Zurich University
of Applied Sciences

School of
Engineering
InIT Institute of Applied
Information Technology

Project work (Informatic)

An interactive Web User Interface for creating database queries using natural language

Author	Gabriele Pace Timothé Laborie
Main supervisor	Prof. Dr. Kurt Stockinger Prof. Dr. Mark Cieliebak
Sub supervisor	Jan Milan Deriu
Date	18.12.2020

Erklärung betreffend das selbstständige Verfassen einer Projektarbeit an der School of Engineering

Mit der Abgabe dieser Projektarbeit versichert der/die Studierende, dass er/sie die Arbeit selbständig und ohne fremde Hilfe verfasst hat. (Bei Gruppenarbeiten gelten die Leistungen der übrigen Gruppenmitglieder nicht als fremde Hilfe.)

Der/die unterzeichnende Studierende erklärt, dass alle zitierten Quellen (auch Internetseiten) im Text oder Anhang korrekt nachgewiesen sind, d.h. dass die Projektarbeit keine Plagiate enthält, also keine Teile, die teilweise oder vollständig aus einem fremden Text oder einer fremden Arbeit unter Vorgabe der eigenen Urheberschaft bzw. ohne Quellenangabe übernommen worden sind.

Bei Verfehlungen aller Art treten die Paragraphen 39 und 40 (Unredlichkeit und Verfahren bei Unredlichkeit) der ZHAW Prüfungsordnung sowie die Bestimmungen der Disziplinarmassnahmen der Hochschulordnung in Kraft.

Ort, Datum:	**Name Studierende:**
Winterthur, den 18.12.2020	Gabriele Pace
Winterthur, den 18.12.2020	T Laborie

Abstract

Databases have a lot of structured data. Accessing this data is hard for non-programmers, because knowledge of a Structured Query Language (SQL) is required. In order to simplify this process and to allow laypersons to access the data, this work aims to develop a straightforward and comprehensive web-based application.

The user can ask a question in English, which is then processed by a neural network that is pre-trained on a specific database. The user then receives the corresponding SQL query, along with an operation tree and a generated question for debugging purposes.

The neural network needs a lot of training data to convert the human-written questions into SQL queries. Our website makes this easier by providing developers with an interface to test questions.

This work is based on the previous works of J. Deriu et al. [1] and N. Kaiser, P. Schläpfer [2], which provides a basic user interface with some hardcoded questions, along with a neural network to process those questions. However, it is not possible to ask custom questions, therefore our final work is to create a new user interface that allows the user to ask their own questions.

The result is a website with a user-friendly interface that makes it easy to ask questions in English and get the corresponding SQL query. The interface also allows the user to correct any mistakes that the neural network makes.

This work could later be expanded to make the system automatically execute the SQL query so the user can get the answer right away.

Zusammenfassung

Datenbanken haben viele strukturierte Daten. Der Zugriff auf diese Daten ist für Nicht-Programmierer schwierig, da Kenntnisse in einer strukturierten Abfragesprache (SQL) erforderlich sind. Um diesen Prozess zu vereinfachen und Laien den Zugriff auf die Daten zu ermöglichen, zielt diese Projektarbeit darauf ab, eine unkomplizierte und umfassende webbasierte Anwendung zu entwickeln.

Der Benutzer kann eine Frage auf Englisch stellen, die dann von einem neuronalen Netzwerk verarbeitet wird, das in einer bestimmten Datenbank vorab trainiert wurde. Der Benutzer erhält dann die entsprechende SQL-Abfrage zusammen mit einem Operationsbaum und einer generierten Frage für Debugging-Zwecke.

Das neuronale Netzwerk benötigt viele Trainingsdaten, um die von Menschen geschriebenen Fragen in SQL-Abfragen umzuwandeln. Unsere Website erleichtert dies, indem sie Entwicklern eine Schnittstelle zum Testen von Fragen bietet.

Diese Arbeit basiert auf die vorherigen Arbeiten von J. Deriu et al. [1] und N. Kaiser, P. Schläpfer [2], die eine grundlegende Benutzeroberfläche mit einigen fest codierten Fragen sowie ein neuronales Netzwerk zur Verarbeitung dieser Fragen bereitstellen. Es ist jedoch nicht möglich, benutzerdefinierte Fragen zu stellen. Daher besteht unsere letzte Arbeit darin, eine neue Benutzeroberfläche zu erstellen, über die der Benutzer seine eigenen Fragen stellen kann.

Das Ergebnis ist eine Website mit einer benutzerfreundlichen Oberfläche, die es einfach macht, Fragen auf Englisch zu stellen und die entsprechende SQL-Abfrage zu erhalten. Die Schnittstelle ermöglicht es dem Benutzer auch, Fehler zu korrigieren, die das neuronale Netzwerk macht.

Diese Arbeit könnte später erweitert werden, damit das System die SQL-Abfrage automatisch ausführt, sodass der Benutzer die Antwort sofort erhalten kann.

Table of contents

Abstract 2
Zusammenfassung 3
1. Introduction 5
1.1 Current status 5
1.2 Goal 5
2. Theoretical principles 6
2.1 SemQL 6
2.2 Tree structure 7
3. Our work 8
3.1 How literals are found 9
3.2 The generated tree 10
3.3 Fixing questions 11
3.4 Example questions 11
4. System Architecture 12
4.1 Overview 12
4.2 Technology 12
5. Example questions 13
5.1 Question 1 13
5.2 Question 2 14
5.3 Question 3 15
6. Results 17
6.1 Evaluation 17
7. Discussion and outlook 23
8. Appendices 24
8.1 User manual 24
8.1.1 Requirements 24
8.1.2 Local installation 24
8.1.3 Known issues 26
8.2 Server deployment 27
8.3 API 27
8.4 Questions 27
9. Bibliography 28
9.1 Sources 28
9.2 List of illustrations 29
9.3 List of tables 29

1. Introduction

The world generates over 160 petabytes of data per day [3] and this number is growing rapidly.

A lot of this data is stored in structured databases. Answering questions over structured data is a difficult task in natural language processing. It is usually approached by converting a natural language question into executable queries.

Accessing structured data is hard for non-programmers as it requires a Structured Query Language (SQL), which requires special training to use. For the purpose of simplifying this process and to allow laypersons to access the data, this work aims to develop a straightforward and comprehensive web-based application.

Using this application, the user will be able to ask a question in English, which will then be processed into a tree by a neural network that is pre-trained on a specific database. This tree is then converted to an SQL query, alongside a generated question for debugging purposes.

1.1 Current status

This project is based on two previous works. one that converts natural language into a tree form using a neural network, and another that converts the tree into a generated text and an SQL query using Python. These two works are by Deriu et al. [1] and Kaiser and Schläpfer [2].

Furthermore, Deriu already implemented a basic website to process some hardcoded questions and to see their representation as a tree.

1.2 Goal

The goal of this work is to create a website that allows the user to interact with the existing backend by writing custom questions. The website then needs to display the generated tree, the SQL query and finally the synthetic question, in user-friendly way. If the synthetic question is not semantically correct compared to the original, it should be possible to correct it using a button.

2. Theoretical principles

The basis of this web application is to formulate a question in natural language so that the neural network can generate an SQL query based on the written question. This is done using SemQL and some pre-trained language models.

