Animal Information Processing and Retrieval System

João Alves
FEUP
Porto, Portugal
up202007614@up.pt

Marco André
FEUP
Porto, Portugal
up202004891@up.pt

Mário Ferreira
FEUP
Porto, Portugal
up201907727@up.pt

Ricardo Matos
FEUP
Porto, Portugal
up202007962@up.pt

ABSTRACT

A-Z-Animals.com is an award-winning online encyclopedic resource with the goal of educating people about the natural world, offering a wide range of verified curiosities about animals.

This project aims to process the data from this website to create an information retrieval system about animals and being this paper an outlining of the comprehensive process of developing an information retrieval system for animal data using *Apache Solr*. The project initiated with the acquisition and analysis of a substantial animal dataset, involving data reduction techniques and transformations to enhance usability.

A detailed conceptual model was constructed, illustrating the entities within the dataset and their relationships. Subsequently, the focus shifted to the implementation of the information retrieval system using *Apache Solr*, involving data indexing, schema design, and query parameterization.

The system's effectiveness was rigorously evaluated through five distinct search scenarios, employing various configurations, including two schemas and two query parsers (*Apache Lucene* and *EDisMax*). Results and findings from this evaluation phase contribute to a deeper understanding of the system's performance, paving the way for future enhancements. Metrics such as precision, recall, F-Measure, and Mean Average Precision were collected and analyzed for the different configurations.

Results indicate the system's capability to retrieve relevant information across various scenarios and detailed analysis and findings for each search scenario, along with graphical representations, contribute to a thorough understanding of the system's performance.

A set of global parameters with an ideal setup were derived from the results and then many approaches were taken to improve upon the search system, with detailed discussions of the explored strategies and metric results to provide a base of statistical understanding.

The paper concludes with a discussion of the achieved outcomes, emphasizing the combination of a well-curated dataset and an optimized search system to deliver valuable insights into animal information.

KEYWORDS

dataset, information retrieval system, data retrieval, data preparation, data processing, data characterization, statistical analysis, search engine, Solr, EDisMax, query parser, Apache Lucene, indexing, tokenization, query, search system, learning to rank, semantic search

1 INTRODUCTION

The *A-Z-Animals.com* [1] organization is composed of an experienced editorial team responsible for researching over more than a 100 encyclopedias and other definitive guides about the animal kingdom. The core values of the organization revolve around delivering precise, current information to foster free knowledge dissemination and raise awareness about the issue of extinction while also supporting animal conservation efforts.

The organization maintains a comprehensive scientific database containing information on a lot of known animal species, as well as intriguing scientific facts and discoveries related to these creatures. This information encompasses scientific classification, geographical distribution, estimated population size, and much more specie specific curiosities.

The organization's commitment to providing precise and current information about the animal kingdom, inspired the development of the developed Animal Information Processing and Retrieval System, contributing to free knowledge dissemination.

In this paper, we outline the key objectives of our project, detailing the methodologies employed and presenting noteworthy findings.

Our specific goals include enhancing the accessibility and searchability of animal information, providing a more specialized search experience, and contributing to the understanding of the animal kingdom. Throughout the development process, we utilized advanced data processing techniques to transform raw data from *A-Z-Animals.com* into a structured and informative dataset that forms the foundation for our information retrieval system.

In the subsequent sections, we delve into the details of the processes involved in the creation of said Information Retrieval System. By achieving these specific goals, our system seeks to make a valuable contribution to the field of animal information processing.

2 INSIGHTS INTO ANIMAL CLASSIFICATION

It is crucial to have a comprehensive understanding of the problem domain being addressed in order to conduct accurate and meaningful data analysis as this knowledge forms the bedrock upon which informed data-driven decisions can be made.

In that spirit, an essential aspect to grasp in this context is the animal kingdom classification system, from which one can understand how all living organisms are organized and related. Initially proposed and developed by *Carl Linnaeus*, this system creates a hierarchy of groupings called taxa, as well as binomial nomenclature that gives each animal species a two-word scientific name. The more features a group of animals have in common, the more alike they are. This classification process involves considering attributes such as Kingdom, Phylum, Class, Order, Family, Genus, and Species.

3 DATA PIPELINE

Following this initial contextual introduction, our team developed a Data Pipeline (see figure 1), encapsulating and describing the major steps to transform the source data into the final treated dataset to be used on the next project phase. In the following sections an in-depth view of all steps will be provided.

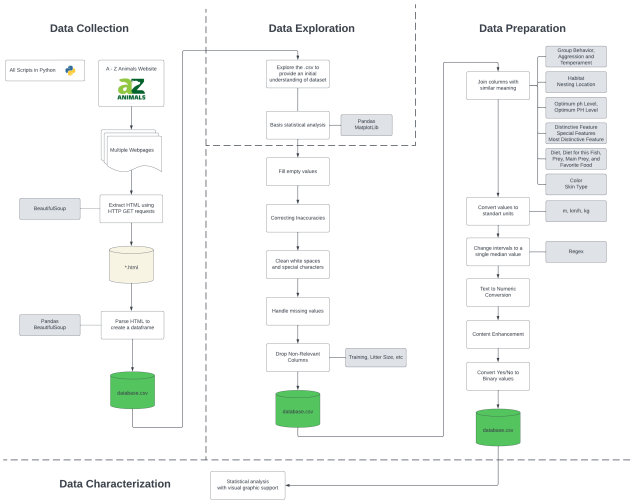

Figure 1: Data Pipeline for processing data into dataset

4 DATA COLLECTION

In the project's inception, the primary objective was to gather data from *A-Z-Animals.com*, a website that lacked an available API for data retrieval. Therefore, the data collection needed to be done via *web scraping* , which was divided into two distinctive steps: **data scrapping** and **data parsing**.

To accomplish both mentioned steps, Python scripts [5] were employed utilizing the 'requests' library [2] for making HTTP GET requests, mimicking a web browser, and 'Beautiful Soup' library for parsing the downloaded HTML content. This combination of these tools allowed us to efficiently extract comprehensive information about the animals present on the data source.

4.1 Data Scrapping

In the initial phase of our project, HTML collection was conducted through GET requests, utilizing the 'Beautiful Soup' library [9] to filter out irrelevant tags. These excluded tags, such as *<script>*, *<style>*, *<meta>*, *<img>*, and others tags were deemed irrelevant and thus were removed to avoid unnecessarily increasing the size and clutter of the downloaded HTML pages. As a result, cleaner HTML files containing the primary content of interest were obtained for all animals present on the source website.

It's relevant to disclosure that while the website has clear stipulations regarding licensing content for other websites (such as quoting), there is no mention (and thus no prohibition) on scrapping for personal/educational uses.

Notwithstanding, it's worth noting that our team was mindful of the web scraping activities being conducted and implemented deliberate pauses, achieved through the 'sleep()' function, between consecutive requests. This considerate approach was employed to prevent overwhelming the servers and to ensure a respectful and efficient data collection process.

4.2 Data Parsing

Following the HTML collection, embarking on a parsing process, extensive use of the 'Beautiful Soup' library was made. This step involved navigating and extracting pertinent data from the HTML content, effectively organizing information found within various tags. For instance, all <p> tags of specific classes were aggregated into a single 'text' variable, consolidating all descriptive written information about the animals into a single feature. Additionally, there were numerous tables in the web pages with well-structured data that was easily parsed and organized. All these endeavours culminated into a *DataFrame* created with aid of the 'pandas' package [4], containing all animals and their respective features.

The structured data adheres to a concrete format, with the *DataFrame* featuring the subsequent columns:

- 'Name': The animal's name;
- Taxonomic Information (divided into 7 columns regarding scientific classification);
- 'Quote': A noteworthy characteristic pertaining to the animal in the form of a short phrase;
- 'Text': Descriptive aggregated long text information;
- Facts columns (see below)

A large amount of supplementary animal facts columns were also derived from data extracted from 'Facts Box' tables of the web page but have been intentionally omitted in the above list for the sake of brevity. Further details regarding these attributes we'll be discussed later.

