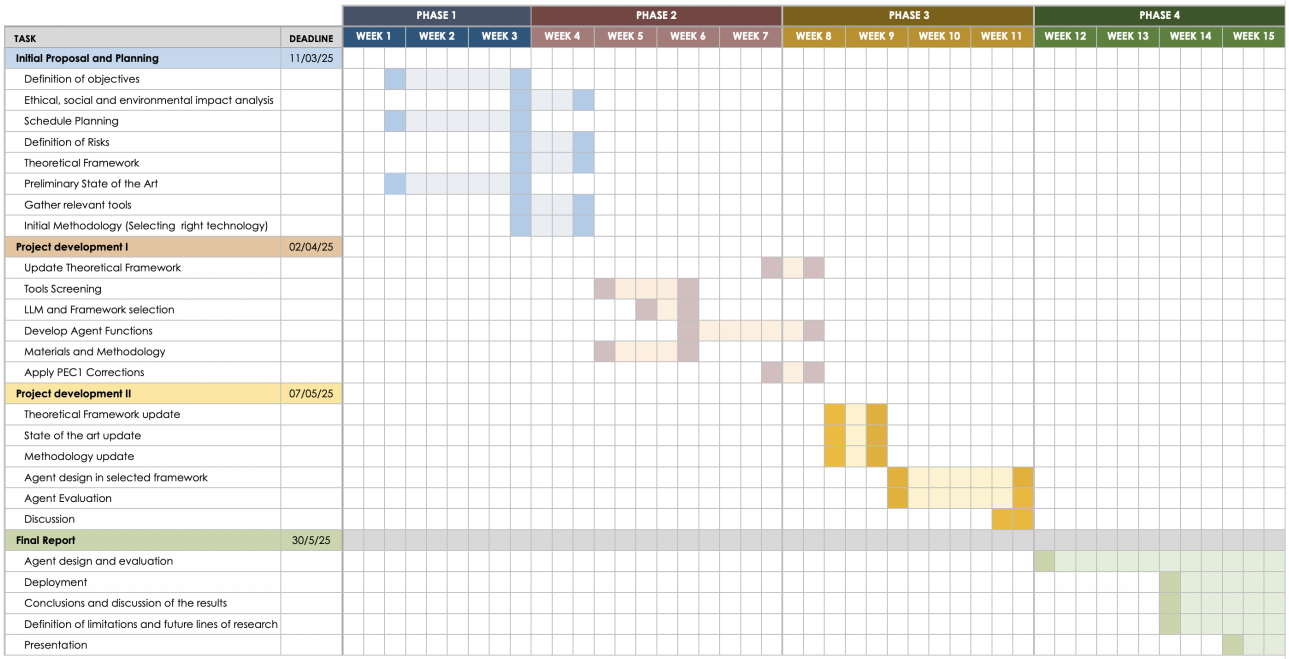

Figure 1.2: Gantt Chart

1.5.3 Potential Risks and Alternatives

Several risks have been identified that could potentially affect the project timeline:

High Impact Risks

1. **Data and Tools Accessibility**: Acquiring necessary data and tools can be time-consuming and may result in an insufficient collection of sources. Additionally, many available tools may not provide open or free APIs, limiting accessibility.

Mitigation: Broaden the types of data and tools used. Explore alternative open-source solutions. Establish collaborations to access proprietary tools.

2. **High Computational Resources**: Running LLMs can require a vast amount of computational resources which can lead to high operational costs.

   Mitigation: Reduce context size. Use smaller models. Implement cloud-based solutions.

3. **Maintenance Challenges**: Gene editing is a rapidly evolving field, with new information emerging frequently. Failing to keep AI models updated may lead to outdated recommendations and incorrect outputs.

   Mitigation: Integrate real-time data sources. Update LLM and framework will be updated to integrate superior-performing models as they become available.

Moderate Impact Risks

1. **Poorly Defined Prompts (Galileo AI, 2024)**: A well-defined task is essential for effectively operating AI agents. Otherwise, agents may struggle to make appropriate decisions.

   Mitigation: Define clear objectives. Use prompt engineering techniques that have proven to work with agents in the past.

2. **Output Evaluation and Correctness (Galileo AI, 2024)**: Ensuring both the accuracy and correctness of outputs is critical, as errors could lead to unintended or harmful genetic modifications. Establishing clear success metrics remains challenging due to biological and AI Agents complexity.

   Mitigation: Conduct continuous evaluation. Use real-world experimental validation. Incorporate feedback loops and implement rigorous verification protocols to assess correctness and safety.

3. **Planning and Reasoning Failures (Galileo AI, 2024)**: Effective planning and reasoning are essential for AI agents to execute the design of complex gene-editing experiments. Planning allows agents to anticipate future states, make informed decisions, and carry out tasks in the required order, while reasoning enables them to interpret information, solve problems, and adjust strategies dynamically. However, LLMs often struggle with these capabilities.

   Mitigation: Implement task decomposition and utilize specialized agents for the different groups of tasks to make them more manageable (multi agents). Generate multiple plans and select the most appropriate one based on the context.

4. **Agent Scaling (Galileo AI, 2024)**: The AI Agent should efficiently scale for broader, higher amount and future more complex applications that may appear in the field of gene editing.

   Mitigation: Design scalable architectures. Optimize resource management. Monitor performance.

Low Impact Risks

1. **Tool Calling Failures (Galileo AI, 2024)**: One key benefit of AI Agents over LLMs is their ability to solve complex problems by calling multiple external tools. However, agents often struggle with tool calling due to incorrect parameters, misinterpreted outputs, or integration failures.
    - *Mitigation: Define clear parameters. Validate tool outputs. Implement tool selection verification.*
2. **Infinite Looping Risks(Galileo AI, 2024)**: AI agents may get stuck in unproductive loops when performing tasks.
    - *Mitigation: Establish clear termination conditions*
3. **Project Scope**: Without a well-defined project scope, the AI agent may not align with the intended gene-editing objectives.
    - *Mitigation: Clearly define the project scope. Set realistic and measurable goals.*
4. **Limited Available Time**: The amount of time assigned to this project is very limited (375h). Broad goals and tasks might delayed project's progress. Additionally, risks and minor challenges threat achieving the main goals.
    - *Mitigation: Establish narrow goals and structured timeline. Prioritize critical development phases. Allocate buffer time for unexpected challenges.*

Chapter 2

Theoretical Framework

This chapter outlines the essential concepts for comprehending the research conducted in this project.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a core area within computational linguistics, which focuses on developing computer systems capable of understanding and generating human language. According to Chowdhary (2020), it is defined as follows:

> *Natural language processing is a collection of computational techniques for automatic analysis and representation of human languages, motivated by theory.*

NLP can be classified into Natural Language Understanding (NLU) and Natural Language Generation (NLG). NLU examines various aspects of language, including phonology (sound systems), morphology (word formation), lexicon (word meanings), syntax (sentence structure), semantics (sentence meaning), discourse (how sentences form coherent texts or conversations), and pragmatics (how context shapes meaning). Further, NLG is the process of producing phrases, sentences and paragraphs that are meaningful from an internal representation (Khurana et al., 2023).

2.2 Large Language Models

In recent years, Large Language Models are very large deep learning models that are pretrained on vast amounts of data. (LLM) have transformed the field of NLP — these advanced AI models are highly effective at understanding and producing natural language. LLMs are

deep learning models trained on vast amounts of data and containing millions or billions of parameters.

The architecture of language models used to rely on Convolutional Neural Networks (CNN) or Recurrent Neural Networks (RNN) (in particular long short-term memory (Hochreiter and Schmidhuber, 1997) and gated recurrent (Chung et al., 2014) neural networks) including a encoder-decoder and sometimes even an attention mechanism until the appearence of the vanilla transformer architecture that relied solely on attention and self-attention mechanisms presented by Vaswani et al. (2017) (the term "vanilla" refers to the original, unmodified version of a model or algorithm).

As mentioned by this source:

> *Transformer, a model architecture eschewing recurrence and instead relying entirely on an attention mechanism to draw global dependencies between input and output [...] the first transduction model relying entirely on self-attention to compute representations of its input and output without using sequence-aligned RNNs or convolution.*

By removing recurrence and convolutions, vanilla transformer showed to be superior in quality while being more parallelizable and scalable in terms of training. The use of attention mechanism and its extension as self-attention mechanism introduced by this source revolutionazed language modeling.

2.2.1 Attention and Self-Attention Mechanisms

Attention mechanism was first introduced by Bahdanau et al. (2014) as a novel architecture that learns to align and translate simultaneously. This mechanism was introduced as an extension to the basic encoder-decoder model in order to solve the potential issue of compressing all the necessary information of a source sentence into a fixed-length vector which made it difficult to cope with long sentences.

Instead, this mechanism encodes the input sentence into a sequence of vectors and chooses a subset of these vectors adaptively while decoding the translation. In other words, it enables the model to attend to all words in the sequence, allowing for more effective handling of long sentences.

Self-Attention (Vaswani et al., 2017) extends the attention mechanism allowing neural networks from the encoder and decoder to understand a word in the context of the words around it. As described in this source, the vanilla transformer architecture makes use of attention and self-attention mechanisms in a multi-head attention structure in three different ways: in encoder-decoder attention layers (typical encoder-decoder attention mechanism - allows every position in the decoder to attend over all positions in the input sequence) and self-attention layers in both, the encoder and decoder allowing these two components to attend all positions in the previous layers.

2.2.2 Vanilla Transformer Architecture

The vanilla Transformer was originally conceived for machine translation tasks and has since served as the foundational reference point, from which many variants and improved architectures—such as BERT, GPT, and T5—have since evolved. This architecture consisted of an encoder and decoder (both are NN architecture capable to model sequences). The encoder reads the input sequence, understands its context while the decoder generates the corresponding output. These are composed of the parts shown in figure 2.1.

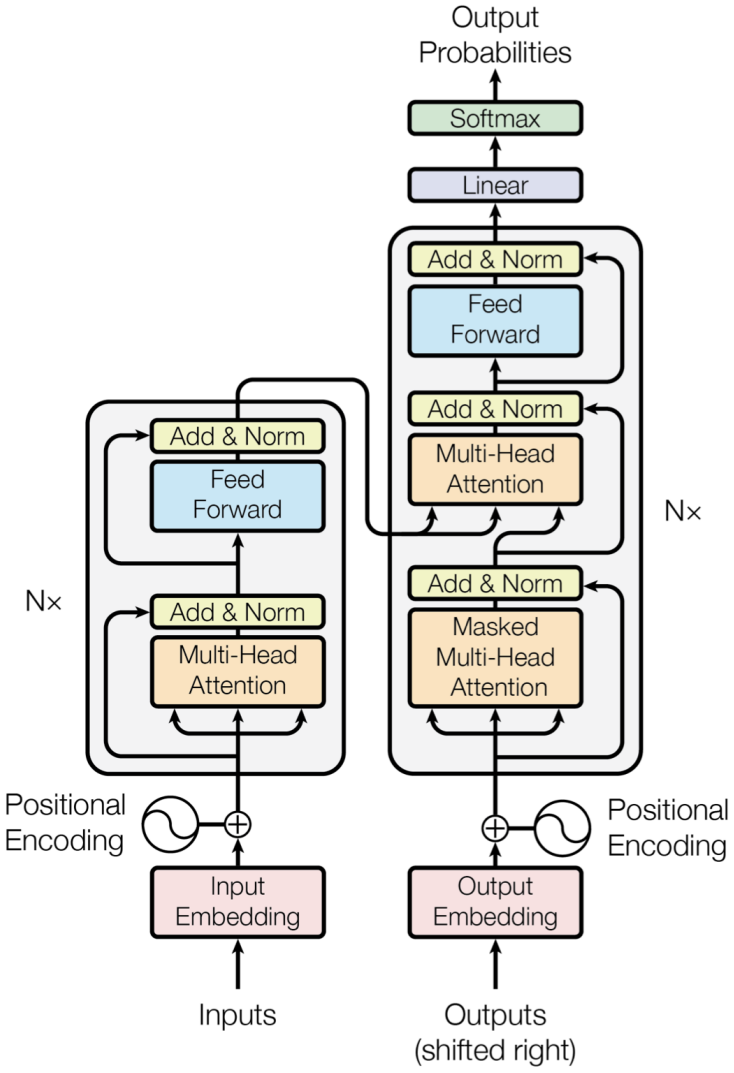

Figure 2.1: Transformers architecture (Vaswani et al., 2017).

Encoder

1. Input embeddings: The input is fed into a layer for word embedding, converting the input tokens to vectors of dimension 512. The tokenization concept is introduced in Section 2.2.5.

2. Positional encoding: Positional encoding, alongside attention and self-attention mechanisms, represented a game changer in transformers architecture. In transformers, unlike RNNs or CNNs, there is no inherent notion of word order since words are not processed sequentially or locally (in chunk of words). The model sees all words at once. Consequently, the token order information is assigned to the input embeddings before being processed through a sine-cosine encoding which output is a matrix with each row representing an embedding of the sequence summed with its positional information.

3. Residual connection: transports unprocessed input of a sublayer to a layer normalization function to preserve key information such as positional encoding.

4. Multi-head attention: Integrates several self-attention mechanisms, allowing each word to attend to all other words in the sentence in order to generate more informative and context-aware embeddings.

5. Post-layer normalization: layer that follows every attention and feedforward sublayers.

6. Feed-forward network: this network contains two fully-connected layers and uses ReLU as the activation function. It should be noticed that it is a position-wise network.

Decoder

1. Output embeddings: the input of output embeddings differs between training and inference (prediction). The decoder uses the actual target sequence during training while during inference it receives its own previously generated tokens as input to generate the next one (outputs shifted right). Hence, the model is auto-regressive. However, the mechanism of embedding is similar to the encoder one.

2. Positional encoding: equivalent to the encoder one. Positional embeddings are transferred to the masked multi-head attention layer.

3. Masked multi-head attention: During the inference phase, the model needs to behave auto-regressive since it does not know the correct following token. However, while training, this masking ensures that the predictions for position i can depend only on the known outputs at positions less than i. This mechanism forces the transformer to learn how to predict, as future tokens are masked in order to parallelize the process of training to eliminate the need for sequential execution in training and speed up the process.

4. Post-layer normalization: same as in encoder.

5. Multi-head attention: takes the output from the encoder and combines it with previous layers from the decoder.

6. Feed-forward network: same as in encoder.

7. Linear layer and Softmax: produces the next probable element of a sequence, thanks to the softmax classifier that emits probabilities of an output.

2.2.3 LLM Transformer Variant

Transformer variants have led to the development of different models (Burtenshaw et al., 2025). LLMs are typically decoder-based models with billions of parameters. A decoder-based Transformer focuses on generating new tokens to complete a sequence, one token at a time. This architecture proves highly effective for a range of tasks, including text generation, chatbot interactions, and code generation. The GPT model is widely recognized as a leading example of a decoder-based architecture.

2.2.4 LLM Foundational Models

Pretraining is the act of training a model from scratch: the weights are randomly initialized, and the training starts without any prior knowledge. This process requires large amounts of data and time. Pretrained models can be adapted to specific tasks through methods like fine-tuning or prompting. Fine-tuning is an additional training with a dataset specific to the aimed task. The fine-tuning will only require a limited amount of data: the knowledge from the foundation model is "transferred," hence the term transfer learning. Prompting involves giving a model task-specific instructions or relevant examples to help it understand the context and produce a suitable response (Huggingface, 2022; Caballar, 2024).

Foundation models are pretained models. They serve as the common basis from which many task-specific models are built via adaptation. The term 'foundation' highlights the significance of architectural stability, safety and security of the model as well-executed foundations are reliable groundwork for building future applications (Bommasani et al., 2021).