2.1 SemQL

Semantic Query Language (SemQL) is used to convert the question into an Operation Tree (OT). It is an SQL-based language for databases.

In database schemes, the columns of tables have a name. SemQL uses “Qualified Attribute Names” instead so that the code remains implementation-independent [4]. This is a good feature for creating structurally independent database schemes.

SemQL allows defining of SQL-Like clauses such as:

- Condition
- Expression
- Order by Clause [4]

Supported atomic operations:

- `GET-DATA`: Gets a set of data from the database and returns it. Not defined in bag algebra.
- `EXTRACT-VALUES`: extracts a column from a table with attributes and outputs their values as a list. This operation is not defined in rational databases.
- `FILTER`: Filters a data node according to a condition. The condition is specified on an attribute and uses one of the operations {`<`, `≤`, `≥`, `>`, `=`, `≠`, `BETWEEN`}.
- `MERGE`: Connects two data nodes. It corresponds to the equi-join of two Tables from the relational bag algebra.
- `DISTINCT`: Removes duplicates from a list of values.

Supported aggregates operations:

- `SUM`: Calculates the sum of a list of numerical values.
- `AVERAGE`: Calculates the average from a list of numerical values.
- `MAX`: Selects the largest value from a list of numerical values.
- `MIN`: Selects the smallest value from a list of numerical values.
- `COUNT`: Returns the number of entries present in a data node.

Supported Set operations:

- UNION: Combines two or more result sets into a single set, without duplicates [5].
- INTERSECT: Takes the data from both result sets which are in common [5].
- DIFFERENCE: Returns all records from the first “SELECT” statement that are not in the second “SELECT” statement [6].

After the OT is generated, it can be used to generate the text-based question, using a system known as Tree2Text. It recursively converts the operations into text. For example:

	Text2Tree
SUM	*[[sum(T, A)]]* = What is the total [[A]] of [[T]]?
AVERAGE	*[[average(T, A)]]* = What is the average [[A]] of [[T]]?
DONE	*[[done(R)]]* = What are the [[R]]?
COUNT	*[[count(R)]]* = How many [[R]] are there?

Table 1: Tree2Text conversion example

All these operations are shown in the generated tree of the Web UI.

2.2 Tree structure

The neural network converts questions into a tree structure (a “logging tree”). Every node corresponds to an operation. This type of tree starts at the bottom and ends at the top. [2]

The different operations are explained in the “SemQL” section.

For a better visualisation, the nodes that are used to generate text are coloured blue. The red nodes do not generate any text and are only used for the SQL query.

The following Illustration shows how the logging tree is displayed in the Web interface.

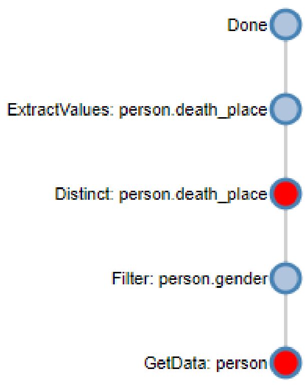

Done
ExtractValues: person.death_place
Distinct: person.death_place
Filter: person.gender
GetData: person

Illustration 1: Example Logging Tree structure in Web UI

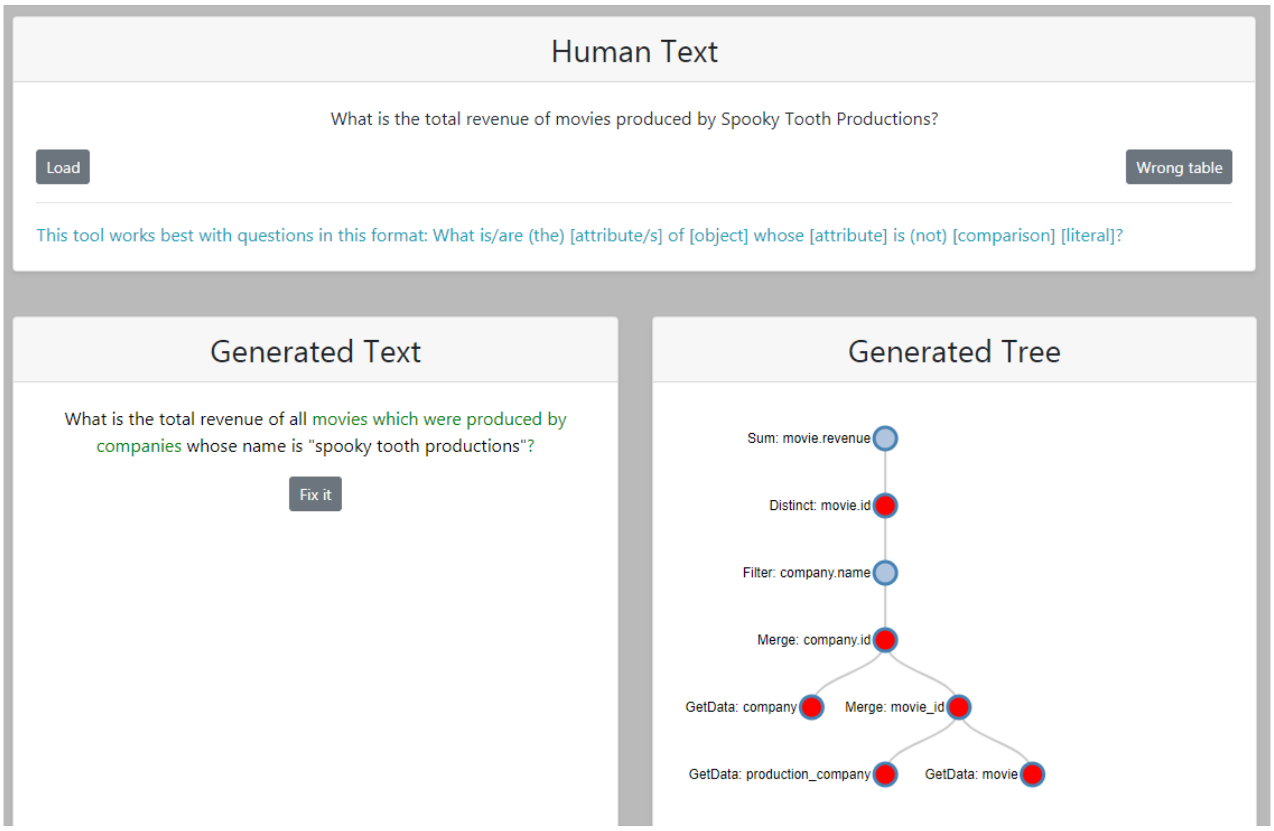
Trees are an optimal visualisation for inexperienced users as they are easy to understand.

3. Our work

We started with a spreadsheet made by previous contributors containing 208 questions. Each question also had a corresponding tree (known as the gold tree) and was used to train the neural network.

Initially, we created a website to compare the human-made data with the generated data. We then improved it so that it became possible to ask custom questions.

The final user interface looks like this:



Human Text

What is the total revenue of movies produced by Spooky Tooth Productions?

This tool works best with questions in this format: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

Generated Text

What is the total revenue of all movies which were produced by companies whose name is "spooky tooth productions"?

