Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Master's Thesis

# Applications of Semantic Networks to Fundamental Physics

**Patrick Richter**

E-Mail: patrick-richter@posteo.de
Matr.-Nr.: 6765940

First Supervisor: Prof. Dr. Gregor Kasieczka
Second Supervisor: Prof. Dr. Peter Schleper

# Abstract

Semantic networks are a powerful tool for representing knowledge in a structured manner. Their possible uses are diverse and range from the creation of knowledge databases to automated reasoning. Physics, on the other hand, is a truly sophisticated science of immense complexity and depth that describes our real world using mathematical models. This work examines how physical findings can be represented using semantic networks. It uses various approaches to structure physics knowledge in different ways. Modern methods of generative AI are used to create and evaluate the semantic network data. This also provides informative insights into the internal physics understanding of large language models.

# Contents

# 1. Introduction

## 1.1. A brief history of knowledge representation in physics

Physics is a diverse and fascinating science that humanity has been studying for many centuries. The driving force is always the desire to find models for our world to explain the numerous physical phenomena observed in various experiments. The physical worldview accessible to us today has been formed over time from many considerations and observations and is subject to constant change. Throughout history, physical theories have often been replaced by others. It is just as possible that the theories that best describe our world according to current knowledge will be replaced by better ones in the future. New theories can only be developed because they build on the existing knowledge base of previous theories. Therefore, an essential part of science is to preserve the knowledge that has already been gained and link it with new information. This process is the basis for any kind of progress.

Even if the basic principle remains the same, the methods of preserving and linking knowledge change over time. While in the early history of humanity, the transmission of knowledge was only possible orally, the art of writing developed around 3500 BC [1]. Only through this invention is it possible for us today to understand the findings of ancient physicists such as Archimedes, who, over the centuries, inspired many people to conduct their own research. Another milestone in the transmission of knowledge was Johannes Gutenberg's invention of the modern printing press in the middle of the 15th century. This invention made it possible to distribute and share new physical findings with a much larger audience and to open a broad discourse about them. Early modern astronomers such as Nicolaus Copernicus benefited from this invention [2]. The availability and dissemination of their findings accelerated scientific progress in the following years.

The first scientific journal, "The Philosophical Transactions", was founded in 1665 by the Royal Society of London [3]. It initiated a culture of scientific publishing that has been maintained to this day. In the second half of the 19th century, the first scientific journals were founded that were dedicated exclusively to physics, which established physics as a separate discipline of science [4].

The possibility of disseminating knowledge received a massive boost through the invention of the electronic computer and the subsequent development of the Internet. The HTML format, which forms the basis for the modern World Wide Web, was developed in 1990 by the physicist Tim Berners-Lee [5]. It was initially designed primarily for the

dissemination of scientific knowledge. The first Website outside of Europe and the first database on the web was the Stanford Physics Information Retrieval System (SPIRES), which is the predecessor of INSPIRE-HEP, a digital library for papers about particle physics [6].

In 2017, the Google Brain team introduced the transformer architecture [7], which is the basis for large language models, a new type of artificial intelligence that can answer questions by using their internal knowledge representation. Since then, many different large language models have been developed, some of which are specifically trained on physics data like the Xiwu model [8], which is specialized in high energy physics data, and the Cosmosage model [9], which is specialized on cosmology data. Despite ongoing research, it is still not fully understood how the knowledge is encoded in these models, and researchers often refer to them as black boxes [10].

For this reason, there is a need for more transparent and interpretable knowledge representation methods that can store large amounts of physics knowledge in a way that both humans and machines can understand. A promising approach is the use of semantic networks. In this thesis, different methods for representing physics knowledge as semantic networks will be explored and evaluated.

## 1.2. What is a semantic network?

A semantic network is a mathematical graph that represents some abstract meaning [11] [12]. It consists of nodes representing concepts and edges representing relationships between them.

There are many different types of semantic networks [12]. They differ in what kind of concepts and connections are allowed and how their structure is to be interpreted in terms of meaning. Some consist of directed edges, while others are non-directed graphs. Some are organized as a tree structure, while others are not hierarchical. Some are mathematically formal, while others only suggest a freely interpretable meaning. There are semantic networks that allow a wide range of possible connections, while others are restricted to a fixed set of edge types. There is a lot of disagreement in the literature about what counts as a semantic network and what does not [13]. Some sources also list neural networks as examples of semantic networks [12].

One common type of semantic network that is easy to read and understand is a network built of semantic triples of natural language. Such semantic triples consist of three elements: subject, predicate, and object. The predicate connects the two concepts of the subject and the object into a natural language sentence. An example of a semantic triple is the sentence "Physics is a subfield of science," which can be represented as the triple $\langle$ *Physics* | *is a subfield of* | *Science* $\rangle$. Figure 1 shows a semantic network built of such triples.