Most of the well-known foundation LLMs come from a variety of leading AI research organizations. These include Deepseek-R1 by DeepSeek, GPT-4 by OpenAI, Llama 3 developed by Meta (formerly Facebook AI Research), SmolLM2 from Hugging Face, Gemma by Google, and Mistral from the homonimus company. Each of these models brings unique strengths and design choices tailored to different applications and performance goals.

2.2.5 LLM Pipelines

This pipeline presents a structured approach for applying a Large Language Model (LLM) to tasks such as text generation, summarization, and question answering. The figure 2.2 outlines key stages.

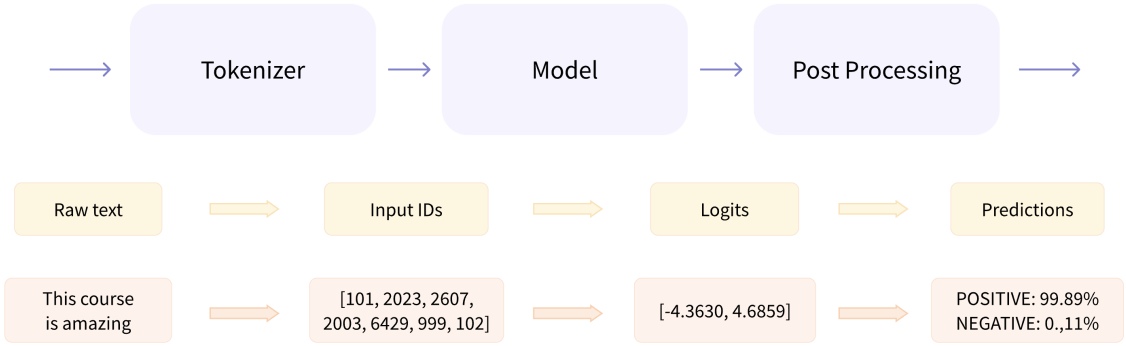

Figure 2.2: LLM Pipeline (Huggingface, 2022)

Text Preprocessing: Tokenization

A "token" is the unit of information an LLM works with. You could think of it as a "word", but for efficiency reasons LLMs don't use whole words (Burtenshaw et al., 2025). For instance, the word skyscrapper consists of four tokens (sk-ys-cr-apper) with the following IDs in the model gpt-4-text-embedding-ada-002: [4991, 1065, 5192, 3183]. The objective of an LLM is to predict the next token, given a sequence of previous tokens.

For this purpose, LLMs use special tokens to define the boundaries of structured elements in its input prompts and generated outputs. The most important of those is the End of sequence token (EOS) - It should be noticed that the forms of special tokens are highly diverse across model providers. For instance, in GPT-4 EOS is ¡—endoftext—¿ while in LLama3 is ¡—eot_id—¿. LLMs work in an autoregressive manner, meaning each generated token is fed back into the model to predict the next one. This process continues until the model outputs the EOS token, indicating that the response is complete (Burtenshaw et al., 2025).

Feature Extraction and Text Analysis: Model

Once the input text is tokenized, the model computes a representation of the sequence that captures information about the meaning and the position of each token in the input sequence through the Transformer architecture —a deep learning architecture based on the "Attention" mechanism (Vaswani et al., 2017). When predicting the next word, not every word in a sentence is equally important. The attention mechanism allows to identify the most relevant words to predict the next token.

Finally, the transformer representation goes into the model "head", which outputs scores that rank the likelihood of each token in its vocabulary as being the next one in the sequence. Based on these scores, we have multiple strategies to select the tokens to complete the sentence. The easiest decoding strategy would be to always take the token with the maximum score.

Post-Processing

The output values from our model are not directly interpretable as probabilities. Instead, these are logits, which are the raw, unnormalized scores generated by the model's final layer.

To obtain probability values, these logits must be passed through a SoftMax layer (Burtenshaw et al., 2025).

2.3 AI Agents

AI agents are autonomous systems that perform specific tasks by making rational decisions based on user input to achieve specific tasks thanks to the implementation of LLMs (AI models that process natural language) or large multimodal models (AI models that process natural language, images, video and/or audio). These systems are capable of interpreting user instructions, maintaining context throughout conversations and selecting the right tools.

2.3.1 LLM vs. Traditional Agents

The concept of AI Agents has been around for decades. As early as 1959, (McCarthy, 1959) proposed the "advice taker", considered the first complete AI system. In the following years, the concept of agent aroused through computing programs that could sense and reason in specific environments to achieve certain tasks. In approximately 2015, interest in AI agents was renewed with the rise of reinforcement learning and gaming strategies (e.g. AlphaGo (Silver et al., 2016) and OpenAI Five (Berner et al., 2019)). Recently, the emergence of LLMs has boost AI agents once again.

Traditional Agents were deterministic, designed to excel in specific tasks relying on a set of rules. Consequently, they had limited adaptability and used to struggle with tasks outside their initial scope. However, the integration of LLMs in AI agents transformed them into probabilistic and flexible agents capable of adjusting to new situations, integrating various tools and learning from fault behavior (Zhao et al., 2023).

2.3.2 LLM AI Agents

A highly illustrative and comprehensible conceptual framework with three key components: brain, perception, and action has been proposed for the design and construction of LLM-based agents by (Xi et al., 2025). The brain is primarily composed of an LLM which serves as the central controller of an AI agent. The perception is basically the input received by the agent which is afterwards sent to the brain. Finally, the action is composed by the mechanisms involved in the response of the input stimulus (see figure: 2.3). Another view would be the "action-response" mechanism. The agent serves as the receptor of an action or stimulus (perception component) and subsequently triggers a response (action component). The brain is responsible for processing both the stimulus and the underlying action mechanisms that produce the response.

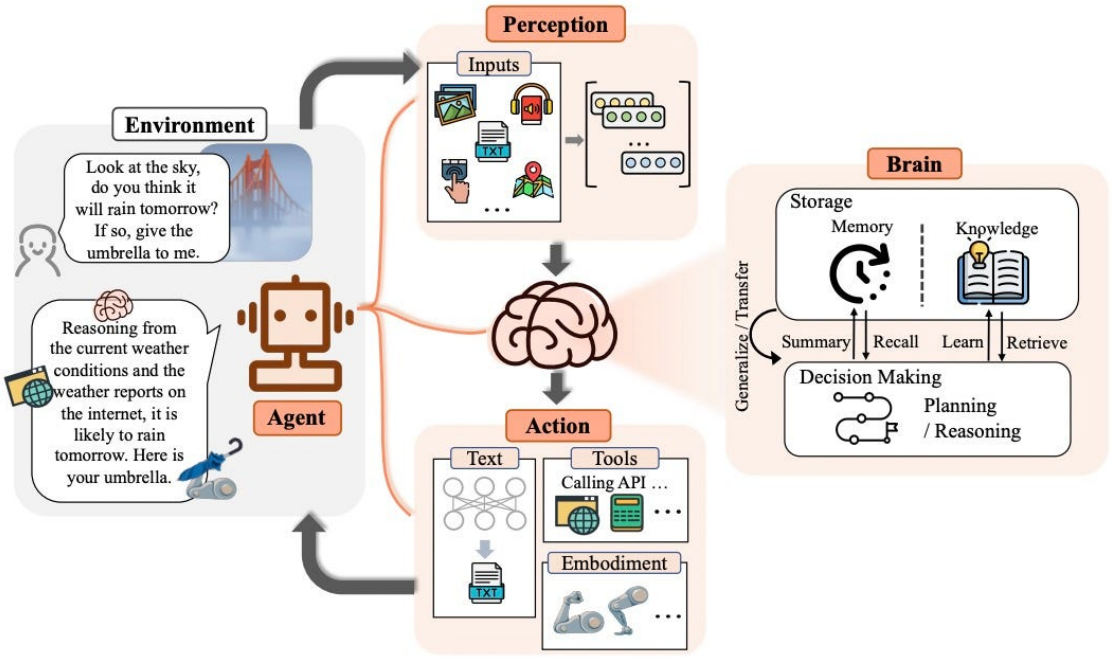

Figure 2.3: AI Agents Framework by Xi et al. (2025)

A highly detailed framework was proposed by Larsen et al. (2024) in which an AI Agent is composed from the main following components, see figure 2.4.

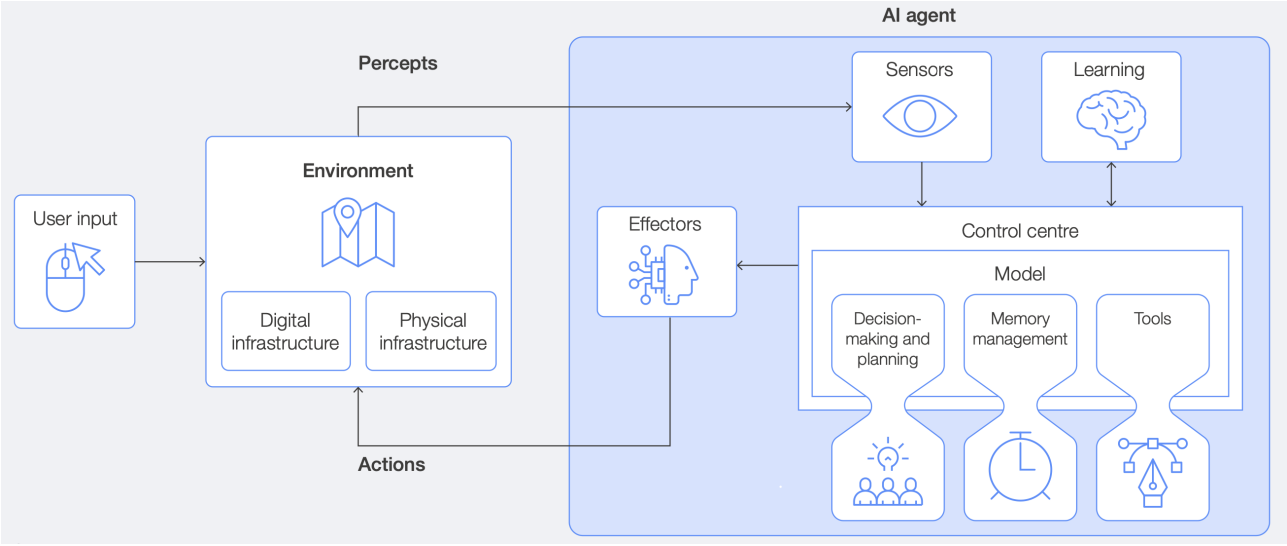

Figure 2.4: AI Agents Core Components (Larsen et al., 2024)

- User Input: instructions received by the user. In this case, input will be prompts via chat-based interface but instructions could be also introduced by voice-based commands or pre-recorded data

- Environment: it represents the arena where the agent utilizes its sensors and effectors to perceive and act over it guided by user inputs and the actions determined by the control center. The environment can be either a physical or digital infrastructure.
- Sensors: the agent perceives its environment through sensors, which may be physical (e.g., cameras, microphones) or digital (e.g. database queries).
- Percepts: data the AI agent receives from sensors or other sources, forming its understanding of the environment.
- Control centre: core of the AI agent. It processes information, makes decisions, and plans actions using models that evaluate options and select optimal outcomes.
    - Decision-making and planning: LLMs or large multimodal models LMM (AI models that process natural language, images, video and/or audio) outputs enable decision-making and planning.
    - Memory management: It enables the AI agent to retain previous interactions and context, which is essential for keeping conversations consistent in chatbots.
    - Tools: enable the AI agent to access and interact with multiple functions or modalities.
- Effectors: the tools an agent uses to take actions upon its environment.
- Actions: alterations made by the effectors. In digital environments they could be linked to updating a database.

There is an application layer surrounding the control centre and involving effectors-sensors, acting as the interface between the AI agent and its environment.

The learning component is an inherent part of the model, allowing the AI agent to enhance its performance over time as the model gathers more input, through machine learning and deep learning techniques (supervised learning, reinforcement learning, reinforcement learning with human feedback, transfer learning, fine-tuning).

An AI agent system combines multiple AI agents. Each agent typically has specialized capabilities, knowledge, and decision-making processes, while collaborating and sharing data to achieve a common system objective. The future of AI agents, based on the same source, points toward Multi-Agent Systems (MAS) — organized structures that combine independent AI agents and AI agent systems that collaborate, compete or negotiate to achieve collective goals.

LLM Agents Tasks

According to Zhao et al. (2023), the tasks of an LLM-AI Agent can be classified into:

- Chatbot: an increasing number of AI agents are being designed as LLM-based chatting agents.
- Game: gaming agents excel at specific games.
- Coding: several AI agents have been developed to support programming efficiency.
- Design: agents focused on transforming user's ideas into designs.
- Research: a range of AI agents has been introduced for conducting autonomous scientific research.
- Collaboration: this section focuses on multi-agent systems such as MetaGPT and Multi-GPT that enable multiple agents to autonomously divide tasks and collaborate to complete them, with MetaGPT placing greater emphasis on applications within the software industry.
- General purpose: these AI agents are designed to carry out a wide variety of tasks across different domains.

2.4 Gene Editing

2.4.1 CRISPR-SpCas9 System

Prior to being harnessed as a genome engineering tool, the CRISPR system served an important function in nature. CRISPR (Clustered Regularly Interspaced Short Palindromic Repeat) sequences were initially discovered in the E. coli genome in 1987 (Ishino et al., 1987). However, their function as a prokaryote adaptive immune response against bacteriophages that detects and eliminates foreign DNA was elucidated in 2007 (Barrangou et al., 2007).

This adaptive immune system serves as a genetic memory. It is based on the inclusion of small fragments of the phage DNA (spacers) in the bacteria/archaea genome, specifically into the CRISPR array during infection. This array consists of repeated sequences of bacteria/archaea genetic code, interrupted by "spacer" sequences – genome fragments from past phages invaders. These small fragments will be used as guides by Cas proteins to confer bacteria with resistance against these invaders.

The popularity of CRISPR is largely due to its simplicity. Before CRISPR-Cas9, genome engineering approaches like zinc finger nucleases or transcription-activator-like effector nucleases were very complex and had a lower adaptability which limited the field of gene editing. As shown in figure 2.5, the CRISPR-Cas system relies on two main components:

1. **Guide RNA (gRNA)**: recognizes and directs the Cas nuclease to the target DNA region of interest
2. **CRISPR-associated (Cas)**: nuclease,a non-specific endonuclease in charge of making the double-strand break in the target DNA region. It is worth noting that there are several versions of Cas nucleases isolated from different bacteria. The most commonly used one is the SpCas9 nuclease from *Streptococcus pyogenes* (therefore the so called 'CRISPR-SpCas9 System').

In 2012, (Jinek et al., 2012) used a single guide RNA (sgRNA) as crRNA+tracrRNA and demostrated that CRISPR-Cas9 system could be used for "RNA-programmable genome editing". The crRNA provides the 17–20 nucleotide sequence complementary to the target DNA, guiding Cas9 to the correct site, while the tracrRNA acts as a binding scaffold that enables Cas9 to associate with the sgRNA complex. The crRNA is the customizable component that enables specificity in every CRISPR experiment (Synthego, 2025b).

In essence, the sgRNA and SpCas9 nuclease form a ribonucleoprotein (RNP) complex. The presence of a specific protospacer adjacent motif (PAM) in the genomic DNA is required for the complex to bind to the target sequence. Then, the Cas9 nuclease makes a double-strand break in the DNA (commonly known as the scissors). Then, the cell's natural DNA repair mechanisms triggered by the double-strand break lead to either gene knockout (via frameshift mutation) or knock-in (if a DNA repair template is provided) (Synthego, 2025b).