Generated Tree

Illustration 2: Final web user interface

The user can write a custom question at the top and then click on “Load” to process it. The question is sent to the backend over an Application Programming Interface (API) where the neural network generates the tree. Some Python code that had been implemented by previous contributors then generates the corresponding SQL query and a natural language question using the tree. The newly generated natural language question is useful for debugging in case the neural network fails to generate a good tree. In this case, a correct tree is generated, as we can see in the above screenshot.

The backend already included an API to train the neural network. In order to add support for custom questions, a new API function needed to be implemented. It is important that this API function allows requests from any origin in order for the frontend to be able to communicate with the backend.

While experimenting with the system, an optimal question structure was discovered:

What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

Formulating questions using this structure is not mandatory, but it greatly reduces the chance that the neural network makes a mistake.

- `COMPARISON`: Refers to operators such as "`EQUAL`" or "`LESS THAN`".
- `OBJECT`: Refers to a database table, such as "`person`".
- `ATTRIBUTE`: Refers to an attribute of a table such as "`birthday`".
- `LITERAL`: Refers to an attribute of a table such as "`birthday`".

3.1 How literals are found

Questions usually contain a literal. In the screenshot, the literal is "Spooky Tooth Productions". The neural network does not figure out what the literal is, so initially "None" was used as a placeholder. To show the literal in the generated text some new functionality needed to be implemented.

First, an inverted index of the database was generated using C#.

This index contains the list of all the literals that exist in the database along with the tables that they exist in. For example, the literal "the wiz" shows up in "`cast.character`" and "`movie.title`".

Next, some Python code goes through all these literals and finds the biggest one that exists in the original text, in this case, "Spooky Tooth Productions". Only the literals that are in the correct table are considered. As can be seen in the generated tree, the filter node is looking for the name of a person so the system only looks at person names.

Numbers, if there are any, are always considered to be literal.

If the question contains the string "whose name is", the text afterwards is always considered to be the literal.

If the question contains quotes, the text in those quotes is always considered to be the literal. This can be used as a failsafe in case the literal detection worked incorrectly.

This system is far from perfect and often makes mistakes. In order to improve it, future contributors could attempt to modify the neural network so that it can also detect literals.

3.2 The generated tree

Another feature of the website is the ability to click on the blue operation nodes to highlight the words within the generated text that the node generated. The red arrow in the screenshot shows what this looks like:

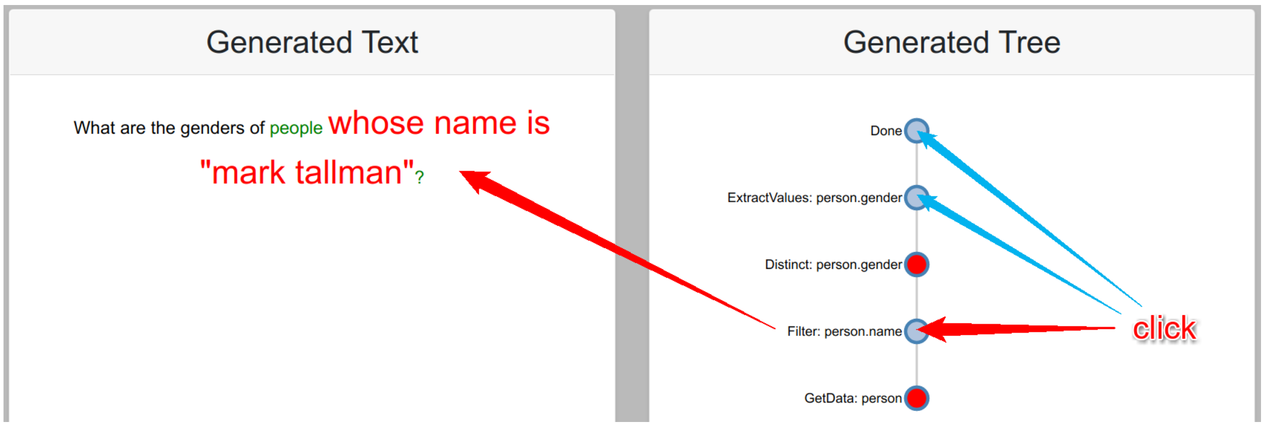


Illustration 3: Clicking on an operation node in the Generated Tree

Implementing this required some modifications to the backend so that the generated question keeps track of which node generated which words.

The red operation nodes do not generate any text, so they cannot be clicked on.

The green text is not generated by the nodes, it is part of a template which is unique for every attribute. These templates were generated by previous contributors.

3.3 Fixing questions

If the generated text is partially incorrect, it is possible to fix it by clicking on the “Fix it” button.

After clicking on the button, the generated question will be copied to the “Human Text” panel above:

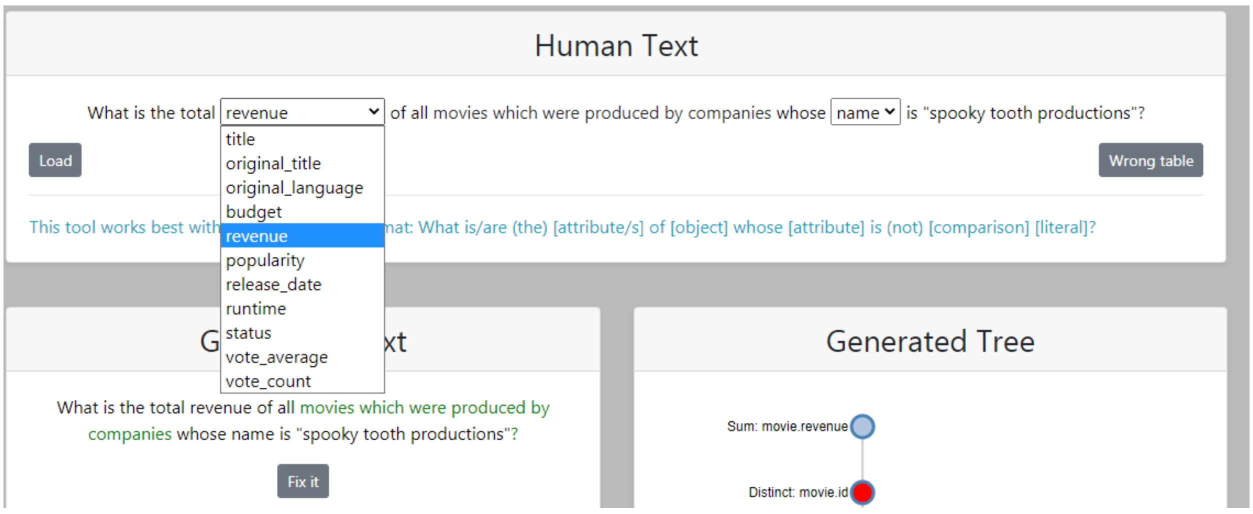

Human Text
What is the total revenue of all movies which were produced by companies whose name is "spooky tooth productions"?
title
original_title
original_language
budget
revenue
popularity
release_date
runtime
status
vote_average
vote_count
Load
Wrong table
This tool works best with … mat: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?
Generated Text
What is the total revenue of all movies which were produced by companies whose name is "spooky tooth productions"?
Fix it
Generated Tree
Sum: movie.revenue
Distinct: movie.id

Illustration 4: Changing the attributes in the Human Text (dropdown menu)

It is now possible to correct any wrong attributes by selecting them in a dropdown menu. Another feature of our work is that the attributes displayed in the dropdown menu are relevant to the question topic, so no irrelevant attributes are displayed. This makes it easy for the user to select attributes. If the neural network incorrectly used the wrong table, the user can click on the “Wrong table” button and fix the text themselves.