With this, the data collection phase was successfully concluded, resulting in a cohesive CSV file, serving as our starting point for data preparation and processing in the subsequent phases of the project.

5 DATA EXPLORATION

An initial data exploration is crucial to guide us into making informed and accurate analysis on the data, revealing hidden insights and data anomalies that should be answered via data cleaning and transformation.

At first inspection, the dataset is constituted of 2778 entries with 61 attributes, thus revealing a high dimensionality with multiple attributes with more than 50% missing data (a problem known as 'curse of dimensionality' that can be clearly seen in figure 2). This phenomenon can be attributed to the distinct traits of various animals groups that typically do not overlap and therefore leaves all other excluded animals with null values. For instance, this may include data such as 'ideal water's pH level' for fish species or even 'wingspan' for birds.

Moreover, due to human imprecision, some features appear duplicated in concepts but with subtle name differences, aggravating the high dimensionality and the missing values count. An example of this phenomenon are the pairs 'Diet' and 'Diet for this Fish' or even 'Color' and 'Skin Color'.

Finally, there were numerous instances where data consistency regarding measurement systems (e.g. km/mph, in/cm), and even representation formats varied. This inconsistency extended to the inclusion of non-numerical content (e.g. 'hundreds', '10M', '3-5 months') making a more detailed statistical analysis challenging to achieve before processing.

The initial data exploration granted us the knowledge needed to resolve the problems present in the dataset during the subsequent Data Preparation phase, described in the section that follows.

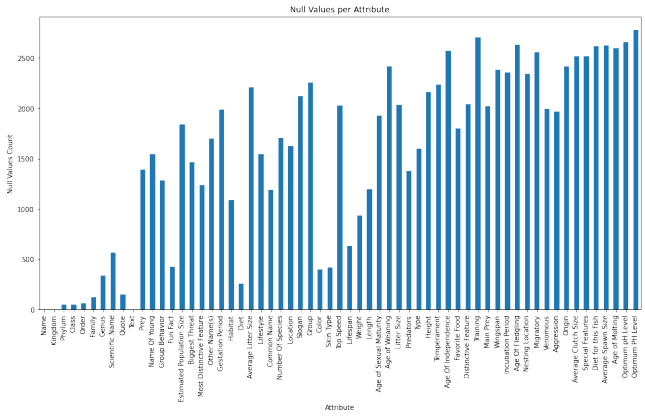

Figure 2: Quantity of null values per attribute

6 DATA PREPARATION

Despite the overall quality and accuracy of the source data, like stated in the Data Exploration section 5, numerous inconsistencies were still detected, which posed challenges at arriving at a clean dataset.

Besides the problems found during the exploration step, some other errors in the data were found for specific entries that could be explained by both some human error on the source and by irregular HTML page format that deviated from the norm and hence made the built parser fail for some instances.

With all that being said, major cleaning and transformation operations should take place in order to get a satisfiable dataset to work with. So, regarding data processing, two different phases were created and that will be discussed in greater detail below: **Data Cleaning** and **Data Transformation**.

6.1 Data Cleaning

Due to various reasons, the parsed data was far from clean, so, our team developed a reproducible cleaning script that was solely responsible for cleaning operations of various kinds that are summarized below:

- **Fill empty values**: Certain values, like for the 'Name of Young' feature were empty to well known cases (such as dogs and cats). Specifically for those cases, the empty cells were filled with the appropriate values.
- **Correcting inaccuracies**: Multiple parsing errors due to pages layouts variance that resulted in nonsensical results in certain attributes (like years measured in meters) were corrected.
- **Removing white spaces and other special characters**: In order to address and eliminate unwanted characters like '\t', '\n', extra spaces or quotes around text entries, regular expression substitution was applied.
- **Handling Missing Values**: In many cases, missing values were replaced with empty strings and, in instances where value omission was in itself an indicator (for instance, in the 'Venomous' column, the null values were replaced by 'No').
- **Dropping irrelevant columns**: Features that were deemed irrelevant or which sparsity wouldn't justify their existence were dropped to reduce noise.

6.2 Data Processing

Data processing is, together with data cleaning, an essential step to ensure data quality, consistency, and readiness for analysis. This section outlines the various techniques and procedures applied to process the data, resulting in a more structured, standardized, and informative dataset, laying the foundation for meaningful analysis during the data characterization phase that follows.

6.2.1 Feature Engineering and Data Reduction. Like previously stated, the dataset contained a large number of attributes that, after closer inspection during the data exploration phase, revealed themselves to have overlapping information or information that, nonetheless, could be simplified into less, more informative features by merging columns in the dataframe. This allows for a great reduction in dimensionality (which in turn boosts data organization and future tasks' performance) while still preserving the relevant information. Below listed are the changes made in this regard:

- The 'Aggression,' 'Group Behavior,' and 'Temperament' attributes were parsed and merged into a single 'Behavior' column, providing a comprehensive description of an animal's overall behavior.
- The 'Color' and 'Skin Type' values were merged into a 'Body' attribute, which describes the animal's appearance and coloration.
- The 'Habitat' and 'Nesting Location' features were combined into a 'Natural Habitat' attribute to provide information about where the animal is typically found.
- Food-related features, such as 'Diet,' 'Diet for this Fish,' 'Prey,' 'Favorite Food,' and 'Main Prey,' were processed and merged into a single 'Diet' attribute to describe the animal's eating habits.

Feature merging required, for many cases, regular expressions for the construction of new phrases that encapsulated all the information neatly into a small text block. As it can be observed in figure 3, the actions taken mitigated the impact of missing data in certain attributes, improving the overall descriptive quality and, together with the data cleaning dropped attributes, reducing the feature count from 61 to 42.

6.2.2 Data Transformation. After applying data reduction techniques, there were still several attributes that required transformations to enhance the dataset usability and analytical capabilities. These transformations are of various kinds and, in a nutshell, include:

- **Unit Conversion and Standardization**: Standardized units for measurement attributes like 'weight', 'speed', 'lifespan', and 'height' by converting various units into SI Units. In addition, non-numeric data (like 'one million') that was

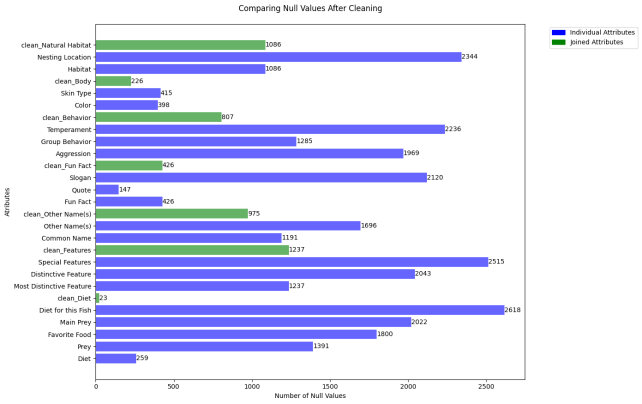

Figure 3: Comparing Number of Null Values After Feature Engineering

present in these attributes were replaced by numerical values, ensuring these features are consistent and comprised only of numeric data.

- **Interval Standardization**: Numeric intervals, particularly in the 'Lifespan' column, were standardized to represent midpoint values in order to facilitate statistical analysis.
- **Text to Numeric Conversion**: Non-numeric vague terms (such as 'a few') were replaced with concrete numerical values and instances with ambiguous meaning (e.g. 'thousands of specimens'), requiring regular expression substitution for integers that respected the order of magnitude in question.
- **Content Enhancement**: Some attributes such as 'Lifestyle' underwent a transformation where concise, descriptive sentences were generated from long descriptions to capture the essence of each animal's way of life.
- **Boolean Value Transformation**: Attributes containing boolean values, such as 'Venomous' and 'Migratory,' were transformed into binary format. 'Yes' was represented as 1, and 'No' as 0, enhancing compatibility.

7 DATA CHARACTERIZATION

With a cleaner dataset in hands, a more profound statistical analysis can be performed in order to better understand the data. From the abundant insights made, for the sake of brevity, only the two most relevant of them are presented in figures 4 and 5.