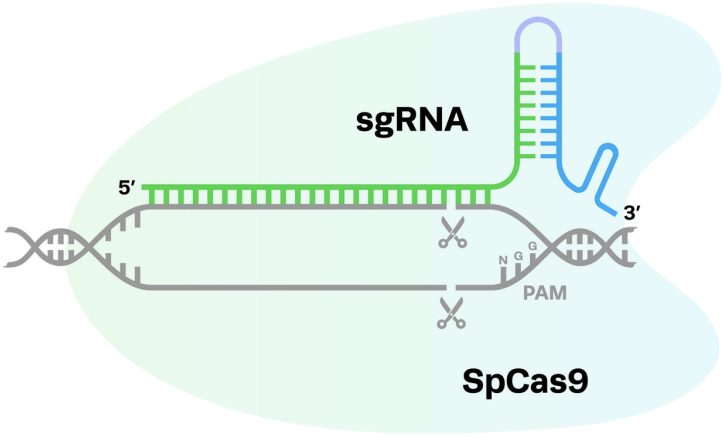

Figure 2.5: The CRISPR-SpCas9 System by Synthego (Synthego, 2025b)

2.4.2 gRNA & sgRNA

Building upon the prior mention of gRNA and sgRNA, it is necessary to clarify their specific differences. What fundamental distinction exists between these two terms within the realm of CRISPR technology?

As the name implies, 'sgRNA' is a single RNA molecule that contains the crRNA sequence fused to tracrRNA sequence by a linker loop. In Figure 2.5, the components of the sgRNA are shown: crRNA (green), Linker loop (purple) and Scaffold tracrRNA (blue). In contrast, 'gRNA' is a general term that encompasses all CRISPR guide RNA formats. However, since sgRNAs have become the most widely used format, the terms are often used interchangeably (Synthego, 2025b).

2.4.3 PAM Sequence

The CRISPR system serves as a bacterial immune defense against viruses (bacteriophage). When infected, surviving bacteria keep a part of the viral DNA as a way to remember the past infection so they can fight off the virus easier next time. During an infection, Cas1 and Cas2 nucleases identify the virus, then Cas9 cuts out a segment of the virus, known as a protospacer, which is added to the CRISPR array between the sequence repeats. In the future, if the same virus invades again, the bacterial cell now has a complementary RNA sequence which recruits the Cas nuclease to cut the viral DNA and stop the infection (Synthego, 2025a).

The question is: how does the viral genome fragment embedded in the bacterial genome not trigger the Cas9 to cut the bacterial genome? The PAM, also known as the protospacer adjacent motif, is a short specific sequence following the target DNA sequence that is essential for cleavage by Cas nuclease (Synthego, 2025a).

The genomic target of the gRNA can be any 20 nucleotide sequence, provided it meets two conditions (Addgene, 2025):

1. The sequence is unique compared to the rest of the genome.
2. The target is present immediately adjacent to a PAM.

The PAM is about 2-6 nucleotides downstream of the DNA sequence targeted by the guide RNA and the Cas cuts 3-4 nucleotides upstream of it. In S. pyogenes, for example, Cas9 recognizes a 5'-NGG-3' PAM (where "N" can be any nucleotide base). However, the spacers in its CRISPR array are coded by 5'-GTT-3', so the Cas9 cannot cut the bacteria's own genome. Another key role of the PAM is guiding Cas nucleases to potential targets. Cas first scans DNA for the correct PAM, and only then checks if the adjacent sequence matches the guide RNA before initiating a cut (Synthego, 2025a).

Nucleases bind to its target sequence only in the presence of a PAM on the non-targeted DNA strand by the sgRNA. Consequently, the genome regions that can be targeted are limited by the locations of these PAM sequences. Nevertheless, Cas nucleases isolated from different bacterial species recognize different PAM motifs, expanding the range of editable sites across the genome (Synthego, 2025b). This diversity allows researchers to choose or engineer Cas variants best suited for specific genomic contexts (see table 2.1). For instance, the SpCas9 nuclease cuts 3-4 nucleotides upstream of the PAM sequence 5'-NGG-3' (where 'N' can be any nucleotide base), while the PAM sequence 5'-NNGRR(N)-3' is required for SaCas9 (from *Staphylococcus aureus*) to target a DNA region for editing (Synthego, 2025b).

CRISPR Nuclease	Organism Isolated From	PAM Sequence (5' to 3')
SpCas9	*Streptococcus pyogenes*	NGG
hfCas12Max	Engineered from Cas12i	TN and/or TNN
SaCas9	*Staphylococcus aureus*	NNGRRT or NNGRRN
NmeCas9	*Neisseria meningitidis*	NNNNGATT
CjCas9	*Campylobacter jejuni*	NNNNRYAC
StCas9	*Streptococcus thermophilus*	NNAGAAW
LbCpf1 (Cas12a)	*Lachnospiraceae bacterium*	TTTV
AsCpf1 (Cas12a)	*Acidaminococcus sp.*	TTTV
AacCas12b	*Alicyclobacillus acidiphilus*	TTN
BhCas12b v4	*Bacillus hisashii*	ATTN, TTTN, and GTTN

Cas14	Uncultivated archaea	T-rich PAMs (e.g., TTTA for ds-DNA); none for ssDNA
Cas3	In silico analysis of prokaryotes	No PAM sequence requirement

Table 2.1: Summary of Cas and other nuclease variants used in CRISPR experiments and their PAM sequences

It is worth mentioning that although PAM sequence is essential for binding, it should not be included in the sgRNA. Bacteria avoid cutting their own DNA by excluding the PAM sequence when integrating viral DNA fragments into their CRISPR array. This ensures that the Cas nuclease only targets foreign DNA containing a PAM. Following this principle, researchers typically design gRNAs without the PAM to prevent self-targeting, particularly in plasmid-based delivery systems. However, newer CRISPR applications are beginning to challenge this approach (Synthego, 2025a).

2.4.4 Off-Targets

CRISPR off-target editing refers to the non-specific activity of the Cas nuclease at sites other than the intended target, causing undesirable or unexpected effects on the genome. While these off-target effects are typically linked to known homologous sites, even precise, on-target edits can lead to issues like chromosomal translocations or chromothripsis (Roberts, 2025).

Wild-type CRISPR systems have a reasonable level of tolerance for mismatches between their target sequence and their guide RNA (gRNA) (also called 'promiscuous'). In fact, SpCas9 can tolerate between three and five base pair mismatches which can lead to double strand breaks in undesired sites in the genome if they bear similarity to the intended target and have the correct PAM sequence (Roberts, 2025).

Off-target effects can confound the results of your experiments and decrease repeatability. However, the level of risk depends largely on the intended application of the experiment and where the off-target edits occurs (coding vs non-coding areas). It is worth noting that achieving precise genome editing with CRISPR requires a careful balance between the Cas nuclease, guide RNA (gRNA), and the chosen delivery method (Roberts, 2025).

2.4.5 CRISPR Knockout

The most basic CRISPR mechanism is knockout which disrupts the activity of a gene. This mechanism consists of a double-strand break in the DNA induced by Cas9 and directed by a

gRNA specific to the gene of interest which then will be repaired through non-homologous end joining (NHEJ), an error-prone process that returns the Wild Type, produces insertions/deletions or Frameshifts (ideal for knockout) in the desired gene, disrupting its activity (see Figure: 2.6) (Addgene, n.d.).

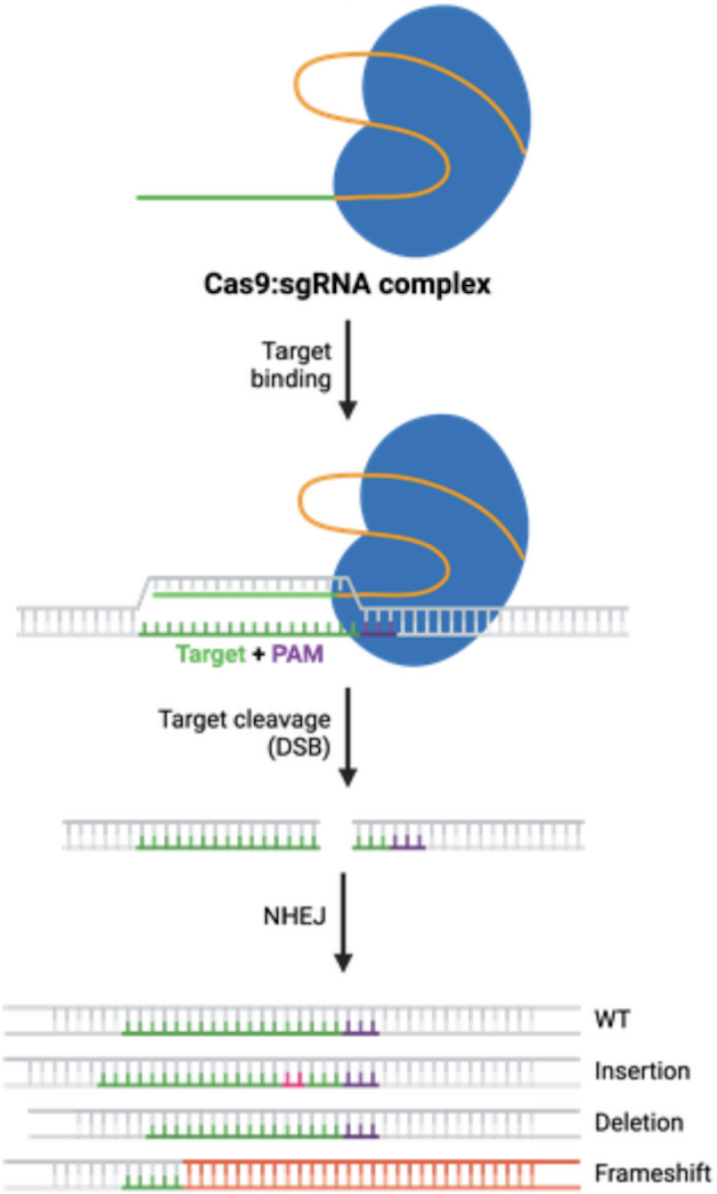

Figure 2.6: CRISPR Knockout schema by (Addgene, n.d.)

To fully grasp the impact of mutations, it's important to understand the concept of codon degeneracy. This refers to the redundancy in the genetic code, where most amino acids are encoded by more than one codon. A cell reads a gene's code in groups of three bases called codons during protein synthesis. Each of these "triplet codons" corresponds to one of 20 different amino acids used to build a protein. Additionally, specific stop codons signal the end

of protein synthesis. If a mutation occurs it may result in the addition of the wrong amino acids to the protein and/or the creation of a codon that stops the protein from growing longer (Adams, 2025).

Frameshift according to at the National Institutes of Health (2025) is 'An insertion or deletion involving a number of base pairs that is not a multiple of three, which consequently disrupts the triplet reading frame of a DNA sequence. Such variants (or mutations) usually lead to the creation of a premature termination (stop) codon, and result in a truncated (shorter-than-normal) protein product. Also called frameshift variant.'

Other insertions or deletions might not shift the reading frame — it just adds or removes amino acids. The protein may still be partially or fully functional. However, in most cases, NHEJ gives rise to small indels in the target DNA that result in amino acid deletions, insertions, or frameshift mutations leading to premature stop codons within the open reading frame (ORF) of the targeted gene producing a loss-of-function mutation within the targeted gene.

2.4.6 CRISPR Screening

Functional genetic screens are a powerful tool for understanding genetics. These screens analyze cellular phenotypes arising from genome-wide perturbations. There are two modalities of genetic modulation used in screens: gain-of-function or loss-of-function. CRISPR screening is a large-scale experimental approach used to screen a population of mutant cells to discover genes involved in a specific phenotype. Specifically, CRISPR introduces mutations to genes that render them nonfunctional (not post-transcriptional regulation).

After a set of genes is perturbed, a functional assay is used to qualitatively or quantitatively evaluate the effects of CRISPR-mediated knockouts. There are two broad categories of assays: binary and multiparametric. Binary assays identify cells based on the presence or absence of a desired phenotype while multiparametric assays measure multiple parameters simultaneously. Tecnhology advances (e.g., in imaging) have enabled assays to measure a wide range of cellular features, including morphology and protein localization.

Now that we've reviewed various functional assays, we can explore how they integrate into CRISPR screening workflows, which are typically carried out using one of two main formats: pooled or arrayed screens.

2.4.7 Pooled CRISPR Screens

In pooled CRISPR screens, a diverse library of sgRNAs is delivered into a population of cells using lentiviral vectors, with each virus typically encoding a single sgRNA. To ensure that most cells receive only one sgRNA, a low multiplicity of infection (MOI) is applied (commonly 5–30%). Upon infection, the lentiviral DNA—including the sgRNA sequence—is stably integrated into the host genome and subsequently transcribed to produce guide RNAs (Spencer, 2025).

For genome editing to occur, the Cas enzyme must also be present in the cells. This is achieved either by using a pre-engineered cell line that constitutively expresses the Cas nuclease or by co-delivering the Cas gene alongside the sgRNA in the same or a separate lentiviral construct. Lentiviruses pose biosafety concerns, yet they remain the vector of choice for CRISPR screening due to their ability to integrate stably into the mammalian genome. This stable integration ensures persistent expression of sgRNA across multiple cell generations. After editing, cells continue to carry and express these guide sequences, enabling researchers to track which sgRNAs were present in surviving or proliferating cells (Spencer, 2025).

Since all knockouts are performed in a single tube of cells, pooled screens cannot directly associate phenotypes with specific perturbations and therefore require binary assays to separate cells based on phenotype. After the cells are sorted, the integrated sgRNAs are sequenced via NGS revealing which genes are linked to the observed phenotype based on sgRNA enrichment or depletion

2.4.8 Arrayed CRISPR Screens

Traditionally, CRISPR screens use pooled lentiviral libraries targeting many genes at once. However, when researchers already have a short list of genes of interest, a different approach can be taken. In these cases, synthetic gRNA CRISPR libraries use arrayed formats where each well contains one or more guides targeting a specific gene. In this experimental design, gRNAs can be complexed with an appropriate Cas enzyme in each well and delivered into cells via electroporation or lipofection. Each well can then be monitored for phenotypic effects, such as cell viability or reporter expression (Spencer, 2025).

This method avoids the need for NGS but may require costly automation, especially with large libraries. By eliminating lentiviruses, only 1–3 well-characterized target sites per gene are needed, as there is no chance that lentiviral integration will disrupt an important genomic sequence. Avoiding lentivirus also removes biosafety requirements and assures that the synthetic gRNAs will not continue to exist in the cells after the knockout has occurred, decreasing the likelihood of off-target editing effects. This makes the method ideal for advanced screens focused on phenotypic changes observable via microscopy, and suitable for non-dividing cells like primary cells and neurons that aren't compatible with binary live/dead readouts used in pooled screens (Spencer, 2025).

It is worth noting that both pooled and arrayed screens are useful in screening workflows. For instance, if one aims to identify new drug targets, a pooled format may be appropriate as a primary screen to identify a broad set of target genes in simple but easy editing cell models while an array may be used as a secondary screen to validate the hits in more relevant cell models.

Chapter 3

State of the Art

The format of DNA sequences holds remarkable similarity to natural language since it is structured as multiple strings lined together. "In this analogy, each nucleotide in a sequence read is akin to a character, each read is akin to a sentence, and the entire genome is comparable to the full article" as mentioned by Ruan et al. (2025). Consequently, LLMs have been used to make genomes closer to the concept of 'word' in natural language through tokenization. As the volume of sequencing data continues to expand, these models have become increasingly valuable in the field of genomics for capturing complex patterns, supporting a range of applications including fitness estimation and sequence design.

In fact, LLMs are behind multiple tools in the field of genomics. For instance, some models are DNAGPT aimed at DNA analysis tasks with GPT as core model (Zhang et al., 2023), ChatNT, a Multimodal Conversational Agent for DNA, RNA and Protein Tasks (Richard et al., 2024), megaDNA which introduces a novel approach to modeling genomic sequences (Shao and Yan, 2024), UTR-LM designed for Decoding Untranslated Regions of mRNA and Function Predictions (Chu et al., 2024), GENA-LM that form a family of open-source foundational DNA language models for long sequences Fishman et al. (2025) and CD-GPT (Central Dogma - GPT), a model that aims to bridge the gap between molecular sequences through central dogma (Zhu et al., 2024).