After clicking on “Load” the question will be processed again.

3.4 Example questions

By scrolling down on the website (*see* 8.2 Server deployment, “Frontend Server link”), the user can see 208 example questions that were written by previous contributors.

4. System Architecture

4.1 Overview

Here is the current system architecture.

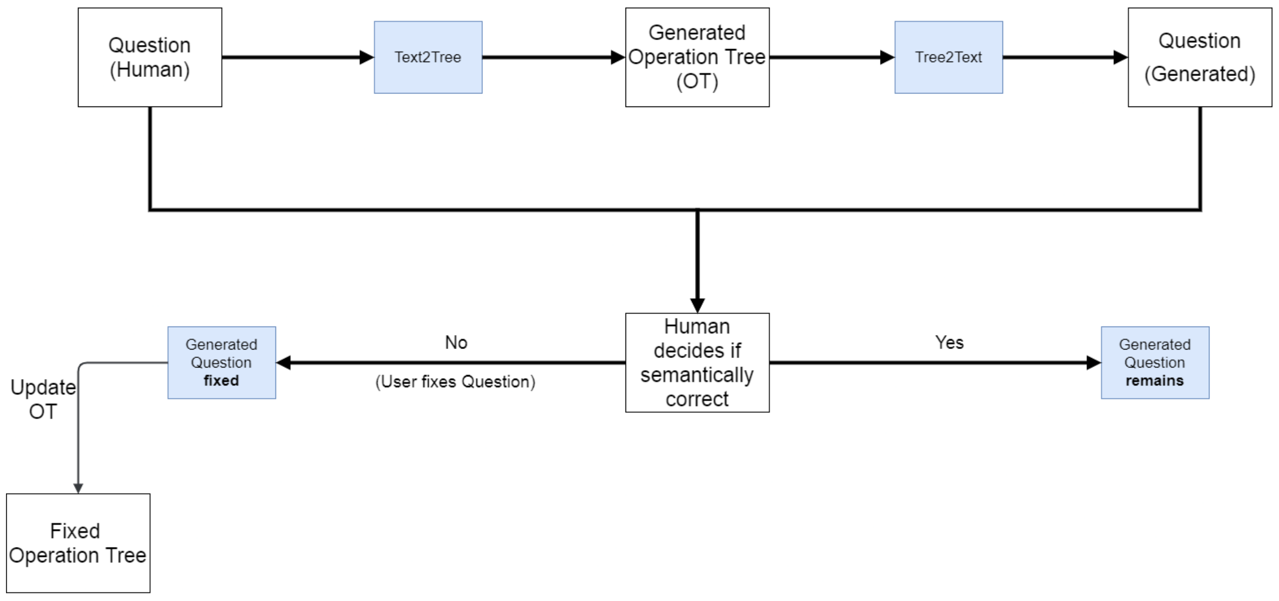

Question (Human)
Text2Tree
Generated Operation Tree (OT)
Tree2Text
Question (Generated)
Human decides if semantically correct
No
(User fixes Question)
Yes
Generated Question **fixed**
Generated Question **remains**
Update OT
Fixed Operation Tree

Illustration 5: Overview of project work

4.2 Technology

The backend is based on the work of Deriu et al. [1] and is made in Python. C# was used to generate the inverted index.

The frontend part of the application uses the following technologies:

- jQuery (JavaScript)
- RiotJS (JavaScript framework)
- Bootstrap (CSS)

5. Example questions

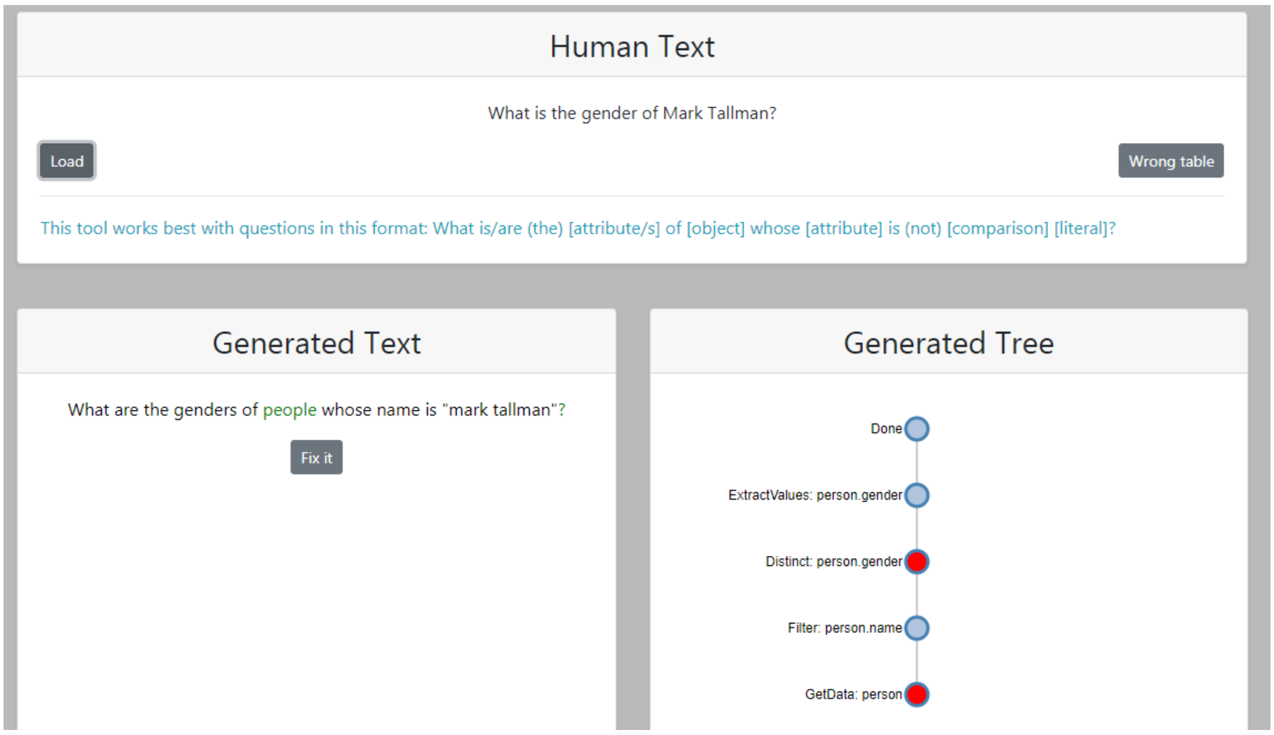
This part explains how the application works in practice. This will be demonstrated with three questions:

- **Question 1:** What is the gender of Mark Tallman?
- **Question 2:** How old is Mark Tallman?
- **Question 3:** Where did non male persons die?