From the figures 4 and 5 one can infer that for most animals there is a very long text field (average of 1200 words with Q1 and Q3 still around the 1000 words normal distribution), being this large amount of text very useful for more accurate information retrieval.

Another very relevant thing to note about the data is that it very apparent bias towards dogs and wolves (Genus *Canis* has 525 entries because it includes all dog breeds), which represent almost 20% of all animal entries. This is a fact to be wary of as statistical results and search retrieved results can be skewed with so many individuals of a single genus.

Finally, while the source website for the data claims to include animals from all ranges, some specific classes of animals are very

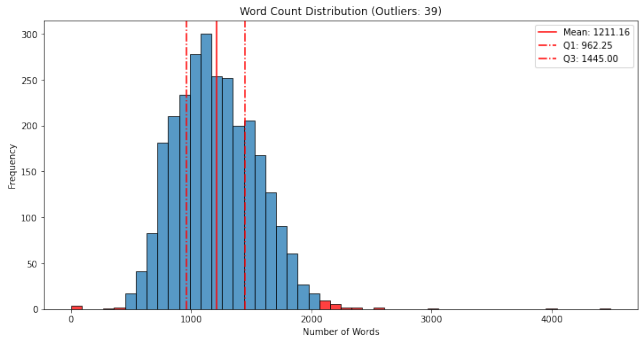

Figure 4: Text field word count

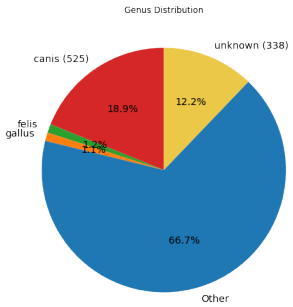

Figure 5: Genus Type Distribution

poorly represented (e.g. insects, spiders and fishes), which limits the search opportunities to fewer types of animals and their features.

8 CONCEPTUAL DATA MODEL

In figure 6, the conceptual model illustrates the entities within the dataset and that consists of animals, which are complemented by entities representing natural habitats and the taxonomic classification chain. The remaining features are intrinsic to each animal entry.

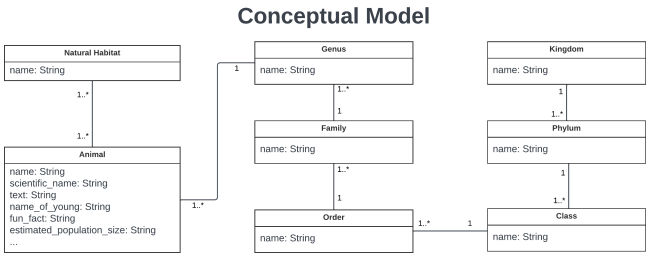

Figure 6: Conceptual model

9 PROSPECTIVE SEARCH TASKS

Some of the information needs that are expected to be met at the end of the project with the implementation of the information retrieval system are:

- Find energetic dog breeds suited for hunting;
- Find animals native to North America that like to eat insects;
- Find animals that change the color of their skin, fur or feathers for the purpose of camouflage;
- Animals that generally walk in groups with a respective hierarchy and how they deal with territory;
- Find animals besides birds that migrate to Mexico;

10 COLLECTION AND INDEXING

For information collection and indexing, our team used a information retrieval tool named *Apache Solr* [8], which is an open-source, highly scalable search platform built on top of *Apache Lucene* [3], offering powerful full-text search capabilities and flexible document-oriented indexing. In the *Solr* process, a user begins by defining a schema to index imported data, specifying indexing and query rules for the fields. Subsequently, the query system is employed to search for and retrieve documents based on the established schema and rules.

10.1 Document Definition

In the earlier data preparation stage, the cleaned and processed animal data was stored in a CSV file, making trivial the data importing into Solr, given its ability to accept CSVs for indexing collections. To facilitate the import process even further and make it reproducible, we mapped the data directory to a directory within a docker container.

The outcome is a well-organized collection of individual animal entries. Each entry serves as a distinct document within the information retrieval system, encapsulating comprehensive details about the respective animal.

10.2 Indexing Process

Solr indexing configuration are specified in a *JSON* schema file, which, for each feature present in the dataset, assigns a custom type field and a set of operations to be performed on indexing and querying times.

The indexing is re-done each time the startup script is re-run. For that, we make sure to re-create the previous utilized core (ultimately dropping every document and schema that was on it) and send the new schema and data, in this order, for the indexing to occur again as a clean slate.

10.3 Schema Details

In order to understand the impact of different combinations of tokenizers and filters analyzers on query results, the team implemented 2 different Solr schemas. While with the first setup is more simple (from here on out mentioned only as Schema 1), the second one (Schema 2) is more fine-tuned and has additional tokenizers and filters in order to produce more relevant and precise results on search.

The different schema's impact on the performance metrics will be discussed in deeper detail in the follow-up section 11.2.

For the schemas, four custom field types were created to suit distinct data categories present in the dataset, utilizing tokenizers and filters analyzers (done at both index and query time) to enhance the precision of the query system like presented below:

- **generalTextField:**
  - **Tokenizers:** StandardTokenizerFactory
  - **Filters:** ASCIIFoldingFilterFactory, LowerCaseFilterFactory, KStemFilterFactory, ManagedSynonymGraphFilterFactory, FlattenGraphFilterFactory.
- **numeric_general:**
  - **Tokenizers:** KeywordTokenizerFactory
  - **Filters:** PatternReplaceFilterFactory (removes non-numeric characters).
- **shortTextField:**
  - **Tokenizer:** StandardTokenizerFactory
  - **Filters:** LowerCaseFilterFactory, ASCIIFoldingFilterFactory, BeiderMorseFilterFactory, ManagedSynonymGraphFilterFactory, FlattenGraphFilterFactory.
- **commaSeparatedText:**
  - **Tokenizers:** PatternTokenizerFactory (pattern: ",")
  - **Filters:** TrimFilterFactory.

The fields in which these custom field types were applied can be seen in A.1.

It's important to note that the simplified schema 1 loses most of the above mentioned analyzers, and only uses StandardTokenizerFactory, ASCIIFoldingFilterFactory, LowerCaseFilterFactory and some other basic filters. The team's experiments found that these configurations grant satisfiying documents relevance when searching. Some of the analyzers worth special mentioning and that are only present on schema 2 and that enhance the results considerably (as will be shown later) are:

- **KStemFilterFactory:** Stemming algorithm for English, enhancing searchability by reducing words to their root form.
- **ManagedSynonymGraphFilterFactory:** Considers synonymous terms using a managed synonym list for English, broadening search scope.
- **BeiderMorseFilterFactory:** Considers phonetic variations of terms through the Beider-Morse phonetic matching algorithm.

10.4 Information Retrieval

With the data indexed in *Solr*, the system was configured for effective information retrieval. Several query parameters and setups were considered to optimize the search functionality. Namely, the team experimented with two different query parsers: *Apache Lucene* [3] and *EDisMax* [8].

10.4.1 Query Parameterization. The main goal of employing Solr's query system is to fulfill the information requirements outlined in subsection 10.4.2. To achieve that, query parameterization needs to take place in order to fine-tune and guide the searches being performed to retrieve relevant results.

In the scope of exploration of query parameterization, we delved into the strategic use of fields, terms, and independent boosts. We also embraced the intricacies of phrase matching with slops, wildcards, and proximity searches in order to improve query overall

results relevance. As the source of the data used is considered trustworthy and grammatically correct, there was no need for using fuzziness (word editing distance).

This approach extends to both *Lucene* and *EDisMax* query parsers, ensuring a fluid and comprehensive optimization of search outcomes.