An extended and comprehensive list of models, including models based on LLMs and others built on various transformer architectures, is presented in Benegas et al. (2025). This paper details key aspects of numerous models, including their pretraining data sources, tasks (such as Causal or Masked Language Modeling), as well as their architectural designs and tokenization methods.

Recently, multiple AI Agents have been developed for the genomics field (Zhou, 2024). One example is BioDiscoveryAgent, an agent for designing genetic perturbation experiments and GeneAgent: Self-verification Language Agent for Gene Set Knowledge Discovery using Domain Databases. It is also worth mentioning GenoTEX, a benchmark for evaluating LLM-Based exploration of gene expression data.

LLMs are opening new frontiers in gene editing by supporting innovation in areas like drug discovery and development. LLMs can aid experts in identifying novel and advancing gene editing strategies by driving forward de novo molecule generation, gene network analysis, binding site prediction and other related applications (Zheng et al., 2024).

For instance, LLMs could support the design of Adeno-associated virus (AAV) vectors - delivery vehicles in gene therapy that transport therapeutic genes into target cells - by summarizing scientific literature to identify novel strategies and by analyzing genomic sequences to predict the most effective vector sequences for safe and efficient gene delivery

Furthermore, LLMs could even assist in the design of the novo gene editing methods emulating the mechanism used by Yeh et al. (2023) who through a new deep-learning-based protein design strategy it named “family-wide hallucination”, which they used to make a unique light-emitting enzyme. In fact, the LLM Evo has been used synthetic CRISPR-Cas molecular complexes and transposable systems (Nguyen et al., 2024).

Due the broad implementation of AI agents in the field of genomics, there are limited examples of AI Agents specialized in the field of gene editing. In fact, to the best of our knowledge, only a model has been developed to this day: ”CRISPR-GPT”, an AI Agent that streamlines gene-editing experiment designs by breaking down the complex process into a sequence of the following manageable tasks (Huang et al., 2024):

- Selection of CRIPSR System
- gRNA Design
- Delivery Approach
- Prediction Off targets
- Recomendation of Experimental Protocols
- Validation Approach Recommendation and Primer Design

Additionally, this agent offers a Q&A and Off-target Mode. To do so, this model makes use of 4 core modules (LLM Planner, Task Executor, Tool Provider, LLM Agent) and predefined 4 metatasks: CRISPR Knockout, CRIPSR activation/interference, CRISPR Base Editing and CRISPR Prime Editing. This model tools include Google search, programs like Primer3, and resources such as guide RNA libraries, research papers, and experiment protocols.

Chapter 4

Materials and Methodology

The development of the AI agent was structured into three main phases. Firstly, the selection of appropriate tools and testing of their API calls was carried out, ensuring that each external service functions as expected and meets the project's requirements. Secondly, the agent was designed using the LLM 'gpt-4o-24-08-06' and the framework LangGraph, where its behavior and flow were defined. After, tool integration was performed, enabling the agent to interact with the selected APIs and external functionalities. Finally, the agent was evaluated through performance testing to assess its ability to complete specific tasks. The schema of figure 4.1 represents the steps followed to build the agent.

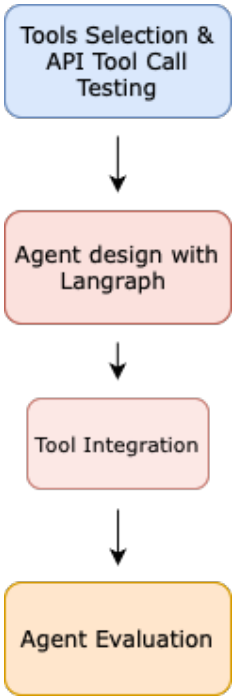

Figure 4.1: Methodology Phases

4.1 Tools Selection

A clear definition of the agent's capabilities was established through the selection of relevant data sources and required tool APIs. The analysis of free APIs was initialized with a tool list originally created by Liwei (2020) (last checked on 8/4/2025). A total of 11 tools with open APIs were obtained GT-SCAN2, CRISPY-WEB (Blin et al., 2016), CRISPRGenome (Rauscher et al., 2016), FORECasT (Allen et al., 2018), FORECasT-repair (Pallaseni et al., 2024), PETAL (Adikusuma et al., 2021), MinsePIE (Koeppel et al., 2023), CUNE (o'Brien et al., 2019), Forecast-BE (Pallaseni et al., 2022), CrisprCasTyper (Russel et al., 2020) and WGE (Hodgkins et al., 2015) (this last one encompasses multiple open API calls).

Multiple tools overlapped in tasks and only a few were available for techniques differing from CRISPR knock-out. As a result, a selection of the most relevant tools for CRISPR knock-out was carried out with the aim of covering at least one of the CRISPR techniques in detail. PETAL and MinsePIE were excluded due to their specific focus on prime editing, while CUNE and FORECasT-BE were similarly disregarded for their focus on base editing.

Subsequently, among the tools focused on CRISPR knock-out, GT-SCAN2—a CRISPR/Cas9 target scanning tool—was excluded from further consideration. This decision was based on its requirement for a larger number of input parameters (including genome, chromosome, start position, a FASTA file of the sequence, and cell line), which makes it less user-friendly compared to alternative tools. Moreover, its off-target analysis is limited to a maximum of 3 base pair mismatches, offering less comprehensive coverage than the WGE Off-Target tool.

CRISPY-WEB, a Cas9 sgRNA scanning tool, was also excluded due to its more complex input requirements compared to other tools. Specifically, it requires the user to either upload a GenBank file containing the target genome or provide an antiSMASH job ID to retrieve data, followed by the specification of a genomic region to scan for sgRNAs. This multi-step process makes it less user-friendly than alternatives such as the WGE CRISPR Search tool. Furthermore, its off-target analysis is limited to only 2 base pair mismatches, offering more limited coverage.

CRISPRCasTyper, a tool designed to detect CRISPR-Cas genes/arrays and to predict subtypes based on both Cas gene content and CRISPR repeat sequences, was also excluded from the selection. Its primary functionality was considered too distant from the core objectives and tasks addressed by the other tools. Additionally, only two API calls from WGE were selected: 'Find Off-Targets for Sequence' and 'CRISPR Search by Exon'. The former was selected over alternatives such as 'Off-Targets for CRISPRs' and 'Off-Targets for CRISPR Pairs', as it provides comparable functionality without requiring a WGE CRISPR ID, thereby simplifying the design workflow.

As a result, the following API calls were excluded due to their dependency on internal CRISPR identifiers, which would introduce unnecessary complexity: 'Find CRISPR ID for Sequence', 'Find CRISPR Sequence by ID', and 'Find CRISPR by ID'. Furthermore, the calls 'Find Marker Symbols for Search Term' and 'Find Exons for Marker Symbol' were omitted due to functional overlap with established reference resources such as NCBI Libraries.

Regarding the CRISPR search functionalities, 'CRISPR Pair Search by Exon' was excluded, as the project does not focus on CRISPR pair designs. Although 'CRISPR Search by Region' could have been a valuable addition, 'CRISPR Search by Exon' was prioritized instead, given the focus of other integrated tools on exons and gene symbols.

As a result, the final selection looked like this: CRISPRGenome, FORECasT, FORECasT-repair, WGE CRISPR Search by exon and WGE off target for sequence. Additionally, the list of tools was expanded with tools giving access to several databases, libraries and a web search tool. The following table show the selected tools (click on the different names to access their official webpage):

Tool	Use
CRISPR Knockout	
FORECasT	The most common use of CRISPR/Cas9 (SpCas9) editing is knock-out genes of interest through a double strand break. In this scenario, the FORECasT model contributes to determine the optimal sequence to be targeted in order to maximize the probability of creating a disrupting mutation via frame-shift in human and mouse genes (Allen et al., 2018). This tool given a sequence in FASTA format and its PAM (NGG) position returns a list of potential mutation outcomes and frequencies for each of them (the ten with highest scores). Additionally, it also offers a batch mode for analyzing multiple target sequences simultaneously. The model is accessible via GitHub for open use in the following link

FORECasT-repair	As specified by Pallaseni et al. (2024): "Genome engineering using CRISPR/Cas9 is a function of cellular repair and the availability of certain repair-associated genes can bias the outcomes of any editing experiment. Knockouts of non-homologous end-joining genes such as Lig4 and Xrcc5 have been shown to bias repair away from small insertions and deletions, while knockouts of microhomology-mediated end-joining or homologous recombination genes bias away from longer deletions. FORECasT-repair is a model of CRISPR/Cas9 editing outcomes in a variety of single-gene-deficient repair backgrounds. It predicts the distribution of insertions and deletions expected at a given target site in 20 knockout contexts to allow better planning of genome editing experiments." Consequently, this tool should be used when planning CRISPR/Cas9 experiments in repair-deficient contexts in human and mouse. It predicts how likely a variety of editing outcomes are at a given target site and given knockout, and the most common use-case is to determine which target sites are most efficient to knock-out a gene via frame-shift. The input is constituted by a sequence in FASTA format and the corresponding PAM (NGG - SpCas9). Additionally, in this tool, all the repair contexts desired must be indicated. This tool also provides a batch mode and its output is similar to the FORECast tool. This model is accessible via GitHub for open use in the following link

WGE CRISPR Search by Exon	This tool finds CRISPRs targeting a specific exon (Hodgkins et al., 2015) in contrast to tools that return sgRNAs for a gene (such as GenomeCRISPR) and its focus is on the CRISPR-Cas system exemplified by Cas9 from *Streptococcus pyogenes*. This API returns a list of potential sgRNA for a specific exon. For each potential sgRNA, detailed information is provided regarding its characteristics and "off-target" potential. In this project, the input it receives consists of the exon id and the genome (Human or Mouse / h38 - m38) and it returns an "id" field (unique identifier for each specific guide sequence), the chromosome name and the end/start positions of this sequence on the chromosome, respectively. The 'off target summary' offers a string-formatted overview of potential off-target sites, categorized by the number of mismatches. The 'exonic' indicates with a boolean value (1 or 0) whether the guide sequence resides within an exonic region. The 'species id' identifies the organism (Mouse = 2; Human = 4 in this context). The nucleotide sequence of the sgRNA is given by "seq". The 'pam right' field, a boolean value, denotes if the essential PAM motif is located to the 3' (right) or 5' (left) of the guide sequence. Finally, the Ensembl exon ID targeted by these guide sequences is also provided.
WGE Off Targets for Sequence	Fetch off-target summary and list of off-target CRISPR IDs for any 20 bp sequence (Hodgkins et al., 2015) for the CRISPR-Cas system exemplified by Cas9 from *Streptococcus pyogenes*. The input requires a sequence of 20 bp, selecting the species (Human or Mouse), and setting pam_right to true or false. The output returns and 'id', an off-target summary (consists of a list of 0 to 4 counting the number of mismatches between the CRISPR guide and potential off-targets; 0 indicates a perfect match) and a list of off-target IDs. If a CRISPR site is found in the genome that exactly matches the search sequence, then the ID of that sequence is provided. If there are multiple exact matches, the ID returned corresponds to the first match. If no exact match is found, the ID returned will be 0. In any case, the off-targets list provides WGE IDs of similar sequences with up to 4 mismatches. In case of obtaining an ID, the user will be asked if it desires a detailed summary of the related off-targets

Reference Resources	
NCBI Libraries	The NCBI Entrez system provides access to 38+ databases (Annotinfo, Assembly, BioCollections, BioProject, BioSample, Blastdbinfo, Books, CDD, ClinVar, dbVar, GAP, GAPPlus, GDS, Gene, GeoProfiles, Genome, GRASP, GTR, IPG, MedGen, MeSH, NLM Catalog, Nuccore, Nucleotide, OMIM, OrgTrack, PCAssay, PCCompound, PCSubstance, PMC, Protein, Protein Clusters, ProtFam, PubMed, SeqAnnot, SNP, SRA, Structure and Taxonomy), covering nucleotide/protein sequences, molecular structures, gene records, and biomedical literature. In this project, the input required consists of the selected database, the search term (e.g. brca1) and the following yes/no questions: "Filter by date?" and "Set max results?". The output consists of UIDs of the research papers associated with the search term that if desired will return summaries of the research papers (UID, Title, First and Last Author, DOI, Date and Journal).
Tavily Web Search	Tavily is the first search engine designed specifically for AI agents and large language models (LLMs). Tavily's Search API streamlines searching, scraping, filtering, and ranking relevant content from up to 20 sources—in a single API call. It uses proprietary AI to deliver precise, context-aware results tailored for AI applications. Developers can customize queries with additional context and limit response size for optimal performance.

Table 4.1: Gene editing selected tools/libraries and their specific use cases

4.2 AI Agent Design

4.2.1 Input

AI agents are capable of processing diverse forms of input data, including text, images, video, audio, and sensor data. Among these, text is one of the most fundamental and widely used input types.

Text inputs can range from simple commands and questions to more complex structures such as articles, reports, code snippets, or medical records. In addition to text, AI agents can process visual data (images and videos) for tasks like object and facial recognition, scene understanding, and activity detection. They can also interpret audio for speech recognition and emotion analysis. Sensor data process environmental or biometric information.

The AI agent for this project will exclusively accept text input. This is a direct consequence of the fact that the available tools are built around DNA/RNA sequences, which are themselves text-based. The subsequent sections detail the designed prompts for evaluation.

4.2.2 LLM

Noticeable foundational LLMs for AI Agents are being developed by major tech companies such as ChatGPT by OpenAI, Claude by Anthropic, and Gemini by Google. Additionally, Alibaba offers Qwen, Meta developed Llama, and DeepSeek is behind the R1 & R3 models.

It is worth noting that OpenAI, DeepSeek, and Google models are generally ranked at the top 3 LLM regarding intelligence and reasoning (Ivancie, 2025). According to this source, Llama and R1 outstand among open source AI Chatbots while OpenAI shows to provide the fastest and smartest models.

Among the OpenAI models, three modalities could be suitable for developing this project: Reasoning, Flagship, and Cost-optimized. The o1-mini, o1, o3-mini, and o1pro models stand out for their reasoning capabilities, excelling at complex, multi-step tasks. On the other hand, GPT-4o mini is recognized as a cost-optimized model. In between are the Flagship models, such as the GPT-4.5 preview and GPT-4o. As this project will not require multi-step high complex tasks, reasoning models were discarded. Additionally, cost-optimized models differ lightly from flagship models in pricing. Consequently, the model "gpt-4o-2024-08-06" was chosen since it provides the best price-quality ratio (Price per 1M tokens: input 2.50$; cached input 1.25$; output 10.00$).

GPT-4o ("o" for "omni") is a flagship model, which excels in tasks involving both text and image inputs, producing text outputs (including structured outputs). It has a 128,000 context window and a maximum output token capacity of 16,384. As the most capable model outside of our o-series models, GPT-4o is designed to handle a wide variety of tasks and is ideal for most applications. Its knowledge cutoff date is October 01, 2023.

4.2.3 Framework

The framework of an agent connects its different components to form the proper agent. Three outstanding frameworks are LangGraph, Autogen and CrewAI.

On one hand, a comparison between them highlithed that Autogen and CrewAI are more intuitive (see the following source: (Galileo AI, 2024)). On the other hand, LangGraph and CrewAi excel in most of the other evaluated sections: multi-agent support, tool coverage, memory support, structured output, caching, replay and available documentation (Galileo AI, 2024). However, LangGraph allows a more extensive customization and control of agents since CrewAI is a higher-level framework with many predefined features (Relari, 2024). Consequently, LangGraph was chosen for the development of this project.