5.1 Question 1

The human-written question is:

What is the gender of Mark Tallman?


Human Text

What is the gender of Mark Tallman?

This tool works best with questions in this format: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

Generated Text

What are the genders of people whose name is "mark tallman"?

Generated Tree

Done

ExtractValues: person.gender

Distinct: person.gender

Filter: person.name

GetData: person

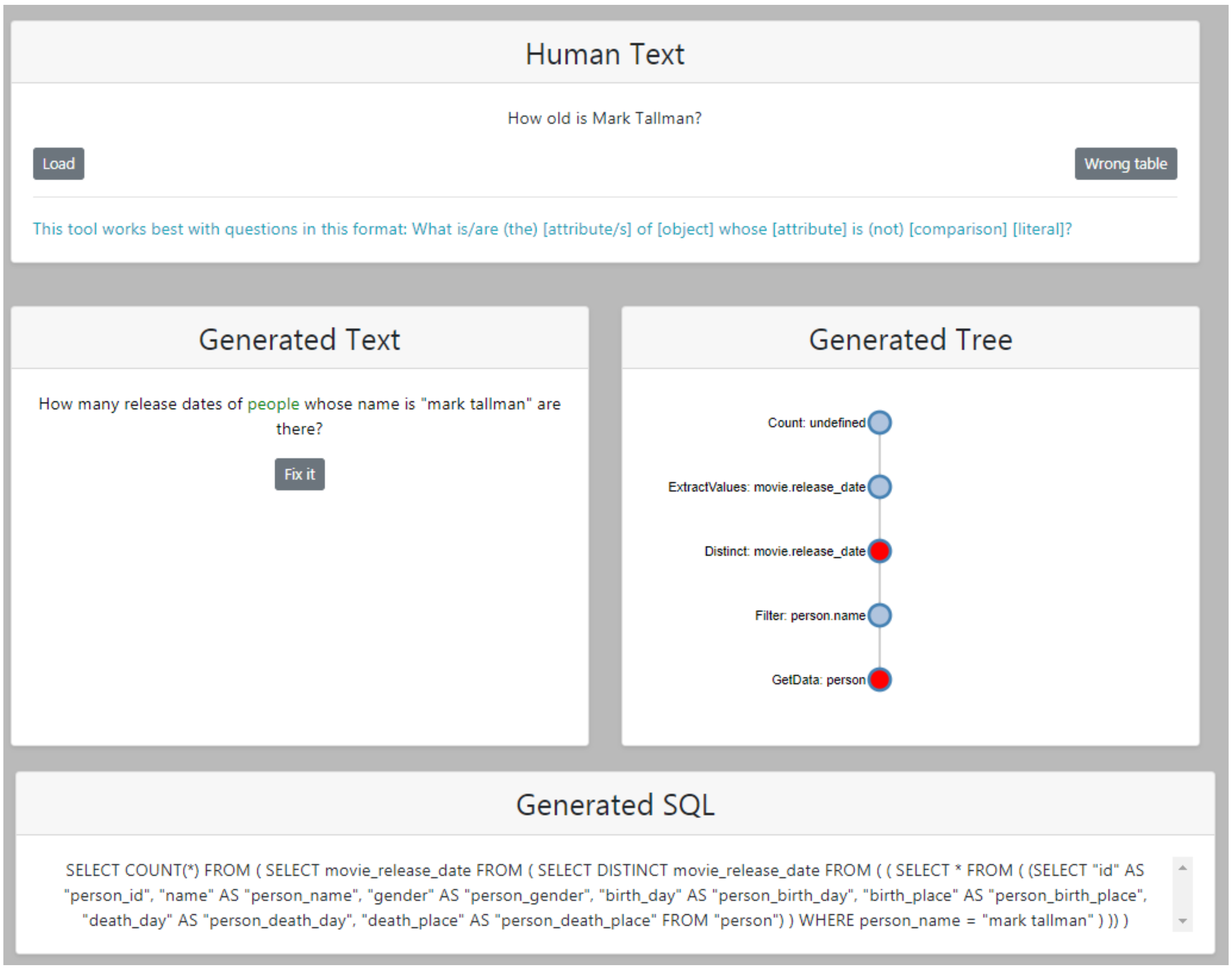
Illustration 6: Screenshot of Question 1

The generated question is semantically the same, and the tree was also generated correctly.

5.2 Question 2

The human-written question is:

How old is Mark Tallman?



How old is Mark Tallman?

This tool works best with questions in this format: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

How many release dates of people whose name is "mark tallman" are there?

Count: undefined
ExtractValues: movie.release_date
Distinct: movie.release_date
Filter: person.name
GetData: person

```
SELECT COUNT(*) FROM ( SELECT movie_release_date FROM ( SELECT DISTINCT movie_release_date FROM ( ( SELECT * FROM ( (SELECT "id" AS "person_id", "name" AS "person_name", "gender" AS "person_gender", "birth_day" AS "person_birth_day", "birth_place" AS "person_birth_place", "death_day" AS "person_death_day", "death_place" AS "person_death_place" FROM "person") ) WHERE person_name = "mark tallman" ) )) )
```