10.4.2 Queries. Like stated on section 10.4.1, the queries employ many of the query parameterizations offered by Solr and the two query parsers being compared to achieve satisfying results for an end-user. The information needs earlier presented in section 9 were converted into the subsequent described queries:

(1) Find energetic dog breeds suited for hunting

```
Genus:Cani^3 AND
Text:("hunting dog"~5 AND "energetic")^5
Text:hunt*
```

(2) Animals native to North America that feed on insects

```
(Origin:"North America"^2.0 OR
Text:"native North America"~5) AND
(Diet:insects^2.0 OR
Text:"eats insects"~5)
```

(3) Animals that can change color, fur or feathers for the purpose of camouflage

```
Text:("change color"~5)^2 OR ("change skin"~5)
OR ("change fur"~5) OR (camouflage^2)
Fun_Fact: ("change color"~5)^2 OR ("change skin"~5)
OR ("change fur"~5) OR (camouflage^2)
Features: ("change color"~5)^2 OR ("change skin"~5)
OR ("change fur"~5) OR (camouflage^2)
```

(4) Animals that generally walk in groups with a respective hierarchy and how they deal with territory

```
(Behavior:Herd^1.8 OR Behavior:pack^1.9
 OR Text:group*^1.2)
(Text:hierarchy^2.2 OR
 Text:"social structure"~4^2.3)
(Text:territory*^1.6 OR
 Text: "deal with territory"~2^2.3)
```

(5) Animals besides birds that migrate to Mexico

```
(Class:Aves^0.01 OR (NOT Class:Aves)^15) AND
Migratory:true^1.5 AND
Text:((migra* AND "to Mexico"~5) OR ("to America"~10))^5
```

11 RESULTS EVALUATION AND DISCUSSION

To assess the effectiveness of the retrieval system, different evaluation setups were established, collecting relevant metrics, and comparing the results in order to better evaluate different setups (namely, comparing both the 2 query parsers and 2 schemas). The evaluation focused on collecting the following metrics: precision (more specifically P@10 because the first entries of a search tend to require more relevance if one wants to achieve a good retrieval system in most situations), recall, F-Measure (both also @10) and Mean Average Precision (MAP).

Recall measurements demand a comprehensive understanding of the entire relevant search results across all documents for each query. However, obtaining these relevant animals is very time consuming due to the dimensionality of the dataset. This involves knowing how relevant each document is to every query, posing a practical challenge. To make it more feasible, we only look at the first 30 returned animals and classify them by relevance to the corresponding query (being this 30 first results used to calculate the Average Precision).

11.1 Evaluation Setup

To automate as much as possible of the retrieval effectiveness quantification process, the team employed a *python* script that received 2 arguments: a text file with a name list of relevant animals for a given search scenario and the *Solr URL* for executing the query and retrieving the *JSON* results; allowing the script to fetch the query results and compare them against the manual defined good results in order to produce a precision-recall curve indicative of the obtained performance. This setup grants the team faster and easier comparisons between different setups and implementations.

A shell script was also employed to automate and store into CSVs the computation of Precision@N, Recall@N, and F-Measure for a given query. This facilitated the storage of search results and utilization of the generated statistics for creating visual graphics that helped us tune our configurations.

11.2 Discussion

The overall queries results were satisfactory and the metrics related to document relevancy are presented in the table 1. The five queries exhibited a consistently high average precision, especially for our optimal configuration (refined schema 2 using *EDisMax*) and which the presented values on the table refer to.

However, in specific instances like query 3, the precision at 10 was not as remarkable, being influenced by misleading text that skewed the system evaluation. It is worth noting, however, that in that particular case, the P@5 was 1.0, which still shows that the first 5 results were still relevant to an end-user.

Query search results are expected to have relativity high precision for the top retrieved documents in order to be considered a good and useful search system for an end-user and gradually fall for lower ranked, less relevant results. The area below a precision-recall curve is a very good indicator of overall information retrieval effectiveness and is another indicator taken into account when evaluating the query result relevancy.

Table 1: Retrieval effectiveness query metrics

Queries / Metrics	AvgP	P@10	R@10	F@10
Query 1	0.98	1.0	0.4	0.57
Query 2	0.94	0.90	0.32	0.47
Query 3	0.74	0.50	0.28	0.36
Query 4	0.83	0.90	0.38	0.53
Query 5	0.94	1.0	0.69	0.81

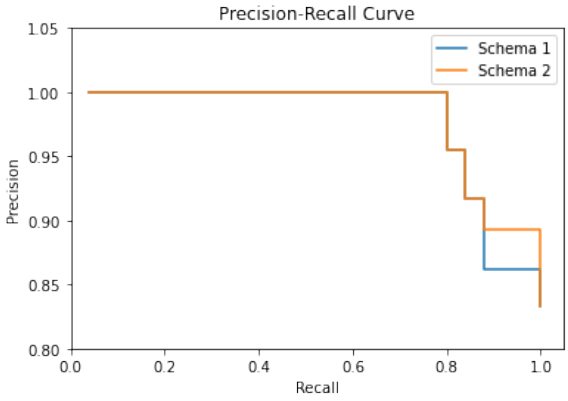

Figure 7: Schema Comparison for Search 1

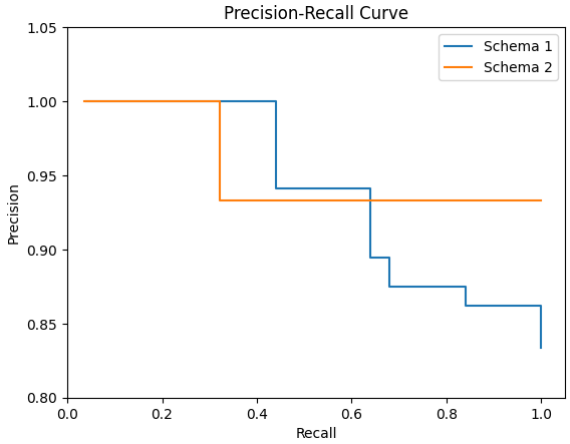

Figure 8: Schema Comparison for Search 2 using EDisMax

From the collected metrics, one can calculate the Mean Average Precision (MAP) for the queries which is 88.6%. This relatively high result is a comprehensive effectiveness evaluation for the developed information system and indicates that the system ranks and retrieves relevant items across various recall levels.

The following subsections cover in a more detailed manner the results and interesting findings relating to each performed search.

11.3 Search Scenario 1

The evaluation of the first search scenario was straightforward, as it involved determining whether a dog breed is suitable for hunting or not. Given that the dataset comprises over 500 identified dog breeds, the system can effortlessly provide many relevant results in the top 30 entries.

As we near the end of the retrieved results, the main text field shifts towards breeds with ancestral roots in energetic hunting breeds that have subsequently dolled those instincts and are no longer suited for hunting and were thus considered irrelevant.

As seen on figure 7, using the *EDisMax* query parser (which gives the same exact results for the default *Apache Lucene* parser), the first 20 results are relevant and schema 2 (the fine-tuned one) has slightly higher precision.

11.4 Search Scenario 2

The main challenge of the second information need was identifying which animals are native to North America. While some have their origin clearly defined in the appropriate field (*Origin*), only a very small subset of these are also animals that feed on insects. Hence, the search for the origin of an animal had to be expanded to consider multiple fields.

As the range of results was broadened, some irrelevant cases were introduced and that resulted mainly from mentions of related species or subspecies of the animal in question, which happened to originate from North America. An example of this is the case of the 'Kenyan Sand Boa', which has mentions to the 'Rubber boa' and the 'Rosy boa', native to North America, that belong to the same subfamily.

When comparing the results from the query for both schemas (figure 8), we can see that the initial query results for both schemas are relevant. However, the remaining ones differ slightly, with schema 2 providing a marginally higher precision overall despite the first schema providing some more relevant results before getting worse than schema 2.

11.5 Search Scenario 3

The third information need presented challenges during manual evaluation due to its broad definition as the query aimed to identify animals capable of changing their color, fur, or feathers for the concrete purpose of camouflage.

For that reason, there were some cases that introduced ambiguity and difficulty in relevancy classification, such as the 'Golden Tortoise Beetle' and 'Keyhole Cichlid'. According to their text description, they are able of changing color when they are disturbed or stressed, but it's not completely clear if they do so, with the purpose of camouflage. Hence, the team decided to include them, since a predator can be considered a cause for disturbance.