LangGraph is an open-source library within the LangChain ecosystem for building, deploying, and managing complex AI agent workflows that uses graph-based architectures to model and manage relationships between the agent components. In this framework, the behavior of the agents is defined using three core components and optional supplementary components.

4.2.4 Core Components

An agent build in LangGraph has the following main components (LangGraph, 2024):

1. State: A shared data structure that represents the current snapshot of the agent. It is typically a TypedDict or Pydantic BaseModel.
2. Nodes: Python functions that encode the logic of agents. They receive the current State as input, perform some computation or side-effect, and return an updated State.
3. Edges: Python functions that determine which Node to execute next based on the current State.

According to Clark (2024), "The state serves as a memory tracking the processes of the agent workflow that both nodes and edges can access and modify during execution to retrieve or update information. State is inherently centralized — simplifying debugging and improving decision-making, scalability, and performance ". To build the graph, the state is first defined, followed by the addition of nodes and edges to establish the logic. The process is completed by compiling the graph into a functional workflow LangGraph (2024).

4.2.5 Supplementary Components:

The supporting features that enhance or provide functionality around the core structure of Langraph can be classified in (Clark, 2024):

- Monitoring mechanism: Human in the loop (HITL) refers to a collaborative process where humans augment the computational capabilities of machines to make informed decisions while building a model.
- Graph Architecture. Stateful graphs: A concept where each node in the graph represents a step in the computation, essentially devising a state graph. Cyclical graph: any graph that contains at least one cycle and is essential for agent runtimes.
- Tools: LangGraph integrates external tools and APIs to enhance the graph's functionality if needed. These tools include RAG, workflows, APIs and LangSmith, a specialized API for building and managing LLMs within LangGraph.

It is worth noting that a memory module—specifically a short-term memory—was integrated into the agent to help maintain contextual information about its current state and interactions.

4.3 Evaluation & Optimization

The evaluation criteria for generative AI agents producing text, such as those powered by LLMs, prioritize the coherence, topical relevance, and factual correctness of their responses. Conversely, predictive AI applications are assessed using metrics like precision, recall, and F1 score to determine the reliability of their predictions. Evaluation (or evals) methods differ widely, but it typically involves the following steps (Stryker, 2024):

1. Definition of goals and metrics: establishing the agent's purpose and metrics.
2. Collect expected data: gathering data to establish verifiable ground truth.
3. Conduct testing: Executing the AI agent in various environments (such as with or without memory) while tracking its performance.
4. Analyze results.
5. Optimization.

The evaluation process involves several key metrics that might differ among projects. In this case, the agent is fundamentally centered around tool interaction and the evaluation will primarily target this aspect. Consequently, the following key dimensions are expected to be monitored (Galileo AI, 2024):

1. System Metrics: This dimension focuses on resource usage and technical performance. The first metric expected to be measured is 'Cost per Tool Task completion' since it provides insights into the resource efficiency of the system, a critical factor for scalability. The second metric is 'Tool task completion time' since long time completion affects negatively user experience.

2. Tool Interaction: Incorrect tool selection would lead to task failures affecting negatively task completion effectiveness and leading to inefficient resource usage. Additionally, in case of selecting the correct tool, providing incorrect arguments would also lead to these challenges. Consequently, this dimension asses the effectiveness of the agent using the tools and it will be measured by tool selection and argument accuracy.

3. Quality Control: This dimension measures the accuracy and reliability of the agent's outputs. The key metric is the output accuracy success rate, defined as the proportion of outputs that meet predefined correctness criteria per tool.

Due to the agent's early development and domain-specific focus, a tailored evaluation framework was selected to assess its performance based on human evaluation which is often a nice starting point

To evaluate the mentioned metrics, a collection of test inputs and expected outputs was prepared (see 5.1). To comprehensively assess each tool's performance and behavior, the evaluation dataset is structured to include six carefully designed questions per tool. These questions are divided into two distinct categories and will have a pass/fail threshold ((AI, 2025)):

- **Clear-cut Queries (3 questions):**
  These are precise questions that the tool is explicitly designed to handle. They include all the necessary input parameters and are phrased clearly. Their goal is to test the tool's core functionality and determine whether it provides correct and reliable outputs under normal usage conditions. The answer given by the agent will be correct when it returns the exact answer obtained by the query on the official tool websites.

- **Incorrect Queries (3 questions):**
  These are intentionally flawed questions. They include missing parameters, unsupported formats, or misunderstandings about what the tool does. These queries are used to test the tool's error handling and ability to provide helpful feedback. A response was considered correct only if the agent accurately identified and alerted the user to the core issue in the query without being misled by other potential query issues. Failure to detect solely this primary problem resulted in the response being marked as incorrect.

Clear-cut questions were designed in base to the example queries provided by the different tools official websites. In case of Tavily, specific questions to gene editing were designed. The steps required by the agent per task will not be measured as the queries are straightforward and will only be considered correct if carried out in one step. It should be noted that the goal is establishing a reliable, measurable agent that can continuously be improved.

4.4 Deployment

Once the AI Agent was evaluated, the next step involved making it accessible through a user-friendly interface. For this purpose, Gradio was chosen. Gradio is an open-source Python library designed for quickly building web-based interfaces for machine learning models, APIs, or general Python functions (Gradio, 2025). It provides built-in functionality to instantly share applications via automatically generated links, eliminating the need for prior knowledge of JavaScript, CSS, or web hosting infrastructure. In this project, the web application was executed locally using a local URL. For demostration purposes, example interactions were recorded and uploaded to the corresponding GitHub repository.

Chapter 5

Results

The developed agent uses LangGraph as a framework. As a result, conversation flows through a series of nodes (specifically as a directed graph). The flow begins at the 'START node', which leads directly into the 'chatbot node', where GPT-4o-2024-08-06 model interprets the user's input and determines if any tools are required. If tool calls are identified, the graph routes execution to the 'tools node', which invokes APIs such as Elixir Forecast, WGE, and NCBI. The tool results are wrapped as ToolMessage's and passed back to the 'chatbot node', creating an iterative loop between chatbot & tools nodes until no further tool usage is needed—at which point the graph reaches the 'END node'. The entire process is checkpointed with print statements, execution time, token usage, and cost at each step. The agent code can be consulted at this link https://github.com/A-Aragon/TFM and it presents the basic architecture shown in figure 5.1. It is worth noting that the agent has short-term memory and remembers the previous query.

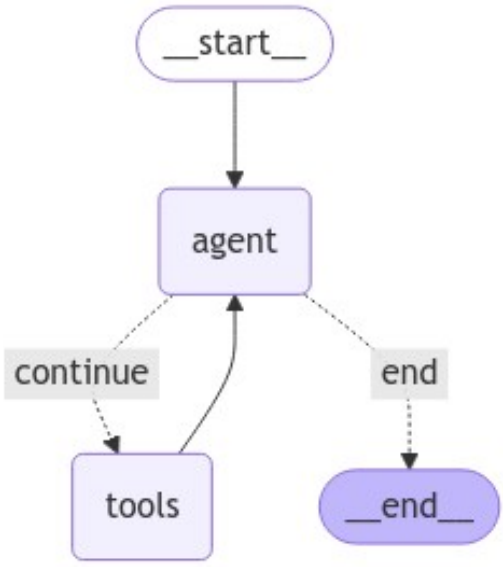

Figure 5.1: Agent Basic Architecture

5.1 Evaluation Dataset

The following query dataset was used to perform a preliminary evaluation aimed at verifying the correct functioning of the tool's API calls and identifying potential areas for optimization. The results have been presented in a concise manner to ensure clarity.

Herramienta: FORECasT		
Query type	**Query**	**Result**
Clear-cut	What mutations are predicted for the CRISPR Target sequence AATCGCC-TAGGTTTCGGCCTCAGTCCAACCAC-GAGAAGTACAGGGCTTTTACCTTTG-CAATCCGGATGGATGTGACGGA with a PAM at position 42?	Mutation outcomes id: 1. 1_L-2C1R0 2. D5_L-4C2R4 3. D3_L-4C1R1 4. D1_L-2R0 5. D9_L-10C2R2 6. D2_L-3R0 7. D12_L-13C2R2 8. D2_L-1C1R3 9. D11_L-11C2R3 10. D1_L-1R1
	What potential mutations would result from this target sequence AATCCG-TAAGCCTAGGTTTCGGCCTCAGTC-CAACCACGAGAAGTACAGGGCTTTTAC-CTTTGGCAATCCGGATGGAT with PAM in position 60?	1. D7_L-8C4R4 2. D1_L-2C1R1 3. D6_L-6C3R4 4. D2_L-1R2 5. D5_L-5C2R3 6. D2_L-3R0 7. D7_L-2C1R7 8. D13_L-10C3R7 9. I1_L-2C2R1 10. D3_L-1R3

	Show me predicted editing outcomes for this CRISPR target sequence AATCCGTAAGCCTAGGTTTCGGCCTCAGTCCAACCACGAGTCGAGTCAATGAGTACTGGGATGTGCTTGCATCATTACCTTTGGCAATC with PAM position 57.	1. I1_L-2C1R0 2. D3_L-4C1R1 3. D7_L-6C2R4 4. D9_L-5C2R7 5. D2_L-3R0 6. D1_L-2R0 7. D3_L-2C1R3 8. D10_L-3C2R10 9. D6_L-7C1R1 10. D2_L-1R2
Incorrect	What mutations are predicted for the CRISPR Target sequence AATCGCCTAGGTTTCGGCCTCAGTCCAACCACGAGAAGTACAGGGCTTTTACCTTTGCAATCCGGATGGATGTGACGGA with a PAM at position 17?	The agent should alert that the PAM position is not correct. Ideally, it should mention that the PAM provided does not match the real location of it in the sequence.
	What mutations are predicted for the CRISPR Target sequence AATCGCCTAGGTTTCGGCCTCAGTCCAACCACGAGAAGTACAGGGCTTTTACCTTTGCAATCCGGATGGATGTGACGGA?	The agent should alert the user of the lack of PAM, a required argument.
	What mutations are predicted for the CRISPR Target sequence and PAM 42?	The agent should alert the user that the target sequence argument lacks.
Herramienta: FORECasT-repair		
Query type	**Query**	**Results**

Clear-cut	What are the predicted mutation outcomes for the CRISPR target sequence AATCGCCTAGGTTTCGGCCTCAGTCCAACCACGAGAAGTACAGGGCTTTTACCTTTGCAATCCGGATGGATGTGACGGA, with the PAM located at position 42, in the Lig3 knockout repair context, and is it possible to compare them against a control condition?	Lig3 repair context: 1. I1_L-2C1R0 2. D5_L-4C2R4 3. D9_L-10C2R2 4. D1_L-2R0 5. D12_L-13C2R2 6. D3_L-4C1R1 7. D28_L-25C3R7 8. I1_L-1C1R1 9. I1_L-1R0 10. D20_L-20C3R4 Control repair context: 1. I1_L-2C1R0 2. D5_L-4C2R4 3. D9_L-10C2R2 4. D12_L-13C2R2 5. D3_L-4C1R1 6. D1_L-2R0 7. D11_L-11C2R3 8. I1_L-1C1R1 9. I1_L-1R0 10. D28_L-25C3R7

	Which mutations are predicted by FORECasT-repair for the target sequence AATCGC-CTAGGTTTCGGCCTCAGTCCAACCAC-GAGAAGTATGCTCATACGGGCTTTTAC-CTTTGCAAT, given a PAM at position 49 and assuming an Ercc1 knockout repair context?	1. I1_L-2C1R0 2. I1_L-1C1R1 3. D10_L-6C3R8 4. I1_L-1R0 5. D8_L-9C2R2 6. D6_L-8C2R1 7. D17_L-17C3R4 8. D3_L-4C1R1 9. D4_L-3C1R3 10. D2_L-2C1R2
	Could you retrieve the predicted mutation outcomes for the CRISPR target sequence AATCGCCTAG-GTTTCGGCCTCAGTCCAACCAC-GAGAAGTATGCTCATAGCTAATCC-TACTGGGTGCAGACTTTTACCTTTGCAAT, using PAM at position 59 and the Poll-deficient repair background?	1. D3_L-4C1R1 2. D6_L-8C3R2 3. D3_L-2C2R4 4. D1_L-3C1R0 5. D10_L-11C2R2 6. D4_L-3C1R3 7. I1_L-3C2R0 8. D1_L-1R1 9. D9_L-8C2R4 10. D30_L-30C2R3
Incorrect	Could you predict the CRISPR editing outcomes for the target sequence AATCGC-CTAGGTTTCGGCCTCAGTCCAACCAC-GAGAAGTACAGGGCTTTTACCTTTG-CAATCCGGATGGATGTGACGGA, using PAM at position 42, under the Lig5 knockout repair context?	Ideally, the agent should alert the user about an invalid repair context.

	What mutations are predicted for the CRISPR target sequence AATCGCCTAG-GTTTCGGCCTCAGTCCAACCACGAGAAG-TACAGGGCTTTTACCTTTGCAATCTA with a PAM at position 55?	The agent should alert the user that the PAM at position 55 is CTA, which is not a valid NGG PAM required for SpCas9 targeting since FORECasT-repair only supports canonical NGG PAMs.
	Can FORECasT-repair predict mutation outcomes for CRISPR edits under the umbrella of Single Strand Annealing (SSA) repair context?	The agent should alert the user about an invalid repair context (it is not included in the tool)***.
Herramienta: WGE CRISPR Search		
Query type	**Query**	**Resultado**
Clear-cut	Retrieve CRISPR sgRNAs targeting exon ENSMUSE00000233752 of the mouse gene CCT2, including off-target summary and the IDs	sgRNAs id: 331073795; 331073796; 331073797; 331073798; 331073799; 331073800; 331073801; 331073802; 331073803; 331073804; 331073805; 331073806; 331073807; 331073808.
	Show me the sgRNAs for the Human (gen: BRCA1), specifically targeting exon ENSE00003510592.	1147803906; 1147803907; 1147803908; 1147803909; 1147803910; 1147803911; 1147803912; 1147803913; 1147803914; 1147803915; 1147803916; 1147803917; 1147803918.
	Show me the sgRNAs for the Human (gen: APOE), specifically targeting exon ENSE00001048576.	1166705471; 1166705472; 1166705473; 1166705474; 1166705475; 1166705476; 1166705477; 1166705478; 1166705479; 1166705480, 166705481; 166705482; 166705483; 166705484; 166705485; 166705486; 166705487; 166705489; 166705488.

Incorrect	Could you return sgRNAs for the gene BRCA1 in Humans?	The agent should notify about an invalid input format since the tool requires an exon Ensemble ID, not a gene name.
	Retrieve sgRNAs for human (grch38) exon ENSMUSE00000233752 (CCT2 gene), including their off-target summaries and unique IDs	The agent should notify the user that the provided exon ID is not valid for the selected species, as it corresponds to mouse.
	Could you retrieve sgRNAs for zebra fish?	The agent should notify the user that zebrafish is not a species option
Herramienta: WGE Off targets		
Query type	**Query**	**Results**
Clear-cut	Which off-target sites are predicted for the guide sequence AAATGGGTGGGAGGCAGGGT in the human genome, with the PAM located on the right?	Off target summary - 0: 1, 1: 1, 2: 4, 3: 89, 4: 885 (id: 922349330). Species: 4.
	Please identify potential off-target sites for the guide RNA sequence ATGCTGACTAAGCAGCTTGA in the human genome, assuming the PAM is located on the right side.	Off target summary - 0: 0, 1: 0, 2: 0, 3: 5, 4: 112 (id: 0). Species: 4.
	Could you return the off-target predictions for the sequence ATGCGGGCTATGCACATTAA in mouse, with the PAM positioned on the left?	Off target summary - 0: 0, 1: 0, 2: 0, 3: 2, 4: 81 (id:0). Species: 2.
Incorrect	What off-targets are predicted for the following guide RNA sequence 'ATGCTGACTAAGCAGCTTGAATGCCTGAA'?	The agent should alert that the guide RNA sequence is longer than 20 bp.
	Could you retrieve off-target information for a 20bp CRISPR guide in Drosophila?	The agent should indicate that Drosophila is unsupported, with the tool limited to Human and Mouse species only.