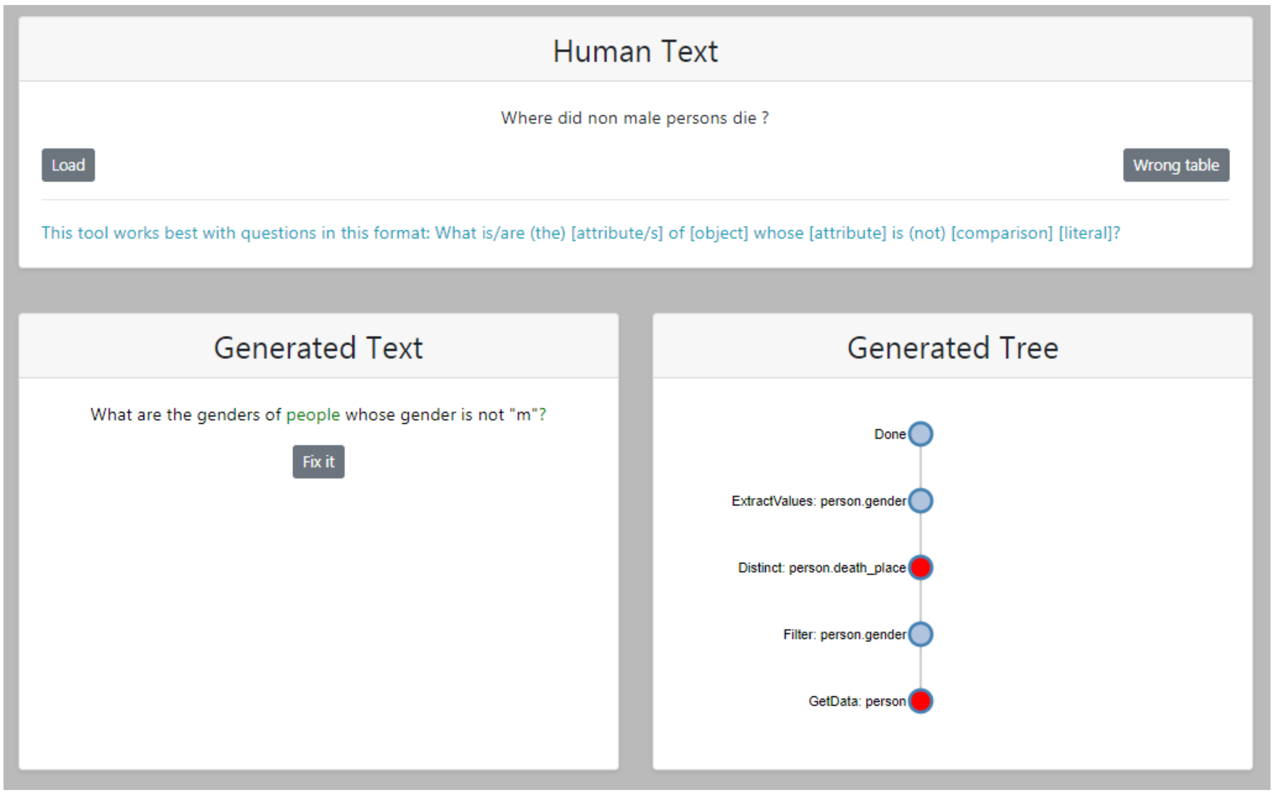
Illustration 7: Screenshot Question 2

The neural network was not successful at generating the tree. As a result, the generated question and SQL query are also wrong. This is usually caused by a lack of training data that uses a certain word. In this case, the neural network has only ever seen the word "old" in the context of movies. In situations like these, the "fix it" button can be used to fix the question. If we instead ask the question "What are the birthdays of people whose name is Mark Tallman?", then a correct tree is generated.

5.3 Question 3

The human-written question is:

Where did non male persons die?


Human Text

Where did non male persons die ?

This tool works best with questions in this format: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

Generated Text

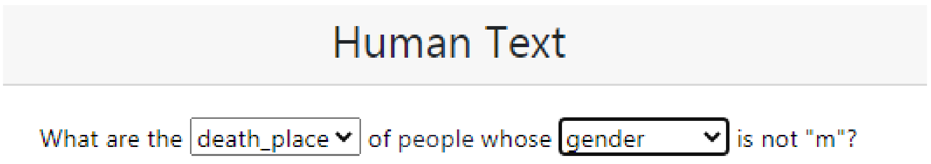
What are the genders of people whose gender is not "m"?

Generated Tree

Illustration 8: Screenshot Question 3

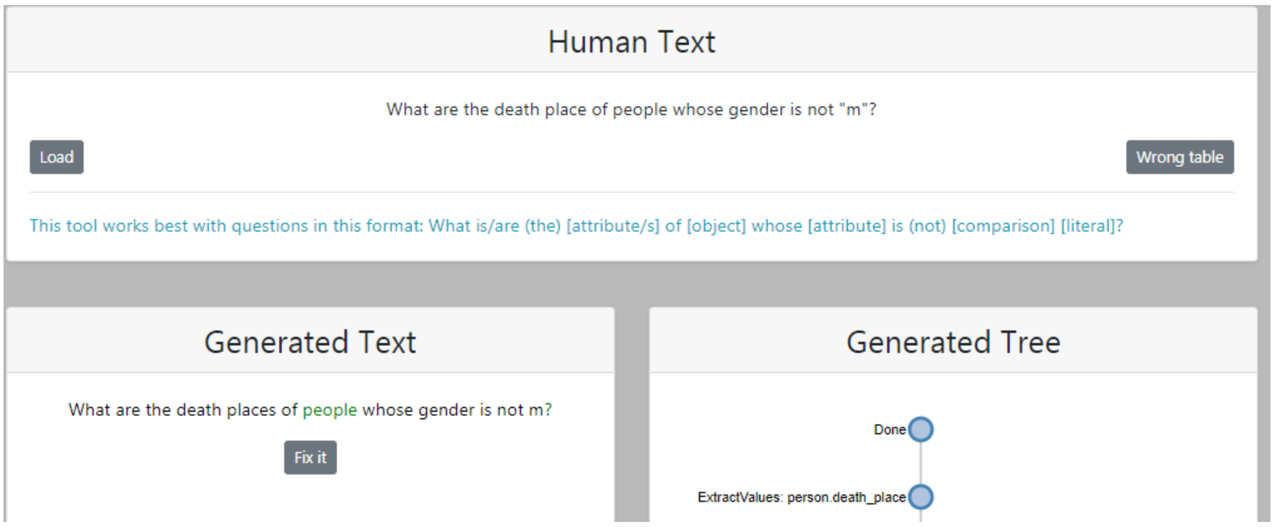
An attribute in the tree was not correctly generated. In this case “ExtractValues” is wrong, and this caused the text to be wrong as well.

Using the “Fix it” button the wrong attribute can be changed to the right one and the question is then processed correctly.


Human Text

What are the death_place of people whose gender is not "m"?

Illustration 9: Question 3 fixed part in Human Text


Human Text

What are the death place of people whose gender is not "m"?

This tool works best with questions in this format: What is/are (the) [attribute/s] of [object] whose [attribute] is (not) [comparison] [literal]?

Generated Text

What are the death places of people whose gender is not m?

Generated Tree

Illustration 10: Screenshot Question 3 fixed

6. Results

6.1 Evaluation

We evaluated how often the website can be used to fix the mistakes that the neural network makes during the conversion of natural language to trees.

30 questions were evaluated. 25 of these are of previous contributions, from a file containing 208 questions (*see* 8.4 Questions). We formulated the rest. We chose diverse questions to test the system's handling of them and to test the detection of literals.

24 out of these 30 questions were correctly processed, 3 were partially correct (one of the attributes was wrong) and 3 were completely wrong. 2 of the questions that were processed incorrectly were possible to fix using the website. One of them needed to be completely rewritten in order to be processed correctly.

The questions that were evaluated correctly are the following:

Original question	**Generated question**
What are the names of people whose birthday is 1938?	What are the names of people who were born on 1938?
What is the total revenue of movies produced by "Spooky Tooth Productions"?	What is the total revenue of all movies which were produced by companies whose name is Spooky Tooth Productions?
What is the total revenue of movies produced by "rrr"?	What is the total revenue of all movies which were produced by companies whose name is rrr?
What is the gender of Mark Tallman?	What are the genders of people whose name is "mark tallman"?
What are the genders of people whose name is Justin Niessner?	What are the genders of people whose name is "Justin Niessner"?
How many death places are there for persons that have won an Oscar for a movie that has a runtime over 25?	What are the death places of people who won oscars for their contribution to movies with a runtime of at most 25 minutes?
What are the revenues of movies with a vote count of 2607 or less?	What are the revenues of movies with a vote count of at most 2607?
What are the revenues of movies with a vote count of 260123127 or less?	What are the revenues of movies with a vote count of at most 260123127?