Moreover, some animals, like the 'Ribbon Eel', change their colors throughout the years, and others, such as the 'Giant Desert Centipede', change their color to warn predators that they are poisonous. Both of them were not considered, as they do not change their colors, with the specific purpose of camouflage.

When concerning schema and query parser comparison in this search scenario (presented in figure 9), it was possible to conclude that the version 2 of the schema had similar performances when using the *Lucene* query parser, and slightly better results when using *EDisMax*, when compared to the version 1 of the schema.

Despite the fact that the *EDisMax* query parser yielded one less relevant result compared to *Lucene* in the given search scenario, its overall performance surpassed it. The average precision for *EDisMax* was significantly higher, with a notable increase from 60% to 74%. This higher precision suggests that, despite having one less relevant result, *EDisMax* exhibited a superior ability to retrieve relevant information.

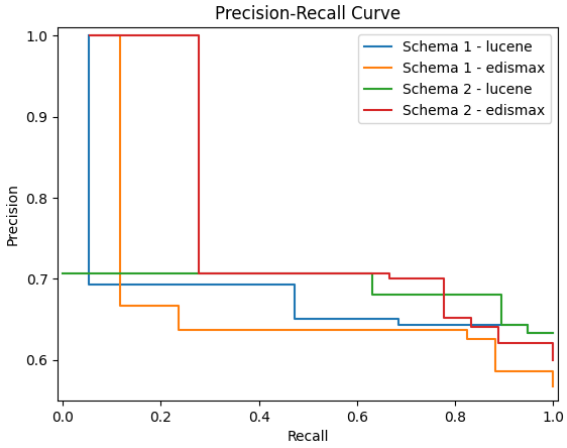

Figure 9: Precision-Recall Curve of Scenario 3

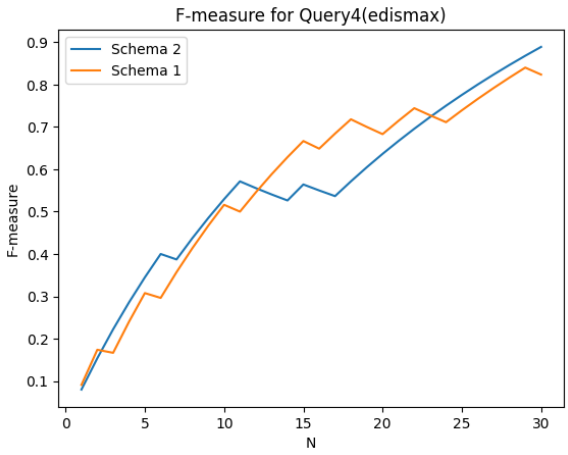

Figure 10: F-Measure for query 4

11.6 Search Scenario 4

The fourth information need proved challenging to manually evaluate because, for instance, animals like the 'African Forest Elephant' or the 'Sika Deer' are animals that, depending on the gender, have different social behaviours. Obviously, this variety of data made it difficult to classify the data as there is not a clear definition on either those animals are relevant or not.

There were also interesting things in the comparison between the 2 search engine versions as seen in figure 10. Although the refined version is better than the simple one, as expected, there is a small interval in which the simple version seems to yield more relevant animals.

11.7 Search Scenario 5

The fifth and final search scenario concerns about finding migration patterns of animals besides birds that have has destination Mexico or nearby regions. It was rather challenging to produce accurate results with this search due to the fact that the animals present on

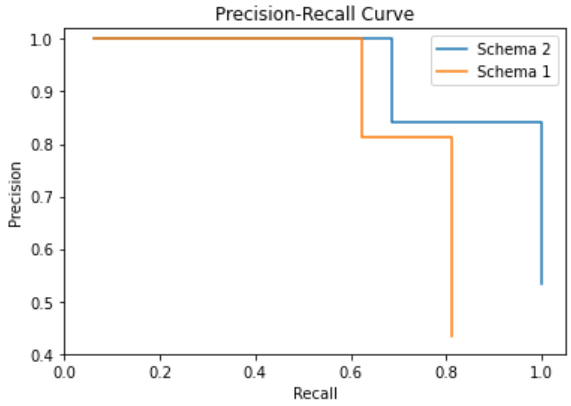

Figure 11: Schema Comparison for Search 5

the dataset that migrate are mostly birds and also because many animal's entries have vague information about migration destinations or give a large geographical area that might or not include Mexico (e.g. "migrates to Center and North America").

Despite the trial and error process of engineering a query that would give relevant entries, the results were still satisfactory. As seen in figure 11 using the *EDisMax* parser (as *Lucene* provides irrelevant documents), the results are relevant for up to 20 entries and also one can infer that the more fine-tuned schema provides higher precision.

12 GENERIC SEARCH WITH GLOBAL PARAMETERS

After making optimized queries for the specific search scenarios, the team derived a global parameterization that would allow for generic queries search in the dataset. For that, *EDisMax* was used with the parameters *qf*, *pf*, *mm* and *ps* to give different boosts to terms and allow for slops in text. The global parameters used are the following:

```
"params": {
    "q": "[Query Text]",
    "defType": "edismax",
    "qf": "Name^2.5 Features^2.0 Fun_Fact^2.0 Diet^2.0
    Text^1.5 Origin^3.5 Features^2.0 Behavior^2.0",
    "pf": "Name^2.5 Features^2.0 Fun_Fact^2.0 Diet^2.0
    Text^1.5 Origin^3.5 Features^2.0 Behavior^2.0",
    "mm": "3<-25%",
    "ps": 5,
    "fl": "Name, score",
    "rows": "30"
}
```

The queries' text were:

- Energetic dog breeds suited for hunting
- North America animals that like to eat insects
- Change the color of their skin, fur or feathers for the purpose of camouflage
- Animals that walk in hierarchical groups or herds and how they deal with territory

- (NOT Birds) migrate to Mexico or migrate to America

As expected, the metrics dropped significantly as the parameterization is no longer well-curated for each search need with a lower MAP of 49%. The metrics using the generic configuration for each query can be seen in table 2.

Table 2: Retrieval effectiveness metrics for global parameters

Queries / Metrics	AvgP	P@10	R@10	F@10
Query 1	0.64	0.70	0.41	0.51
Query 2	0.32	0.2	0.22	0.21
Query 3	0.14	0.10	0.25	0.22
Query 4	0.84	0.88	0.40	0.53
Query 5	0.51	0.4	0.66	0.50

13 EXPLORED IMPROVEMENTS

As the results achieved with the global parameter configuration presented in section 12 were relatively low, with a MAP of 49%, the search system is not providing users with relevant documents consistently. Some queries perform well, like it was the case with query 4 with average precision of 84%, but query 3 was the one that was penalized the most with only 14% average precision.

In order to improve the results achieved with the global parameter configuration, the team explored many different paths. Namely:

- Tailoring synonyms to the animal context
- "More Like This" suggestions in each animal page
- Semantic Search
- Learning to Rank
- GUI interface

The hope is that these changes will contribute to a better overall search system when incorporated together and provide users with more relevant documents about animals in order to achieve a good user experience.

14 GRAPHICAL USER INTERFACE

With the goal of providing a friendly user interface, the team developed a simple website where a user can make queries to the system, with snippets with highlights of query's keywords. To further enhance the search system a "more like this" functionality was developed, allowing users to check similar animals. These functionalities are shown in the figures in the annexes A.4.

15 METRICS RESULTS AND DISCUSSION

Taking into consideration all planned improvements mentioned in section 13, the following subsections will present the impact each individual improvement made in document relevance and discuss them.