	What are the predicted off-target sites for the sequence ATGCTGACTCGGCAGCTTGC in humans?	The user should be informed that the input is missing the PAM orientation (left or right).
Herramienta: NCBI Libraries		
Query type	**Query**	**Results**
Clear-cut	I want to search the 'PubMed' database for 'Brca1'. I also want to filter the results specifically by publication date between 2024 and 2025, ensuring they are primary data. I do not want the results sorted in any particular order. Please do not set a maximum number of results, but do provide summaries of the research papers found	DOIs: 1. 10.14670/HH-18-940 2. 10.3389/fonc.2025.1477910 3. 10.2147/IJN.S491473
	I want to search the 'Genome' database for 'human'. Please do not filter by date, and don't apply any sorting to the results. I only want 1 result back, and I'd like to receive its summary.	1. Assembly: GRCh38.p14, 2. GenBank id: GCA_000001405.29, 3. Scientific name: *Homo sapiens* 4. Release date: Feb 2022
	I would like to search the 'Protein' database for 'p53'. Please do not filter the results by date. I want to limit the results to 2 items and have them sorted by relevance. Additionally, please provide result summaries	Summaries: 1. p53 [*Penaeus japonicus*], 461 aa protein, Accession: BAL15075.1 GI: 357933379 2. p53, partial [*Canis lupus familiaris*], 281 aa protein, Accession: AAC37335.1 GI: 1463021

Incorrect	Search the 'Pubmat' database for 'APOE'	The agent alerts the user that the specified database does not exist or correctly identifies the intended database.
	Search the Gene database for the term 'GBA' with max results = 5	For "Set max results?", the input must consist of a yes/no response and a number. If something else is entered, the chatbot should ask the user if they may have meant 'y' and then a value like 5 or assume that.
	Could you search the Protein database for entries related to P53 and filter the results by date using the year 2024? Also, limit the results to 1.	The "Filter by date?" field in the Entrez tool expects a yes/no response—not a specific year like 2024. If a year is provided instead, the chatbot should either ask the user if they meant 'y' and then a value of 2024 or assume the user meant "yes" and proceed with the query filtered by that year (obtained output: *Daphnia magna*, 413 aa protein, Accession: XAX24432.1, GI: 2734519602).
Herramienta: Tavily		
Query type	**Query**	**Resultado**
Clear-cut	When was CRISPR-Cas9 system designed?	The agent should mentioned the paper with the following DOI: 10.1126/science.1225829
	Which was the first gene editing therapy approved by FDA (use Tavily)?	'Casgevy'

	List of gene editing therapies approved by FDA?	it should return the following list: https://www.fda.gov/vaccines-blood-biologics/cellular-gene-therapy-products/approved-cellular-and-gene-therapy-products
Incorrect	Retrieve protein sequence for Cas9 from Streptococcus pyogenes. Number of exons gene BRCA1? Available tools in this agent?	This question is better suited for NCBI tool. This question is better suited for 'gene' database from NCBI. It is a question for the agent

Table 5.1: Listado de preguntas clasificadas por herramienta, tipo y resultado.

5.2 Evaluation Key Findings

The following charts summarize the results obtained from applying the evaluation dataset to the agent.

5.2.1 System Metrics

Average Time Per Task:

The figure 5.2 shows that Clear-cut questions (C) generally require more time as expected. These tasks typically involve calling an external tool and interpreting the results via the chatbot node. As the number of steps increases, so does the time required to complete the task.

In contrast, Invalid queries (I) are usually handled directly by the chatbot, allowing for faster resolution. An exception to this pattern is Tavily, where the invalid queries were specifically designed to invoke NCBI library calls, rather than being resolved by the chatbot node resulting in a higher time per task. Is is also woth noting that only the question 'Retrieve protein sequence for Cas9 from Streptococcus pyogenes' requires 115 seconds while the others 6.8 s y 4 s.

It is also worth noting that Clear-cut questions involving NCBI libraries take significantly more time compared to similar tasks with other tools. This is because answering these questions

often requires multiple NCBI API calls as the results from some of them are required by others, which substantially increases the total processing time per task.

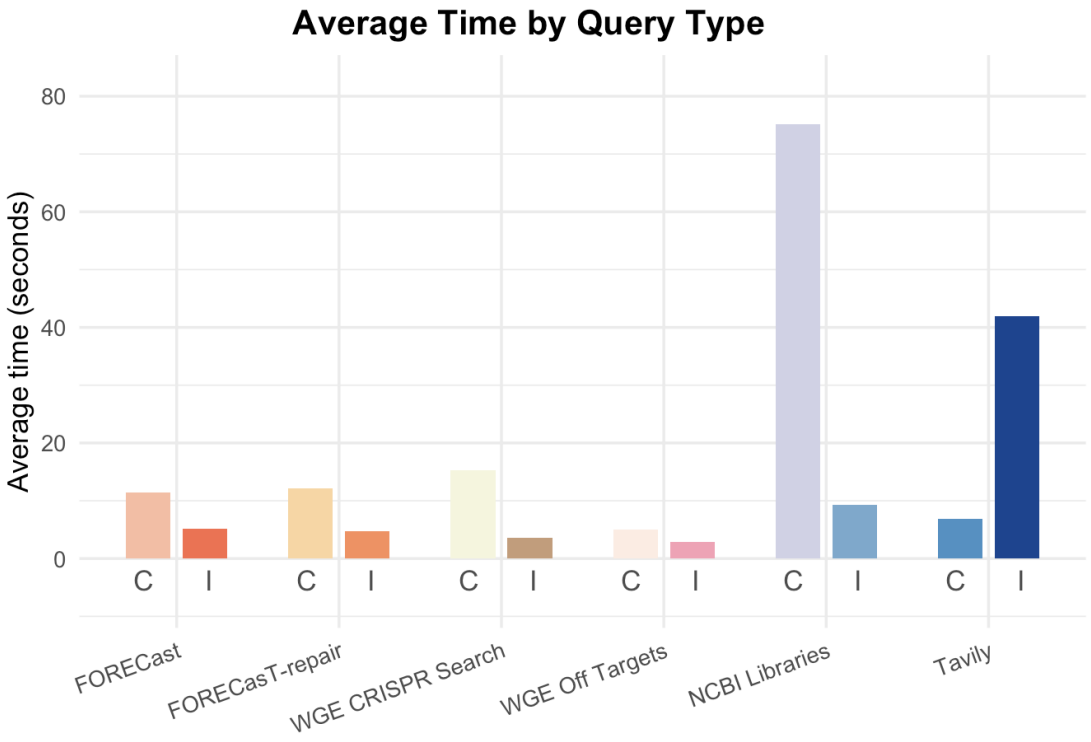

Figure 5.2: Time per task

Average Cost Per Task:

The average cost per task does not exhibit a pattern as clear as the one presented by the average time discussed in the previous section. However, as shown in Figure 5.3, Clear-cut questions still tend to incur higher costs, particularly when using tools like WGE CRISPR Search and NCBI libraries.

This higher cost is might be a result of the way the evaluation was conducted: it involved printing each step required to answer a question, which results in longer and more verbose outputs, especially for these two tools.

Once again, it is important to consider that for Tavily, some of its Invalid queries are deliberately designed to call NCBI libraries. In fact, the first invalid query — "Retrieve protein sequence for Cas9 from Streptococcus pyogenes" — accounts for a substantial portion of the total cost, due to the resource intensive nature of the API calls involved.

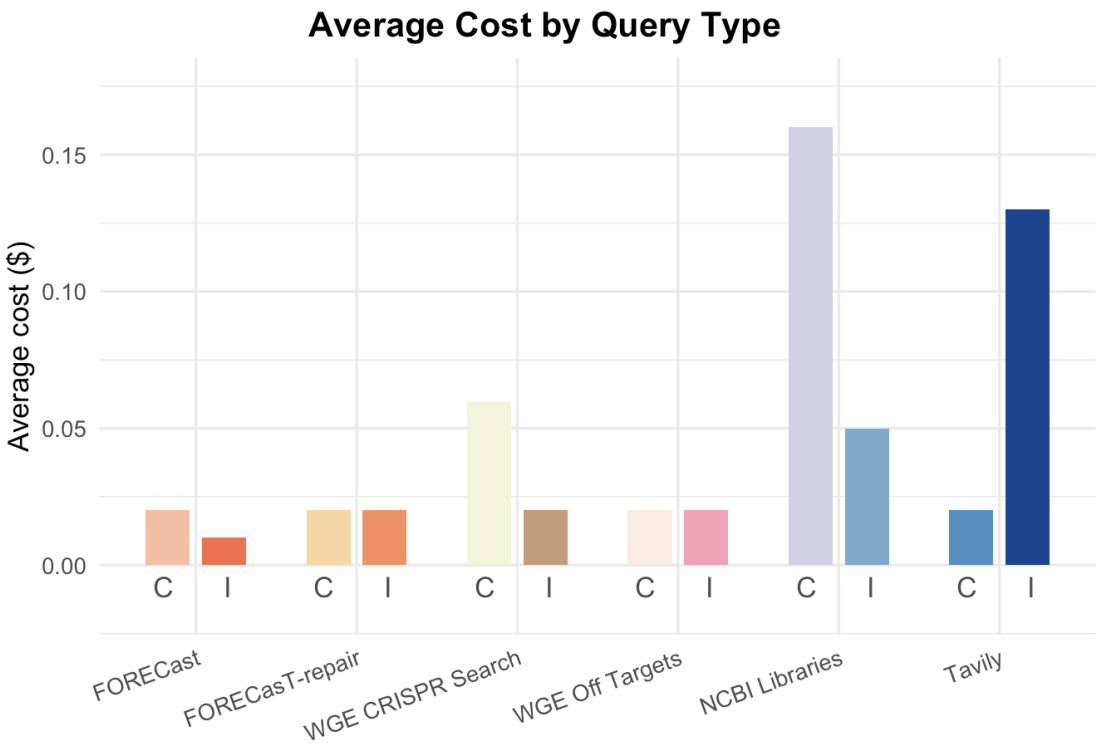

Figure 5.3: Cost per task

5.2.2 Tool Interaction & Argument Accuracy

The following visualizations are organized by tool and focus exclusively on Clear-cut questions, as these are the ones that consistently require tool interaction. In contrast, most Invalid queries are designed to assess the LLM's ability to handle misleading or incorrect prompts, and often do not involve tool calls.

Across all Clear-cut questions, the agent correctly selected the appropriate tool and provided the correct arguments. Only two answers were incorrect, but even in those cases, the correct tool was called with appropriate arguments (see figure 5.4). An exception occurred with Tavily, where the agent failed to make a tool call for one Clear-cut question — "When was the CRISPR-Cas9 system designed?" — and instead answered directly. Because no tool was called, the argument accuracy could not be evaluated for that instance (see figure 5.5).

This highlights a potential flaw in the prompt design for Tavily-related queries. Specifically, it suggests that the prompts may not always require tool use, as the chatbot node is sometimes capable of generating accurate answers without external assistance. Consequently, Tavily presents unique challenges for evaluation, as it differs from other tools whose Clear-cut queries cannot be reliably answered without external calls.

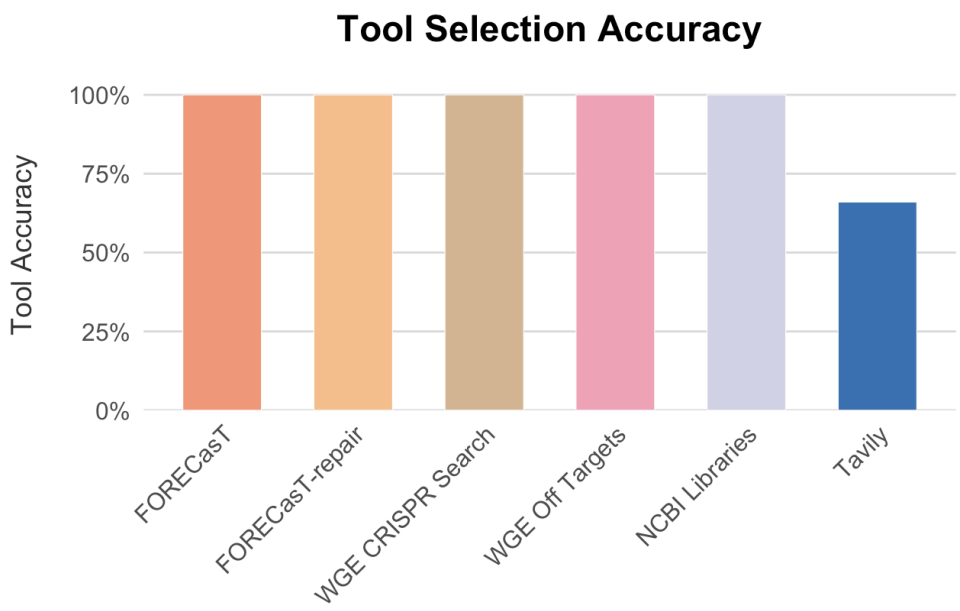

Figure 5.4: Tool Selection

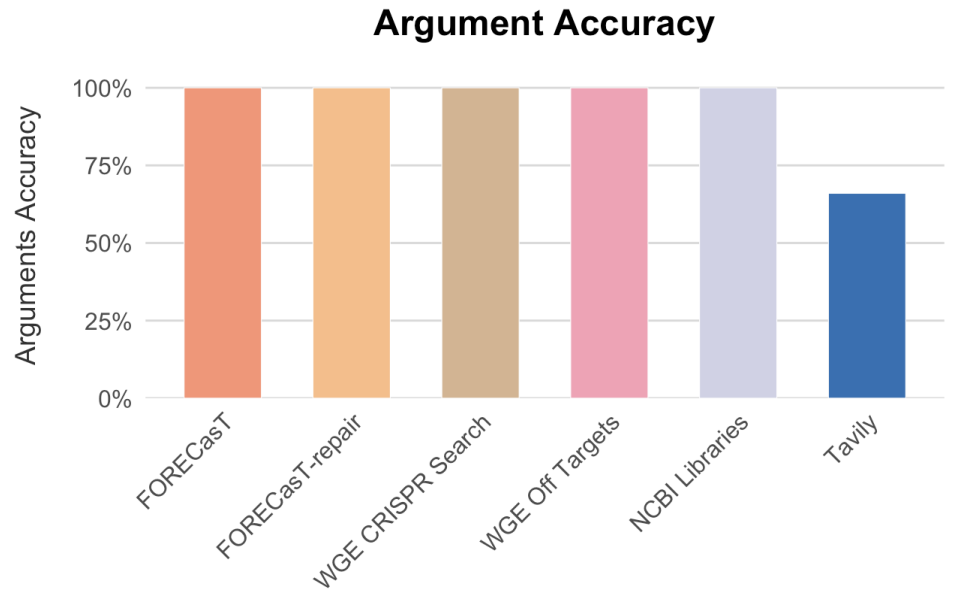

Figure 5.5: Argument Accuracy

5.2.3 Quality Control

This section examines the overall accuracy of the agent. As illustrated in Figure 5.6, the agent shows higher accuracy on Clear-cut questions. This outcome is expected, as the agent's responses to these tool-based queries were explicitly designed and guided by the author, allowing for more controlled and predictable behavior.

Accuracy drops notably in Invalid queries for tools such as FORECasT-Repair and WGE Off-Target. In the case of FORECasT-Repair, this may be a result of its Invalid queries designed to test the detection of unsupported repair contexts—information the agent does not recognize. As a result, the agent incorrectly proceeds to call the FORECasT-Repair tool and handles the lack of output without a defined clarification. This issue could potentially be mitigated by providing the agent with knowledge of the repair contexts supported by the tool, enabling it to produce more accurate and well-informed responses.