Where did non female persons die?	What are the death places of people whose gender is not "f"?
What are the popularities of movies whose name is "Austin Powers in Goldmember"?	What are the popularities of movies whose name is Austin Powers in Goldmember?
What are the names of movies whose release date is lower than 2010?	What are the names of movies who were released before 2010?
What is the average budget of movies with a runtime of at least 177?	What is the average budget of all movies with a runtime of at least 177 minutes?
What are the genders of people whose name is rrr?	What are the genders of people whose name is "rrr"?
What are the birthdays of people whose name is Mark Tallman?	What are the birth days of people whose name is "Mark Tallman"?
What is the budget of the movie with the maximum revenue among all movies that have more than 1101 votes?	What are the budgets of movies with a vote count of more than 1101 with maximum revenue?
How many original languages of movies are there for a movie budget over 140000000?	How many original languages of movies with a budget of "140000000" dollars are there?
Which companies produced movies with a popularity of more than 0.545825?	What are the names of companies which produced movies with a popularity of more than 0.545825?
Are there titles of movies that were not produced by the company "Ministère du Développement Durable et de la Mer"?	Are there any names of movies whose name is not Ministère du Développement Durable et de la Mer?
What is the average revenue of all produced movies except the movies produced in Italy?	What is the average revenue of all movies which were produced in countries whose name is not "italy"?
Are there any names of companies which produced movies whose status is "Angola"?	Are there any names of companies which produced movies whose status is Angola?
What is the total runtime of all movies whose name is Aloha Pictures?	What is the total runtime of all movies whose status is "Aloha Pictures"?
What are the names of movies which were produced by companies whose name is "Entropy"?	What are the names of movies which were produced by companies whose name is Entropy?
What are the names of genres comprising movies with a runtime of more than 92 minutes?	What are the names of genres comprising movies with a runtime of 92 minutes?

Are there any movies for which male people won Oscars?	Are there any names of movies for which people whose gender is "m" won oscars?

Table 2: Correctly evaluated questions

These questions were not processed correctly:

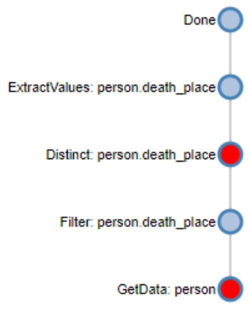
Original question	Generated question	Explanation	Fixed generated question
What are the names of people whose death place is not “Brighton England”?	What are the names of people whose name is not Brighton England?	“death place” was not detected correctly. This can be fixed by changing the question to “*What are the names of people whose death place is not Brighton England?*”	What are the names of people who were not deceased in Brighton England?
Tree  Done ExtractValues: person.death_place Distinct: person.death_place Filter: person.death_place GetData: person			

Table 3: Wrongly evaluated question 1

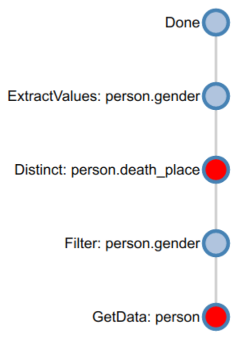
Original question	Generated question	Explanation	Fixed generated question
Where did non male persons die?	What are the genders of people whose gender is not "m"?	"Where" was not detected correctly. This can be fixed by changing the question to "*What are the death place of people whose gender is not "m"?*"	What are the death places of people whose gender is not m?
Tree  Done ExtractValues: person.gender Distinct: person.death_place Filter: person.gender GetData: person			

Table 4: Wrongly evaluated question 2

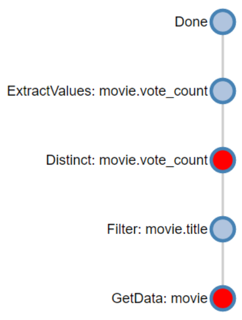
Original question	Generated question	Explanation	Fixed generated question
What are the vote counts for movies whose name is Whipped?	What are the vote counts of movies whose status is "Whipped"?	"name" was not detected correctly. This can be fixed by changing the question to "*What are the vote count of movies whose title is "Whipped"?*"	What are the vote counts of movies whose name is Whipped?
Tree  Done ExtractValues: movie.vote_count Distinct: movie.vote_count Filter: movie.title GetData: movie			

Table 5: Wrongly evaluated question 3

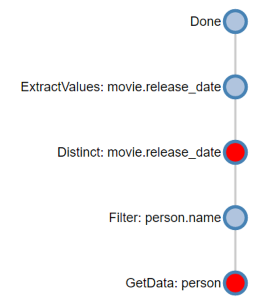
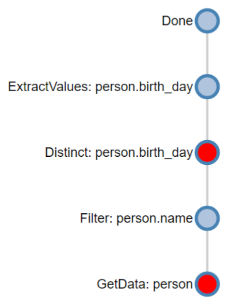
Original question	Generated question	Explanation	Fixed generated question
How old is Mark Tallman?	What are the release dates of people whose name is "mark tallman"?	“How old is” was not detected correctly. This can be fixed by changing the question with Wrong table to “*What are the birthdays of people whose name is "mark tallman"?*”	What are the birth days of people whose name is mark tallman?
Tree (before)  Done ExtractValues: movie.release_date Distinct: movie.release_date Filter: person.name GetData: person		**Tree (after)**  Done ExtractValues: person.birth_day Distinct: person.birth_day Filter: person.name GetData: person	

Table 6: Wrongly evaluated question 4

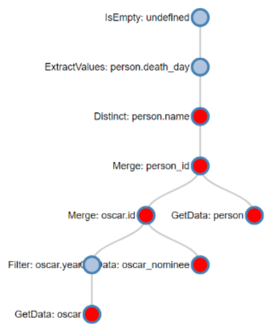
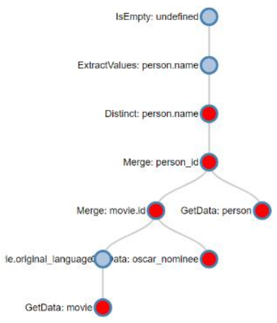
Original question	Generated question	Explanation	Fixed generated question
Is there any person who was nominated for the oscar before or in the year 1978?	Are there any death days of people who were nominated for oscars who were awarded before 1978 for their contribution to movies?	“Is there” was not detected correctly. We tried changing “death days” to “names” in the question by writing “*Are there any names of people who were nominated for oscars which were awarded before 1978 for their contribution to movies?*” but the tree is still generated incorrectly.	-
Tree (before)  IsEmpty: undefined ExtractValues: person.death_day Distinct: person.name Merge: person_id Merge: oscar.id GetData: person Filter: oscar.year ata: oscar_nominee GetData: oscar		**Tree (after)**  IsEmpty: undefined ExtractValues: person.name Distinct: person.name Merge: person_id Merge: movie.id GetData: person le.original_language ata: oscar_nominee GetData: movie	

Table 7: Wrongly evaluated question 5

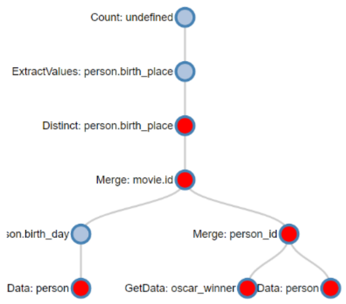
Original question	Generated question	Explanation	Fixed generated question
How many unique birth places are there from oscar winners who were born in 1990 or after?	error on the backend	Cannot be fixed because the Tree2Text system could not process the tree, as it is invalid.	-
Tree  Count: undefined ExtractValues: person.birth_place Distinct: person.birth_place Merge: movie.id son.birth_day Merge: person_id Data: person GetData: oscar_winner Data: person			

Table 8: Wrongly evaluated question 6

This section does not test the detection of literals.