15.1 Synonyms

In order to achieve better results, the team, besides using the default English synonyms analyzer on the *Solr* schema, developed a new context-tailored synonyms.txt file. To achieve it, a set of all words present on the dataset was created and then all words were iterated to find related words in a *word-net*, by using the *nltk* [10]

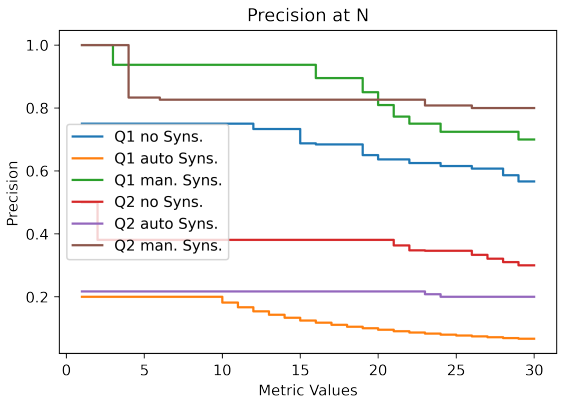

Figure 12: Synonyms Precision Performance

Python package (a package specifically used for natural language processing) and generate word associations.

This process, while not ideal, taking into consideration the very specific thematic being explored (animals) and the amount of data the team had to work with, was the best approach that was able to be achieved. After manual inspection, the results were actually pretty good all-around and captured many word relations as was intended (e.g. the lines: "*distress, distressed, disturbed, stressed, unhappy, upset, worried*" and "*battle, conflict, conflicts, difference, dispute, engagement, fight, infringe, struggle*"). Nonetheless, many instances were spotted where words, while still arbitrarily related, deviated more from their original concept than we would like (e.g. the line: "*dog, chase, chases, following, furrow, pursuit, tag, tail, track, trail*").

Unfortunately, despite our best efforts in the pursuit of enhancing query relevance via the incorporation of synonyms, very unexpected outcomes emerged as entry relevance shapely dropped for all queries as seen in figure 12. The main reason found for this disappointing metrics was the deviation from the original meaning for many words in the queries.

The first query ("Energetic dog breeds suited for hunting"), when equipped with synonyms, exhibited a reduction in relevance with only 2 out of 30 entries considered pertinent. This, paradoxically, contrasts with the baseline configuration without synonyms, which managed to achieve a higher relevance rate of 17 out of 30 entries. This unexpected results could be justified by the complexity introduced by synonyms, potentially expanding the search space in a manner that led to increased noise and decreased precision (e.g. the term "dog" could match with many different words associated with dogs like "tail" and, inevitably, other animals besides dogs ended up showing as results).

Similarly, the second query ("North America animals that like to eat insects"), after the introduction of synonyms, faced a decline in relevance, achieving a mere 6 out of 30 entries that met the criteria. In contrast, the version without synonyms surprisingly outperformed with 9 out of 30 entries considered relevant. This discrepancy again stems from the challenge of precisely defining and managing synonyms, as our overly broad synonyms introduce

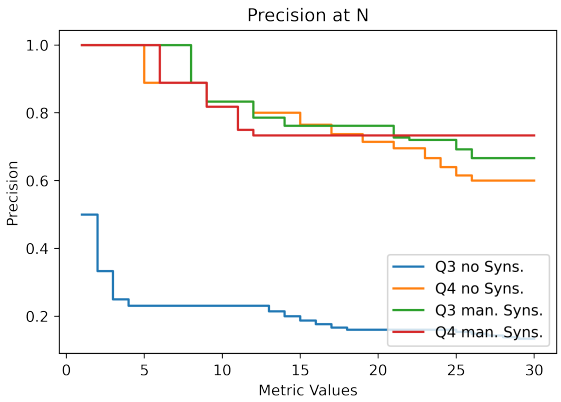

Figure 13: Query 3 and 4 Manual Synonyms

ambiguity and hinder the search engine's ability to accurately discern the user's intent. As such, the complexities introduced by the synonym configuration might have overshadowed the anticipated benefits, emphasizing the delicate balance required when implementing synonym-based strategies for query optimization in *Solr*.

A potential solution to the challenges encountered with the creation of a domain-specific synonym set configuration could involve a more sophisticated and manual approach to synonym generation. Rather than relying on a generic or automated synonym list we should have more fine control over introduced synonyms, taking into account the nuances of the animal-related queries while minimizing the risk of introducing irrelevant terms that worsen noise in result relevance. This described process was done for both queries as proof of concept and the results were notably higher as seen on figure 12, corroborating our theories. Refer also to to figure 13 for the queries 3 and 4 metrics where the impact of manually defining synonyms is again noticeable (specially for the third query).

As the dataset was very large, a more manual approach for all possible queries was deemed unachievable by the team and so, only synonyms for the 5 queries being studied were designed (and can be found in annexes A.3).

15.2 More Like This

The More Like This functionality (MLT) is designed to find documents in a *Solr* index that are similar to a given document. This can be useful to provide recommendations for a user based on a previously explored animal.

15.2.1 Configuration. The parameters of MLT were the following:

```
"params": {
    "mlt.fl": "Name,Text",
    "mlt.mintf": "10",
    "mlt.mindf": "6",
    "mlt.minwl": "3",
    "mlt.maxdfpct":"60",
    "mlt.boost": "true",
    "mlt.qf": "Name^2.0 Text^1.5"
}
```

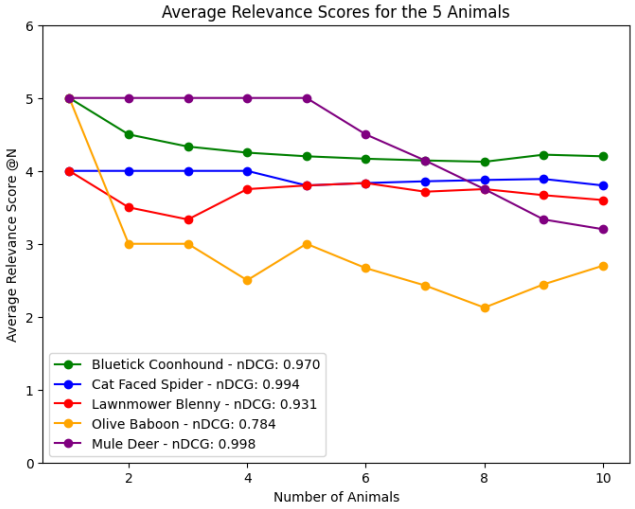

Figure 14: MLT Average @N

The parameters used a strict configuration for the MLT functionality. This was done because it was necessary to filter out some irrelevant results that were initially obtaining a high score due to the large amounts of text that each of these animals had.

15.2.2 Classification. In order to test our results the team decided to analyse the first animal retrieved for each of our previous queries. The animals chosen were: *Bluetick Coonhound*, *Cat Faced Spider*, *Lawnmower Blenny*, *Olive Baboon* and *Mule Deer*.

To classify our retrieved animals we evaluated them on the following scale:

- **0** - Nothing in common
- **1** - Different type of animal and almost no similar features
- **2** - Different type of animal and some similar features
- **3** - Same type of animal and almost no similar features
- **4** - Same type of animal and some similar features
- **5** - Same type of animal and almost all the same features

15.2.3 Results. The team analysed the top 10 results of each animal (total of 50) and classified each of the retrieved animals on the previous scale according to their similarity with the initial animal. Different aspects were taken into account such as appearance, habitat, evolution, size, reproduction, and other features that were mentioned in the Text field.

For each of the retrieved lists we calculated the normalized discount cumulative gain (nDCG). This allows us to us to measure the effectiveness of the retrieval system in capturing relevant information and facilitates the comparison between results.

In the figure 14 we can see that we obtained very satisfying results as 4 out of the 5 analysed animals achieved a nDCG above 0.9. Furthermore, the *Bluetick Coonhound* and the *Mule Deer* achieved a value very close to 1.0.

However, it's worth noting that, even though the *Mule Deer* obtained very high results in the first 5 animals, the score dropped considerably after that. This is due to the appearance of some animals that weren't similar to the *Mule Deer*, but were instead

their predators and, as such, were mentioned as prey in the text, and they also shared some common characteristics such as the habitat.