For WGE Off-Target, similar limitations are observed. In the second Invalid query"Could you retrieve off-target information for a 20bp CRISPR guide in Drosophila?" the agent fails to recognize that Drosophila is not a supported specie, and proceeds with the tool call. In the third Invalid query — "What are the predicted off-target sites for the sequence ATGCTGACTCG-GCAGCTTGC in humans?", the agent assumes a PAM is present in the right position, rather than asking the user for clarification when it is missing.

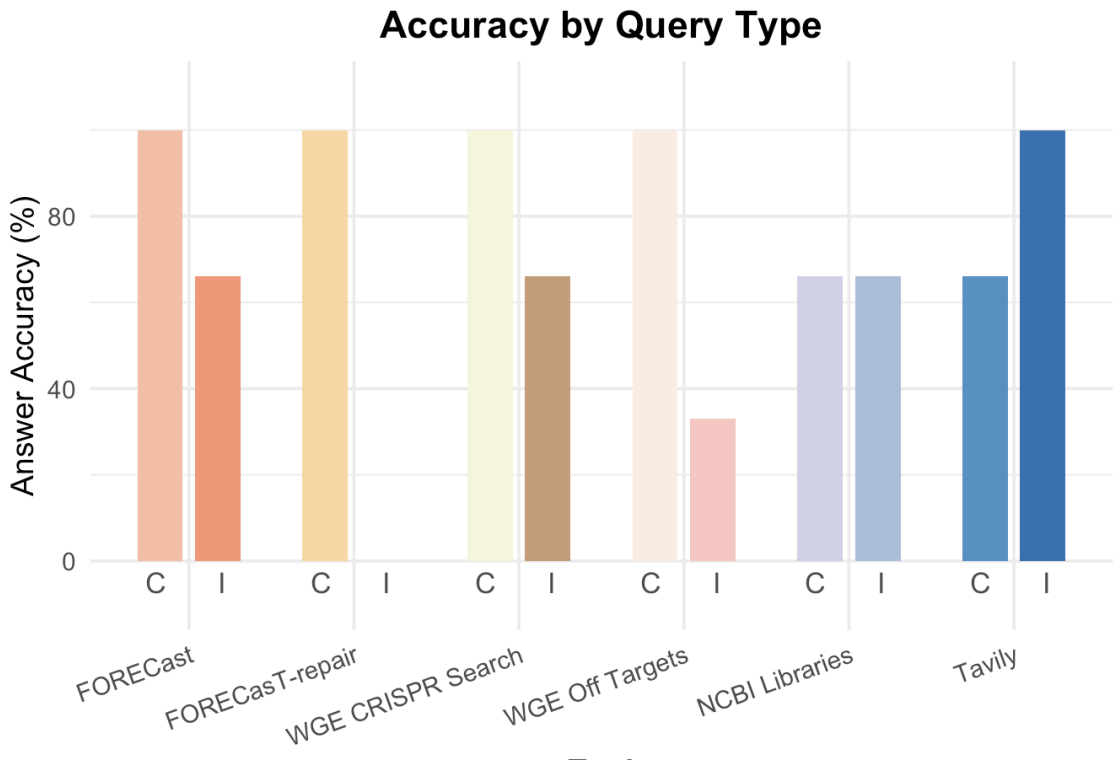

Figure 5.6: Accuracy type of question and tool

Finally, it is worth noting that this initial evaluation yielded a 72.2% accuracy rate, with the majority of correct answers corresponding to Clear-cut questions (see figure 5.7).

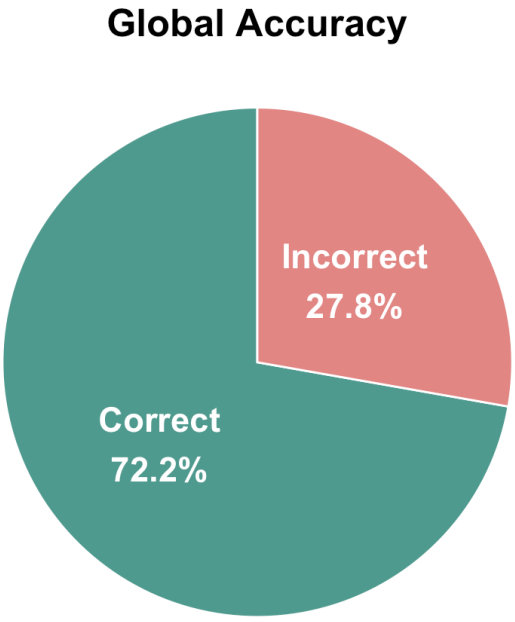

Figure 5.7: Accuracy by tool

Chapter 6

Discussion

Currently, to the best of our knowledge, only a model has been developed to this day in the field of gene editing: 'CRISPR-GPT'.

CRISPR-GPT covers the following CRISPR techniques: Knockout, activation/interference, Base Editing, and Prime Editing. In contrast, our agent focuses on CRISPR Knockout and specifically on the SpCas9 CRISPR System, due to the broader availability of freely accessible API tools for this technique compared to the others. The agent developed in this project aims to complement the CRISPR-GPT module on the Knockout technique and contribute to the democratization of intelligent gene editing agents. CRISPR-GPT assists users throughout the following stages of CRISPR experimental workflows:

1. Selection of CRIPSR System
2. Delivery Approach
3. gRNA Design
4. Prediction Off targets
5. Recommendation of Experimental/Validation Protocols

The agent developed in this project does not allow for CRISPR system selection. Similarly, delivery methods are not addressed, as their selection is highly dependent on project specifics and currently there is a lack of a direct, standardized tool for determining the most suitable option - unlike gRNA design, for which a well-established tool is integrated. Instead, users are encouraged to explore and evaluate delivery strategies by consulting the libraries and web-based resources integrated into the agent. These resources can help users identify the main differences between delivery methods and know which delivery methods have been applied in similar research contexts, providing similar insights to those typically offered by CRISPR-GPT.

Our agent covers gRNA design and off-target prediction in a very direct manner using tools with accessible APIs since these are key steps in CRISPR experiments. Although experimental

and validation protocols are not directly integrated, users can still access relevant recommendations by exploring various NCBI databases integrated into the agent's toolset and through Tavily, a search engine tailored for AI agents.

Building on these core features, the agent further expands its utility with capabilities not available in CRISPR-GPT. It incorporates specialized tools for predicting editing outcomes, such as FORECasT and FORECasT-repair, which estimate the likelihood of various mutation outcomes at a given target site. These tools help determine the most effective sites for gene knockout via frameshift mutations—FORECasT under standard conditions and FORECasT-repair in DNA repair-deficient contexts.

All these features frame the agent as a tool primarily focused on the bioinformatic phase of CRISPR experimentation, particularly the rational design and optimization of gRNAs, offering strong support for early-stage planning before moving to laboratory implementation. Lastly, while CRISPR-GPT offers two modes; one structured around step by step guidance based on the mentioned predefined tasks, and another focused on customized guidance on free style user request — our agent primarily emphasizes customized guidance. This design choice reflects a focus on adaptability, allowing users to engage with the system through natural, open-ended requests that align more closely with diverse and evolving research needs.

Chapter 7

Conclusions

Recent research has focused on enhancing LLM problem-solving by integrating external tools (Xi et al., 2025) leading to the rise of revolutionary AI agents. Traditional Agents were deterministic, designed to excel in specific tasks relying on a set of rules. Consequently, they had limited adaptability and used to struggle with tasks outside their initial scope. However, the integration of LLMs in AI agents transformed them into probabilistic and flexible agents capable of adjusting to new situations, integrating various tools and learning from fault behavior (Zhao et al., 2023).

Recently, several AI agents have been developed for applications in genomics (Zhou, 2024). However, there are still very few examples tailored specifically to gene editing. In fact, to the best of our knowledge, only one such model has been developed to date: CRISPR-GPT, an AI agent that facilitates gene-editing experiment design by decomposing the complex workflow into a series of manageable tasks—pursuing a goal similar to that of the agent proposed in this project.

This highlights a significant opportunity for innovation in the development of AI agents dedicated to gene editing. As the field continues to take off, there is ample room for new contributors and technological advancements. Nevertheless, in order to realize this potential, broader access to open-source tools and APIs is essential.

The present project aims to provide an alternative to CRISPR-GPT, broadening the range of available design assisting tools in the field of gene editing. However, its utility is currently limited by the availability of tools, focusing only on knock-out technique. Consequently, there is a strong need for the development of open and cutting-edge techniques in this field.

Chapter 8

Limitations & Future directions

8.1 Limitations

The development of the agent was shaped by practical constraints typical of early-stage academic projects, particularly regarding time/human resources and low-cost/free tools availability. As a result, multiple CRISPR systems and detailed guidance on certain elements of CRISPR systems were left out from the scope of this initial version. Nevertheless, the agent was designed in the CRISPR-Cas9 knockout technique, ensuring utility despite these limitations.

Additionally, to the best of our knowledge, as of 20/05/2025, it is no longer possible to access the Genome CRISPR API, and the tool's website has also gone offline. This unexpected development affected the final stages of agent construction and testing, highlighting the dependency of this project on third-party services that may become unavailable without notice.

Furthermore, the current evaluation is partially subjectivity, as the agent construction is in an early phase of development and in a highly specialized, narrow domain. The evaluation primarily relied on task-based assessments designed by the author and manual analysis, which, while useful for rapid iteration, limit reproducibility and scalability. Consequently, the main limitations fall into two categories:

8.1.1 Tools Coverage:

- **Limited CRISPR Techniques coverage**: The agent is designed for CRISPR knockout applications only. Other modalities such as CRISPR activation/interference (CRISPRa/i), base editing, and prime editing are not currently covered.
- **Cas Nucleases**: The agent focuses specifically on the classical CRISPR-Cas9 system, particularly the SpCas9 nuclease from *Streptococcus pyogenes*. Consequently, there is no support for alternative Cas proteins (e.g., Cas12, Cas13, or SaCas9).

- **Species Coverage**: The tools integrated into the agent are primarily optimized for use in human and mouse models.
- **GenomeCRISPR**: GenomeCRISPR is a database for high-throughput CRISPR/Cas9 screening experiments. Currently, GenomeCRISPR contains data on the performance of approximately 700 000 single guide RNAs (sgRNAs) used in 500 different experiments performed in 421 different *human cell lines*. This database API stop being accessible without prior notice.

8.1.2 Evaluation

Frameworks are great toolkits for conducting LLM evaluations with custom configurations while Benchmarks are standardized tests that provide comparable results for a variety of models. According to Symflower (2024), think of how many seconds it takes a sports car to reach 100 km/h. That is a benchmark with which you can compare different models and brands. But to obtain that numerical value, you'll have to deal with all the equipment (i.e. a precise stopwatch, a straight section of road, and a fast car to measure). That's what a framework provides.'

- **Absence of Agent Evaluation Benchmark:** Agent benchmarks provide a structured approach to evaluating the capabilities of LLM-based agents by combining three essential components: a set of well-defined tasks, a controlled operating environment (simulated or real), and performance metrics such as success rate, efficiency and accuracy (Yehudai et al., 2025).
- **Absence of Agent Evaluation Framework:** Evaluation platforms are essential for assessing agent performance, as they provide continuous monitoring of agent trajectories while capturing key metrics such as task completion rate, latency, execution speed, and, in some cases, throughput, memory usage, and observability (Yehudai et al., 2025). There are multiple platforms designed to optimize and monitor AI agents, with standout options including LangSmith (Inc. LangChain, 2023), LangFuse (Langfuse, 2023), and Arize (Arize AI, Inc., 2025) (See figure 8.1 for a comparison).
- **Absence of Professionals Evaluation:** It would also be valuable to assess the AI agent's overall effectiveness through feedback from multiple users with expertise in the field of gene editing.

Framework	Stepwise Assessment	Monitoring	Trajectory Assessment	Human in the Loop	Synthetic Data Generation	A/B Comparisons
LangSmith (LangChain)	✓	✓	✓	✓	×	✓
Langfuse (Langfuse)	✓	✓	×	✓	×	✓
Google Vertex AI evaluation (Google Cloud)	✓	✓	✓	×	×	✓
Arize AI's Evaluation (Arize AI, Inc)	✓	✓	×	✓	✓	✓
Galileo Agentic Evaluation (Galileo)	✓	✓	×	✓	×	✓
Patronus AI (Patronus AI, Inc.)	✓	✓	×	✓	✓	✓
AgentsEval (LangChain)	×	×	✓	×	×	×
Mosaic AI (Databricks)	✓	✓	×	✓	✓	✓

Figure 8.1: Overview of Evaluation Support in Major Agent Frameworks (Yehudai et al., 2025)

.

8.1.3 Deployment

To keep a Gradio app—or any web application—running continuously and accessible via a public link without relying on a local machine, deployment to a hosting service is required. Common options include Hugging Face Spaces, which is widely used and particularly suitable for Gradio apps, as well as Google Cloud Run, AWS Lambda or EC2, Heroku (noting recent changes to its free tier), and PythonAnywhere.

However, deploying to these services requires technical resources, infrastructure knowledge, and often financial investment, which were beyond the scope and capacity of this project.

8.2 Future Research Lines

8.2.1 Tools Coverage

Future work should focus on addressing the current limitations by expanding support beyond human and mouse models, integrating additional CRISPR techniques/systems, and incorporating tools for delivery method selection and experimental validation.

8.2.2 Evaluation

A comprehensive evaluation of an AI agent should ideally include systematic comparisons across multiple agents or baselines, with quantitative benchmarks that are reproducible, generalizable, and able to scale across broader domains and use cases. This remains a goal for future work as the agent matures and broader datasets and evaluation frameworks become available.

8.2.3 RAG Integration

Additionally, it would be pretty interesting to add Retrieval Augmented Generation (RAG) into the agent. RAG is an architecture that optimizes the performance of LLMs by connecting them to external knowledge bases. LLM training datasets are finite and limited to the information the AI developer can obtain - mainly publicly accessible data. The data in the RAG knowledge bases provides domain-specific knowledge so the LLM can generate more accurate responses without fine-tuning.

As a result, RAG facilitates cost-efficient AI implementation and scaling by leveraging existing knowledge bases rather than requiring extensive model retraining. This approach also

grants access to current, domain-specific data, thereby significantly lowering the risk of AI hallucinations and consequently increasing user trust in the generated responses. Furthermore, RAG expands the potential use cases for large language models, provides developers with enhanced control over models, simplifies maintenance, and contributes to greater overall data security.

These improvements would aim to enhance the agent's capabilities, scalability, and applicability across a broader range of CRISPR-based research contexts.

Chapter 9

Planning Follow-up and Monitoring

The project followed the planned timeline overall. However, some tasks took considerably longer than expected. For instance, selecting the tools and integrating their APIs, which extended into Phase 2.

Similarly, the agent's design and evaluation phases required more time than anticipated, spanning Phase 3 and continuing through the entirety of Phase 4. Additionally, the evaluation remained relatively preliminary and could have been approached in a more in-depth manner.

Regarding the environmental, ethical-social, and diversity impacts, it should be highlighted that the environmental footprint was lower than expected, primarily due to the limited scale of the agent.

Finally, the direct financial cost—excluding human labor—was minimal, with only around $12 spent on usage of the selected large language model (LLM).

Glossary

9.1 Key Biological Terms

crispr CRISPRClustered Regularly Interspaced Short Palindromic Repeat, a bacterial genomic region used in pathogen defense.

CRISPRa CRISPR Activation; using a dCas9 or dCas9-activator with a gRNA to increase transcription of a target gene.