7. Discussion and outlook

Our website can be used to simplify the work of people who regularly need to obtain data from databases. The website is hosted on a server from the ZHAW and can be accessed using the frontend link (*see* 8.2 Server deployment).

We were successful at building a user-friendly tool to test custom questions. The tree visualization works very well.

We were less successful at detecting the literals however. Our naive approach using an inverted index makes a lot of mistakes.

The evaluation made in the Results chapter revealed useful information about which questions the neural network is capable of processing correctly.

Here is an outlook on how the system could be further developed with some proposals for its subsequent use.

Future contributors could improve the website so that the generated SQL query is automatically ran on the database. This would then allow users to see the answers to their questions.

The neural network could be improved further by collecting data whenever a user fixes an incorrectly processed question. This data could then be used to train the neural network by increasing the number of questions in the training set. Currently the neural network struggles with certain words as it has not seen them very often in the training data.

The literal detection could be improved by modifying the existing neural network so that it also detects literals. This would likely be a much more accurate approach than the current system. Alternatively, the named entity recognition system from libraries like Spacy could be used.

The view of the example questions could be improved so that instead of scrolling down, the questions are listed in a dropdown menu.

The website could be improved by adding an interactive tutorial so that inexperienced users can learn how to use it.

Being able to change the database directly on the website would make it more flexible.

8. Appendices

8.1 User manual

8.1.1 Requirements

Hardware and operating system Requirements:

- **OS:** Windows 10 or Linux distribution (e.g., Ubuntu 20.04 LTS)
- **RAM:** 16GB
- **CPU:** 4-Cores
- **GPU:** 1GB
- **Storage:** 10GB

Software Requirements:

- Python 3.7
- Anaconda (Python environment)
- Pip (Package installer for Python)
- Pytorch (Python library)
- Node.js
- Git

8.1.2 Local installation

The backend can be downloaded here:

https://github.com/lihlith-semql/semql-clients/tree/test_to_tree

The frontend can be downloaded here:

https://github.com/Timotheeee/SEMQL_Website/tree/localhost_installation

8.1.2.1 Instructions to install the backend

Start by cloning the repository (copy the following command without the “$” symbol):

```
$ git clone https://github.com/lihlith-semql/semql-clients.git --branch
test_to_tree
```

Create a new conda environment and install the modules in requirements.txt:

```
$ conda create --name semql --file semql-
clients/program_autoencoder/active_learning/nubia/requirements.txt
python=3.7
```

You may have to adapt that file to your operating system. This environment was created using Windows 10 64-bit.

Next, you need to install the submodules:

```
$ git submodule add https://github.com/lihlith-semql/semql-data.git
semql_data
```

```
$ git submodule add https://github.com/lihlith-
semql/semql/tree/text_to_tree semql
```

```
$ git submodule add https://github.com/lihlith-semql/semql-expert-api.git
expert_api
```

In some cases, you may need to clone each submodule separately.

Install the SemQL library:

```
$ cd semql/

$ python setup.py install

$ cd ../
```

Install the Expert API:

```
$ cd expert_api/

$ python setup.py install

$ cd ../
```

Make sure to run the install commands in the correct folder.

Install spacy (the model):

```
$ pip install -U spacy
$ python -m spacy download en_core_web_sm
```

Install torch: https://pytorch.org

Optionally, change the model by editing "`annotation_app.json`".

Run the server using

```
$ python run.py
```

8.1.2.2 *Instructions to install the frontend*

Use branch "`localhost_installation`":

```
$ git clone https://github.com/Timotheeee/SEMQL_Website --branch
localhost_installation
```

Open another Terminal and execute:

```
$ npm install
```

```
$ npm start
```

The website will be available at:

http://localhost:8001 or http://127.0.0.1:8001

If you encounter problems during installation, see the next chapter.

8.1.3 Known issues

During the local installation of the backend, when using "`python run.py`", the terminal may write an error message like:

"`No module found: <moduleName>`".

This generally happens because the requirements have not been correctly installed.

In case you still run into another "`No module found`" error, simply install the required module:

```
$ pip install <moduleName>
```

8.2 Server deployment

It is possible to use our work on the ZHAW infrastructure.

Frontend Server link: http://pait-semql.herokuapp.com

Backend Server IP-Address: 160.85.252.156 (https://apu.cloudlab.zhaw.ch)

8.3 API

The API to process questions can be found in the file. This file also contains the system to detect literals.

“`semql-clients/templates/src/api/display_tree.py`”.

8.4 Questions

The path with all the questions can be found in the file

“`SEMQL_Website/annotation_data.csv`”.

9. Bibliography

9.1 Sources

[1] J. Deriu, K. Mlynchyk, P. Schläpfer, A. Rodrigo, D. von Grünigen, N. Kaiser, K. Stockinger, E. Agirre and M. Cieliebak, “A Methodology for Creating Question Answering Corpora Using Inverse Data Annotation,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Seattle, 2020.

[2] N. Kaiser and P. Schläpfer, “Interactive Konstruktion von Datenbankabfragen,” in *ZHAW Zurich University of Applied Sciences*, Winterthur, 2019.

[3] J. Milenkovic, “30 Eye-Opening Big Data Statistics for 2020: Patterns Are Everywhere,” [Online]. Available: https://kommandotech.com/statistics/big-data-statistics. [Accessed 14. December 2020].

[4] Semarchy xDM, “Semarchy xDM SemQL Reference Guide,” 17. November 2020. [Online]. Available: https://www.semarchy.com/doc/semarchy-xdm/semsq.html. [Accessed 27. November 2020].

[5] Dot Net Tutorials, “Differences Between UNION EXCEPT and INTERSECT Operators in SQL Server,” [Online]. Available: https://dotnettutorials.net/lesson/differences-between-union-except-and-intersect-operators-in-sql-server. [Accessed 27. November 2020].

[6] TechOnTheNet, “SQL Server: EXCEPT Operator,” [Online]. Available: https://www.techonthenet.com/sql_server/except.php. [Accessed 27. November 2020].

9.2 List of illustrations

Illustration 1: Example Logging Tree structure in Web UI 7
Illustration 2: Final web user interface 8
Illustration 3: Clicking on an operation node in the Generated Tree........ 10
Illustration 4: Changing the attributes in the Human Text (dropdown menu) 11
Illustration 5: Overview of project work 12
Illustration 6: Screenshot of Question 1 13
Illustration 7: Screenshot Question 2 14
Illustration 8: Screenshot Question 3 15
Illustration 9: Question 3 fixed part in Human Text 15
Illustration 10: Screenshot Question 3 fixed........ 16

9.3 List of tables

Table 1: Tree2Text conversion example........ 7
Table 2: Correctly evaluated questions........ 19
Table 3: Wrongly evaluated question 1........ 19
Table 4: Wrongly evaluated question 2........ 20
Table 5: Wrongly evaluated question 3........ 20
Table 6: Wrongly evaluated question 4........ 21
Table 7: Wrongly evaluated question 5........ 21
Table 8: Wrongly evaluated question 6........ 22