The *Olive Baboon* had the lowest score among the analysed animals for a similar reason. Many of the animals that were considered similar to it, were instead preys of baboons. In one special case, the related animal was a dog that was used as a protection means from the baboons.

Nevertheless, the results were still very good, and even the lowest ranking animal, the *Olive Baboon*, obtained a nDCG of 0.784. In conclusion, the effectiveness of the "More Like This" functionality was evident and delivered pertinent document suggestions.

15.3 Semantic Search

Semantic search comes as an alternative approach to querying the system that aims to improve the results obtained by understanding the meaning behind each query. To achieve this, the first step was generating fixed size dense vectors, comprising the semantic meaning for each document. The process of generating these vectors involved selecting a deep learning model and determining which fields of the documents to use, which in this case were the ones used in the global parameters (section 12).

The *sentence-transformers* python library [7], which provides an interface for several pre-trained models, was used in combination with the model *all-MiniLM-L6-v2*, that offers competitive quality when compared to larger models. The next step involved defining how the system performs the similarity-based searches, by creating a new field type in the schema to hold the dense vector values.

The similarity function employed was *cosine* along with the *k-nearest neighbours* algorithm *Hierarchical Navigable Small World*, HNSW, which offers a good compromise between the accuracy and search speed. The system is now able to take a dense vector, generated from the queries' text using the model presented before, and return the most similar documents.

Table 3: Retrieval effectiveness metrics for Semantic Search

Queries / Metrics	AvgP	P@10	R@10	F@10
Query 1	0.50	0.5	0.33	0.4
Query 2	0.42	0.5	0.5	0.5
Query 3	0.22	0.2	0.5	0.29
Query 4	0.17	0.1	0.17	0.13
Query 5	0.0	0.0	0.0	0.0

When comparing the results from the semantic search alternative (present in table 3) with the results from the generic search with global parameters (in table 2), there was a performance increase on only queries 2 and 3, with the remainder experiencing a decrease in the quality of the results. Notably, the last query did not yield any relevant results at all, as the documents retrieved were all instances of birds, making it seem that the set up for semantic search has difficulties dealing negations, such as NOT birds from query 5.

Overall, with a MAP of 26.2 % semantic search did not produce a significant improvement relative to the alternative using the query parser *eDisMax* with global parameters. A possible reason for this outcome may be the fact that the queries employed were tailored to optimize the results for the initial search system, and might not be suited for semantic search. An additional factor to take into consideration might be the possible limitations of the chosen model in encapsulating the semantic meaning of each document into a relatively small set of dimensions, specifically 384. Exploring a more complex model, capable of more effectively capturing the intent behind each document, may lead to an improvement of performance.

15.4 Learning to rank

Learning to rank is a machine learning technique that focuses on training a model to automatically rank items in a specific order of relevance. The underlying idea is to train a model with a set of examples where the correct ordering of items is set. *Solr*'s LTR plugin is a built-in plugin that is disable by default on *Solr* and so, to enable it, the team needed to update the core configuration file.

15.4.1 Features. In order to train the machine learning model, the team had to define a set of features and create the training set. A relevance to each document was manually attributed in order to access the relevance of the document but this approach would be expensive on a larger scale. Therefore, one could alternatively base the relevance score on analytical parameters (using logs from users). This score follows a scale from 0 to 5 that allows the distinction between a high relevant document and a not so relevant document. To each query a set of 30 animals were retrieved and scored by relevancy following the following criteria:

- **0 - Irrelevant Document:** The document is not pertinent to the query.
- **1 - Limited Relevance:** It partially matches the query in a few fields but remains considerably distant from the query itself.
- **2 - Partial Relevance:** The document partially aligns with the query in certain fields, approaching the intended meaning.
- **3 - Semantic Alignment:** Although it matches the query semantically, it may not fully grasp the query's nuanced meaning.
- **4 - Near-Perfect Match:** The document almost completely aligns with the query, lacking only one specific field or detail.
- **5 - Perfect Match:** The document perfectly corresponds to the query in all aspects.

The features used in LTR consisted mainly in the use of the fields matches that are presented in table 4. The original score presented in the table is the score that resulted from our global parameterization mentioned in section 12.

15.4.2 Model. We opt to use a linear model in *Solr LTR*. To train the model we used *Scikit Learn* [6] Python library basing on a **Pointwise approach** and using **a Linear Regression model**. The results presented in table 5 represent an improvement over the last global parameter configuration as only information need 5 got worst compared with our previous search system while achieving a mean average precision of 67,6%.

It's worth mentioning that team did not include any data about query 5 in the training set, hence, it was a bit expected that the results from that query would be worse than the others. In fact,

Features	Description	Weight
queryMatchName	Score of matching Name field	-0.41950315
queryMatchGenus	Score of matching Genus field	0.0
queryMatchClass	Score of matching Class field	0.0
queryMatchOrigin	Score of matching Origin field	-0.42167284
queryMatchFun Fact	Score of matching the Fun Fact	0.16945207
queryMatchMigratory	Score of matching the Migratory	0.4367062
queryMatchText	Score of matching the Text field	0.08357657
originalScore	The original score	0.08357656

Table 4: LTR model and features

what happens is that the model gives a high result for matches in the migratory field and when we search for query 5 a lot of irrelevant animals that migrate are shown as top search results, which may indicate that the model is being over-fitted to those 4 information needs. To make the search system more robust to different queries, the team would probably have to train over more features and information needs.

Table 5: Retrieval effectiveness metrics for Learning to Rank

Queries / Metrics	AvgP	P@10	R@10	F@10
Query 1	0.87	0.9	0.38	0.52
Query 2	0.95	1.0	0.4	0.57
Query 3	0.47	0.2	0.2	0.2
Query 4	0.82	0.7	0.41	0.52
Query 5	0.27	0.3	0.43	0.35

15.5 Final Remarks

Taking into account all the explored improvements mentioned in section 13, the team concludes that most of them contribute to a better overall search system. Some changes like the semantic search were less successful by achieving some higher results than the default global configuration in some queries but getting worse in others. Moreover, the synonyms turned out to be a great way to boost results relevance, however at the cost of manual synonym generation to be usable as the automatic version just added noise to searches and worsen the results.

When it comes to the "More Like This" *Solr* feature, the results were positive with high relevance in animal recommendations for most instances, thus emerging as a good feature to be present in the final search system GUI on individual animal pages without any prejudice.

Finally, learning to rank revealed itself to be a good way to improve result relevance for most queries without many setbacks. The only tricky part and that could still be tinkered could be the training of the model in order to yield even better results, like maybe experimenting with a Parwise approach. Despite all these taken paths, the search system still has room for growth as more advanced techniques could be explored.

16 CONCLUSION

The core objectives for the project's initial phase included searching for and selecting pertinent datasets, conducting exploratory data analysis, evaluating dataset quality, characterizing its fields, and identifying information requirements for the forthcoming system.

In light of those objectives, a substantial dataset was obtained, featuring data that displayed coherence and completeness to the best extent possible within the chosen subset of documents for analysis.

In the subsequent information retrieval phase, we delved into the information retrieval aspect of the project, leveraging *Solr* to create a powerful search system for animal information. Utilizing two different schemas, we designed and executed five search scenario queries, systematically collecting relevant metrics that helped the team make comparisons to ensure the effectiveness and efficiency of the system.

Finally, in the search system phase, the team focused on improving the search system's results as the global query configuration delimited in the end of the previous phase yielded not so good results as the metrics indicated with a MAP of 49% and some queries well below that (like query 3). To enhance the search system, multiple paths were taken and during this exploration, different degrees of success were achieved in order to improve the final search system. Nonetheless, the metrics show the system has improved from the previous phase but still with room for growth.

The outcomes of this information retrieval phase contribute to the goal of establishing a high-quality and relevant information retrieval system. The combination of a well organized dataset and the optimized search system assures a solid system in delivering valuable insights and information about animals.