CRISPRi CRISPR Interference; using a dCas9 or dCas9-repressor with a gRNA to repress or decrease transcription of a target gene.

Cas CRISPR Associated Protein, includes nucleases like Cas9 and Cas12a (also known as Cpf1).

sgRNA Guide RNA (gRNA), a synthetic fusion of the endogenous bacterial crRNA and tracrRNA that provides both targeting specificity and scaffolding/binding ability for Cas9 nuclease. This synthetic fusion does not exist in nature so is commonly referred to as an sgRNA.

9.2 Key Machine Learning Terms

LLM Large Language Models are very large deep learning models that are pre-trained on vast amounts of data.

API Application Programming Interfaces are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols.

Declaration of generative AI in scientific writing

During the preparation of this work the main author used "ChatGPT" in order to improve the readability and language of certain paragraphs of the manuscript.

After using this tool/service, the main author reviewed and edited the content as needed and takes full responsibility for the content of this project.

Bibliography

David Adams. Frameshift mutation. https://www.genome.gov/genetics-glossary/Frameshift-Mutation, 2025. Accessed May 13, 2025.

Addgene. Crispr guide. https://www.addgene.org/guides/crispr/crispr-basics, 2025. Accessed May 13, 2025.

Addgene. Crispr basics. Addgene CRISPR Guide, n.d. Accessed on 10/03/2025.

Fatwa Adikusuma, Caleb Lushington, Jayshen Arudkumar, Gelshan I Godahewa, Yu CJ Chey, Luke Gierus, Sandra Piltz, Ashleigh Geiger, Yatish Jain, Daniel Reti, et al. Optimized nickase-and nuclease-based prime editing in human and mouse cells. *Nucleic acids research*, 49(18):10785–10795, 2021.

Open AI. Evals design best practices. https://platform.openai.com/docs/guides/evals-designmulti-agent-architectures, 2025. Accessed May 28, 2025.

Imad Ajjawi, John Verruto, Moena Aqui, Leah B Soriaga, Jennifer Coppersmith, Kathleen Kwok, Luke Peach, Elizabeth Orchard, Ryan Kalb, Weidong Xu, et al. Lipid production in nannochloropsis gaditana is doubled by decreasing expression of a single transcriptional regulator. *Nature biotechnology*, 35(7):647–652, 2017.

Felicity Allen, Luca Crepaldi, Clara Alsinet, Alexander J. Strong, Vitalii Kleshchevnikov, Pietro De Angeli, Petra Páleníková, Anton Khodak, Vladimir Kiselev, Michael Kosicki, Andrew R. Bassett, Heather Harding, Yaron Galanty, Francisco Muñoz-Martínez, Emmanouil Metzakopian, Stephen P. Jackson, and Leopold Parts. Predicting the mutations generated by repair of cas9-induced double-strand breaks. *Nature Biotechnology*, 37(1):64–72, 11 2018. ISSN 1087-0156. doi: 10.1038/nbt.4317. URL `http://dx.doi.org/10.1038/nbt.4317`.

Andrew V Anzalone, Peyton B Randolph, Jessie R Davis, Alexander A Sousa, Luke W Koblan, Jonathan M Levy, Peter J Chen, Christopher Wilson, Gregory A Newby, Aditya Raguram, et al. Search-and-replace genome editing without double-strand breaks or donor dna. *Nature*, 576(7785):149–157, 2019.

Arize AI, Inc. Agent Evaluation. `https://arize.com`, 2025. Accessed: 2025-06-02.

National Cancer Institute at the National Institutes of Health. Frameshift mutation. https://www.cancer.gov/publications/dictionaries/genetics-dictionary/def/frameshift-mutation, 2025. Accessed May 13, 2025.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Rodolphe Barrangou, Christophe Fremaux, Hélène Deveau, Melissa Richards, Patrick Boyaval, Sylvain Moineau, Dennis A Romero, and Philippe Horvath. Crispr provides acquired resistance against viruses in prokaryotes. *Science*, 315(5819):1709–1712, 2007.

Gonzalo Benegas, Chengzhong Ye, Carlos Albors, Jianan Canal Li, and Yun S Song. Genomic language models: opportunities and challenges. *Trends in Genetics*, 2025.

Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Debiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

Kai Blin, Lasse Ebdrup Pedersen, Tilmann Weber, and Sang Yup Lee. Crispy-web: an online resource to design sgrnas for crispr applications. *Synthetic and Systems Biotechnology*, 1(2): 118–121, 2016.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Ben Burtenshaw, Joffrey Thomas, and Thomas Simonini. The hugging face agents course. https://github.com/huggingface/agents-course, 2025. GitHub repository.

Rina Diane Caballar. What are foundation models? https://www.ibm.com/think/topics/foundation-models, 2024. Accessed April 15, 2025.

K.R. Chowdhary. Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649, 2020.

Yanyi Chu, Dan Yu, Yupeng Li, Kaixuan Huang, Yue Shen, Le Cong, Jason Zhang, and Mengdi Wang. A 5 utr language model for decoding untranslated regions of mrna and function predictions. *Nature Machine Intelligence*, 6(4):449–460, 2024.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Bryan Clark. What is langgraph? https://www.ibm.com/think/topics/langgraph, 2024. Accessed April 15, 2025.

Veniamin Fishman, Yuri Kuratov, Aleksei Shmelev, Maxim Petrov, Dmitry Penzar, Denis Shepelin, Nikolay Chekanov, Olga Kardymon, and Mikhail Burtsev. Gena-lm: a family of open-source foundational dna language models for long sequences. *Nucleic Acids Research*, 53(2):gkae1310, 2025.

Galileo AI. Mastering ai agents, 2024. Available here. Accessed on 3/03/2025.

Nicole M Gaudelli, Alexis C Komor, Holly A Rees, Michael S Packer, Ahmed H Badran, David I Bryson, and David R Liu. Programmable base editing of a - t to g- c in genomic dna without dna cleavage. *Nature*, 551(7681):464–471, 2017.

Gradio. Getting Started. `https://www.gradio.app/guides/quickstart`, 2025. Accessed: 2025-06-02.

Hope Henderson. Crispr clinical trials: A 2024 update, 2024. Available here. Accessed on 7/03/2025.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

Alex Hodgkins, Anna Farne, Sajith Perera, Tiago Grego, David J Parry-Smith, William C Skarnes, and Vivek Iyer. Wge: a crispr database for genome engineering. *Bioinformatics*, 31 (18):3078–3080, 2015.

Kaixuan Huang, Yuanhao Qu, Henry Cousins, William A Johnson, Di Yin, Mihir Shah, Denny Zhou, Russ Altman, Mengdi Wang, and Le Cong. Crispr-gpt: An llm agent for automated design of gene-editing experiments. *arXiv preprint arXiv:2404.18021*, 2024.

Huggingface. The hugging face course, 2022. https://huggingface.co/course, 2022. [Online; accessed 2023/04/25].

Inc. LangChain. LangSmith: Evaluation Framework for AI Applications. `https://www.langchain.com/langsmith`, 2023. Accessed: 2025-06-02.

Yoshizumi Ishino, Hideo Shinagawa, Kozo Makino, Mitsuko Amemura, and Atsuo Nakata. Nucleotide sequence of the iap gene, responsible for alkaline phosphatase isozyme conversion in escherichia coli, and identification of the gene product. *Journal of bacteriology*, 169(12): 5429–5433, 1987.

Mike Ivancie. The best ai chatbots & llms of q1 2025: Complete comparison guide, 2025. Available here. Accessed on 3/03/2025.

Martin Jinek, Krzysztof Chylinski, Ines Fonfara, Michael Hauer, Jennifer A Doudna, and Emmanuelle Charpentier. A programmable dual-rna–guided dna endonuclease in adaptive bacterial immunity. *science*, 337(6096):816–821, 2012.

Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: state of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744, 2023.

Jonas Koeppel, Juliane Weller, Elin Madli Peets, Ananth Pallaseni, Ivan Kuzmin, Uku Raudvere, Hedi Peterson, Fabio Giuseppe Liberante, and Leopold Parts. Prediction of prime editing insertion efficiencies using sequence features and dna repair determinants. *Nature Biotechnology*, 41(10):1446–1456, 2023.

Langfuse. Langfuse: Observability for AI Applications. `https://www.langfuse.com`, 2023. Accessed: 2025-06-02.

LangGraph. Langgraph glossary. https://langchain-ai.github.io/langgraph/concepts/low_level/, 2024. Access

Benjamin Larsen, Cathy Li, Stephanie Teeuwen, Olivier Denti, Jason DePerro, and Efi Raili. Navigating the ai frontier: A primer on the evolution and impact of ai agents. Technical report, World Economic Forum, 2024.

David Liwei. Github - davidliwei/awesome-crispr: List of software/websites/databases/other stuff for genome engineering. https://github.com/davidliwei/awesome-CRISPR, 2020. Accessed April 9, 2025.

John McCarthy. Programs with common sense, 1959.

Eric Nguyen, Michael Poli, Matthew G Durrant, Brian Kang, Dhruva Katrekar, David B Li, Liam J Bartie, Armin W Thomas, Samuel H King, Garyk Brixi, et al. Sequence modeling and design from molecular to genome scale with evo. *Science*, 386(6723):eado9336, 2024.

Aidan R o'Brien, Laurence OW Wilson, Gaetan Burgio, and Denis C Bauer. Unlocking hdr-mediated nucleotide editing by identifying high-efficiency target sites using machine learning. *Scientific reports*, 9(1):2788, 2019.

Ananth Pallaseni, Elin Madli Peets, Jonas Koeppel, Juliane Weller, Thomas Vanderstichele, Uyen Linh Ho, Luca Crepaldi, Jolanda van Leeuwen, Felicity Allen, and Leopold Parts. Predicting base editing outcomes using position-specific sequence determinants. *Nucleic Acids Research*, 50(6):3551–3564, 2022.

Ananth Pallaseni, Elin Madli Peets, Gareth Girling, Luca Crepaldi, Ivan Kuzmin, Marilin Moor, Núria Muñoz-Subirana, Joost Schimmel, Özdemirhan Serçin, Balca R. Mardin, Marcel Tijsterman, Hedi Peterson, Michael Kosicki, and Leopold Parts. The interplay of dna repair context with target sequence predictably biases cas9-generated mutations. *Nature Communications*, 15(1), November 2024. ISSN 2041-1723. doi: 10.1038/s41467-024-54566-7. URL `http://dx.doi.org/10.1038/s41467-024-54566-7`.

Lei S Qi, Matthew H Larson, Luke A Gilbert, Jennifer A Doudna, Jonathan S Weissman, Adam P Arkin, and Wendell A Lim. Repurposing crispr as an rna-guided platform for sequence-specific control of gene expression. *Cell*, 152(5):1173–1183, 2013.

Benedikt Rauscher, Florian Heigwer, Marco Breinig, Jan Winter, and Michael Boutros. Genomecrispr-a database for high-throughput crispr/cas9 screens. *Nucleic acids research*, page gkw997, 2016.

Relari. Choosing the right ai agent framework: Langgraph vs crewai vs openai swarm, 2024. Available here. Accessed on 3/03/2025.

Guillaume Richard, Bernardo P de Almeida, Hugo Dalla-Torre, Christopher Blum, Lorenz Hexemer, Priyanka Pandey, Stefan Laurent, Marie Lopez, Alexandre Laterre, Maren Lang, et al. Chatnt: A multimodal conversational agent for dna, rna and protein tasks. *bioRxiv*, pages 2024–04, 2024.

Rebecca Roberts. Crispr off-target editing: Prediction, analysis, and more. https://www.synthego.com/blog/crispr-off-target-editing, 2025. Accessed May 13, 2025.

Wei Ruan, Yanjun Lyu, Jing Zhang, Jiazhang Cai, Peng Shu, Yang Ge, Yao Lu, Shang Gao, Yue Wang, Peilong Wang, et al. Large language models for bioinformatics. *arXiv preprint arXiv:2501.06271*, 2025.

Jakob Russel, Rafael Pinilla-Redondo, David Mayo-Muñoz, Shiraz A Shah, and Søren J Sørensen. Crisprcastyper: automated identification, annotation, and classification of crispr-cas loci. *The CRISPR journal*, 3(6):462–469, 2020.

Bin Shao and Jiawei Yan. A long-context language model for deciphering and generating bacteriophage genomes. *Nature Communications*, 15(1):9392, 2024.

Cormac Sheridan. The world's first crispr therapy is approved: who will receive it. *Nat Biotechnol*, 42(1):3–4, 2024.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529 (7587):484–489, 2016.

Hana J. Johannesen Netanya Y Spencer. What is crispr screening? https://eu.idtdna.com/pages/education/decoded/article/overview-what-is-crispr-screening, 2025. Accessed May 13, 2025.

Cole Stryker. What is ai agent evaluation? https://www.ibm.com/think/topics/ai-agent-evaluation: :text=Evaluation Accessed May 12, 2025.

Symflower. Evaluating llms: complex scorers and evaluation frameworks. https://symflower.com/en/company/blog/2024/llm-complex-scorers-evaluation-frameworks/, 2024. Accessed May 31, 2025.

Synthego. Importance of the pam sequence in crispr experiments. https://www.synthego.com/guide/how-to-use-crispr/pam-sequence, 2025a. Accessed May 13, 2025.

Synthego. The complete guide to understanding crispr sgrna. https://www.synthego.com/guide/how-to-use-crispr/sgrna, note = Accessed May 13, 2025, 2025b.

David Tilman, Christian Balzer, Jason Hill, and Belinda L Befort. Global food demand and the sustainable intensification of agriculture. *Proceedings of the national academy of sciences*, 108(50):20260–20264, 2011.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Zhiheng Xi, Wenxiang Chen, Xin Guo, Wei He, Yiwen Ding, Boyang Hong, Ming Zhang, Junzhe Wang, Senjie Jin, Enyu Zhou, et al. The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2):121101, 2025.

Andy Hsien-Wei Yeh, Christoffer Norn, Yakov Kipnis, Doug Tischer, Samuel J Pellock, Declan Evans, Pengchen Ma, Gyu Rie Lee, Jason Z Zhang, Ivan Anishchenko, et al. De novo design of luciferases using deep learning. *Nature*, 614(7949):774–780, 2023.

Asaf Yehudai, Lilach Eden, Alan Li, Guy Uziel, Yilun Zhao, Roy Bar-Haim, Arman Cohan, and Michal Shmueli-Scheuer. Survey on evaluation of llm-based agents. *arXiv preprint arXiv:2503.16416*, 2025.

Daoan Zhang, Weitong Zhang, Yu Zhao, Jianguo Zhang, Bing He, Chenchen Qin, and Jianhua Yao. Dnagpt: a generalized pre-trained tool for versatile dna sequence analysis tasks. *arXiv preprint arXiv:2307.05628*, 2023.

Pengyu Zhao, Zijian Jin, and Ning Cheng. An in-depth survey of large language model-based artificial intelligence agents. *arXiv preprint arXiv:2309.14365*, 2023.

Yizhen Zheng, Huan Yee Koh, Maddie Yang, Li Li, Lauren T May, Geoffrey I Webb, Shirui Pan, and George Church. Large language models in drug discovery and development: From disease mechanisms to clinical trials. *arXiv preprint arXiv:2409.04481*, 2024.

Jieli Zhou. Awesome ai agents for scientific discovery, 2024. Available here. Accessed on 10/03/2025.

Haocheng Zhu, Chao Li, and Caixia Gao. Applications of crispr–cas in agriculture and plant biotechnology. *Nature Reviews Molecular Cell Biology*, 21(11):661–677, 2020.

Xiao Zhu, Chenchen Qin, Fang Wang, Fan Yang, Bing He, Yu Zhao, and Jianhua Yao. Cd-gpt: a biological foundation model bridging the gap between molecular sequences through central dogma. *bioRxiv*, pages 2024–06, 2024.