REFERENCES

[1] az animals.com. https://a-z-animals.com/. [Online; accessed on October 8, 2023].
[2] Request lib. https://requests.readthedocs.io/en/latest/. [Online; accessed on October 8, 2023].
[3] Apache Lucene. https://lucene.apache.org/. [Online; accessed on November 15, 2023].
[4] Pandas. https://pandas.pydata.org/. [Online; accessed on October 8, 2023].
[5] Python. https://www.python.org/. [Online; accessed on October 8, 2023].
[6] scikit learn. Scikit Learn. https://scikit-learn.org/stable/. [Online; accessed on December 14, 2023].
[7] Sentence-Transformers. https://www.sbert.net/. [Online; accessed on December 13, 2023].
[8] Apache Solr. https://solr.apache.org/. [Online; accessed on November 15, 2023].
[9] Beautiful Soup. https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Online; accessed on October 8, 2023].
[10] Sphinx and NLTK Theme. Natural Language Toolkit. [Online; accessed on December 13, 2023].

Table 6: Query Relevance Condensed Results

Figure	Results
Fig 7	Schema 1: RRRRRRRRRRRRRRRRRRRRRNRNRNRRRNN Schema 2: RRRRRRRRRRRRRRRRRRRRRNRNRNNRNRN
Fig 8	Schema 1 eDisMax: RRRRRRRRRRRRNRRRRRNRNRRRRNRRRRN Schema 2 eDisMax: RRRRRRRRRRNRRRRNRRRRRRRRRRRRRR
Fig 9	Schema 1 Lucene: RNNNRRRRRNRRRNNRRNRRNNNRRRRRNR Schema 1 Edismax: RRNNRRNNRRNRRNNNRRRRRRNRNNNRRN Schema 2 Lucene: NRNRNNRRNRRRRRRRRNNNRRRRRNNRNR Schema 2 Edismax: RRRRRNNNNNRRRRRRRRNRRNNRNRNNRRN
Fig 10	Schema 1 Edismax: RRNRRNRRRRNRRRRNRRNNRRNNRRRRRN Schema 2 Edismax: RRRRRRNRRRRNNNRNNRRRRRRRRRRRRR
Fig 11	Schema 1: RRRRRRRRRRNNNRRRNNNNNNNNNNNNNN Schema 2: RRRRRRRRRRRNNNRRRRRNNNNNNNNNNN
Fig 12	Q1 no S.: NRRNRNRRRRRRNRRNNRRNNRNRNRNRNN Q1 auto S.: NNNNNNNNRRNNNNNNNNNNNNNNNNNNN Q1 man. S.: RRRNRRRRRRRRRRRRNRRNNNRNNRRRN Q2 no S.: NRNNNNNNNRRRNNNNRRNRRNNNNRNNNN Q2 auto S.: NNNNNNNNNNNRNNNNNRRNRNRNNNNNNR Q2 man. S.: RRRRNRNRRRNRRNRRRRRRRRRNRRNRR
Fig 13	Q3 no S.: RRNNNNNNNNRNRNNNNNNNNNNNRNNNNN Q3 man. S.: RRRRRNRRRNRRRRRNRNRNRNRNNNNNRR Q4 no S.: RRRRRRRRNNRRNRNRNRRRRNNRRNNNRR Q4 man. S.: RRRRRRNRRNRNNRNNRRRRNRRRRRNRRR
Table 5	Q1: RRRRRRNRRRRRRRRNNNRNRNRRRRRRRR Q2: RRRRRRRRRRRRRRNNRRRRRRRRNRRRNN Q3: RRNNNNNNNNRRNNNRRNRNNRRNNNRNNN Q4: RRRRRRNNRNRRRNRRRRNNNRRNNNNNRN Q5: RRRRRRRNRRNRRNNNRNNNNNNNNNNRNN

A APPENDIX

A.1 Field Indexing

The following types were assigned to the corresponding fields:

- **generalTextField:** Text
- **numeric_general:** Lifespan, Weight, Length, Gestation Period, Top Speed, Age of Sexual Maturity, Age of Weaning, Age Of Independence, Wingspan, Estimated Population Size, Incubation Period, Age Of Fledgling, Average Clutch Size, Average Spawn Size
- **shortTextField:** Fun Fact, Biggest Threat, Diet, Lifestyle, Location, Behavior, Age of Molting, Body, Natural Habitat, Features, Name, Kingdom, Phylum, Class, Order, Origin, Height, Family, Genus, Scientific Name, Name Of Young, Type
- **commaSeparatedText:** Predators, Other Name(s)
- **boolean:** Venomous, Migratory

A.2 Query Relevance Results

In the table 6 the relevance results for each graph presented in the report are condensed in a single string using as notation 'R' for relevant and 'N' for non-relevant, allowing for verification of all graphic results presented in this document. All strings have a length of 30 characters as the team always retrieved 30 entries on each query.

A.3 Manual Synonyms File

energetic, hardworking, industrious, tireless, untiring
bulldog, bulldogs
watchdog, watchdogs
sheepdog, sheepdogs
dog, canine, hotdog, weenie, wiener, barker, doggie, doggies, doggy, hound, canid, canines
cross, crossbreed, hybridize, interbreed
breed, bloodline, procreate, reproduce, crossbreeding, crossing, hybridization, hybridizations, hybridizing, interbreeding, breeds, cover, multiply, spawn, stock, strain
desirable, suitable, suited, worthy, fit, suit
incompatible, unsuited, bad for, not recommended
hunting, hunt, hound hunted, run, trace, explore, look, follow, research, hunter, hunters, huntsman, search, searches, seek, traces, tracing, hunts
america, americas, us, usa, mexico, canada, mexican, mexicans, north america
animal, animals, beast, brute, creature, fauna
like, love, fond of, enjoy, enjoys, delight, delights, loving, lovingly, loves, liked, liking, likable, fan
eat, eating, feast, feed, feeding, prey, hunt, consume, ingest
change, convert, exchange, replace, substitute, switch, shift
color, colour, tint
fur, skin, hide, pelt, fleece, pelage, pelages, pelts, skins, furs, hides, feather, feathers
purpose, motive, aim, reason, goal, intention, objective, point, target, use, utility, usefulness, usefulnesses
camouflage, disguise, disguises, camouflaging, hide, mask
walk, walked, wander, walking
group, grouping, groups, pack, packs, grouping, groupings, herd, herds, herding, herde
manage, deal, handle, treat
migrate, migrated, migrating, migration, migrations, migratory, migrates, move, travel

A.4 Graphical User Interface

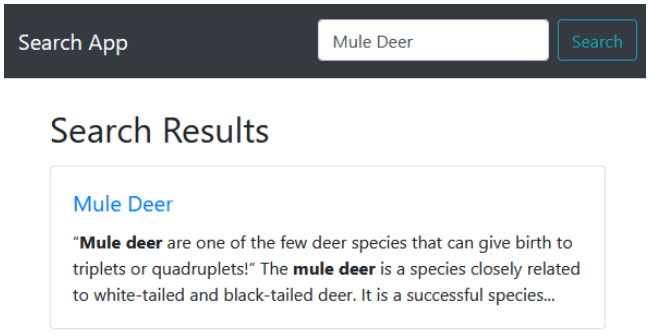

Figure 15: Graphical user interface home

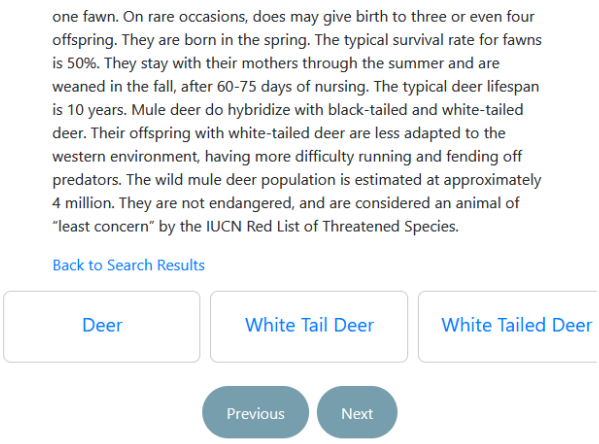

Figure 16: More like this UI