While using a table of semantic triples to handle small semantic networks is often suffi-

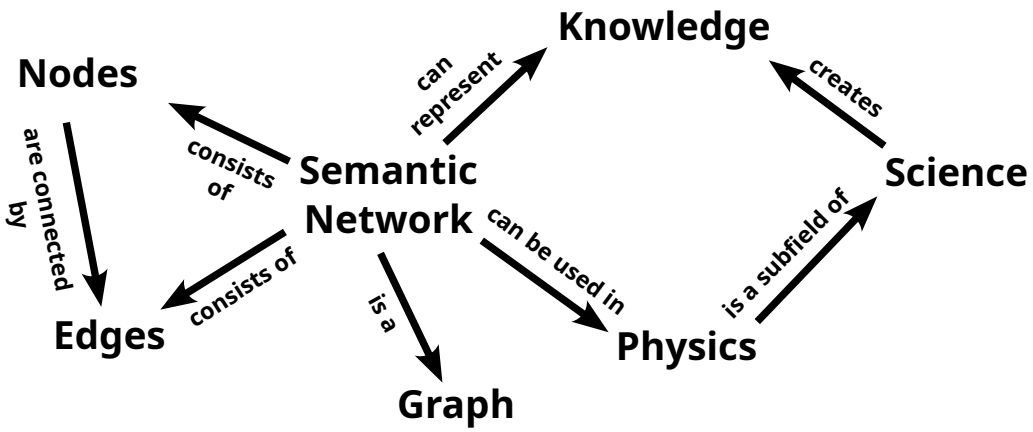



Figure 1: Example for a semantic network build of semantic triples

cient, larger networks usually require a more complex data structure. One common way to store semantic networks is to use a graph database like `Neo4j` [14]. These databases often offer a query language like `Cypher` [15] that allows searching for specific patterns in the network.

An important standard for storing and exchanging semantic network data is the Resource Description Framework (RDF) [16]. RDF is a core technology for the Semantic Web, which is an extension of the World Wide Web for storing data in a machine-readable way that was also coined by Tim Berners-Lee [17]. RDF uses semantic triples to represent data whose elements are addressable by Uniform Resource Identifiers (URIs).

All semantic networks share the property of being able to be used as a knowledge base to store information about various topics. They are, therefore, sometimes referred to as knowledge graphs. This name was initially assigned to semantic networks with a restricted set of connection types but is now also used with a broader meaning [18].

The advantage of using a semantic network as a knowledge base rather than a pure text document is that the different concepts appear only once in the database and are directly linked to all information available about them. This increases the efficiency of searching for specific information, especially when dealing with large amounts of data. For example, if one wants to search for the charge of an electron in an extensive collection of papers, one has to read through all the documents until one finds the information. When the same information is stored in a semantic network, one can simply search for the concept "electron" and see to which value it is connected by the predicate "has charge of." The clear benefit of using semantic networks instead of the trained parameters of a large language model as a knowledge base is that it is possible to isolate the different facts that are contained in the network, check their correctness, and correct them if necessary. Another advantage of using a semantic network to store information is that it is possible to merge different knowledge bases into one without redundancy. Once the identical concepts and relationships are identified, they can be merged and thereby connect the different networks.

## 1.3. Motivation and goal of this thesis

The advantages of using semantic networks to represent knowledge could have great potential to improve scientific research in the future. In a world where the amount of scientific data is growing rapidly, it becomes essential to structure this data so that it can be easily accessed, understood, and discussed by researchers. Therefore, it could eventually become common practice for the scientific community to collectively maintain a large knowledge base containing all the knowledge gained so far. This knowledge base could contain information about the physical theories, the experiments, and the results. It could also contain metadata like reviews and information about the contributing scientists and be used as a platform for scientific discourse. New results could be published directly into this knowledge base and be linked to the existing knowledge. The maintenance of such a knowledge base could be done by a combination of human and artificial intelligence.

While it is not yet clear what technologies could be the foundation for a knowledge base like this, semantic networks are a promising candidate. However, the open question remains about how physics knowledge can be represented as a semantic network. This master's thesis aims to investigate this question and explore different methods for structuring physics knowledge into a semantic network. A particular focus will be placed on using modern machine-learning methods to generate and evaluate the network data. Thereby, it will also be investigated how the internal physics knowledge of large language models can be extracted as a semantic network and how the extracted data can be used to answer physical questions. Finally, it will be explored what possible applications of semantic networks in fundamental physics could be.

# 2. Structuring the different subject areas of physics

This chapter discusses how the different subject areas of physics can be structured. It starts by comparing methods of identifying physics terms. It then explores different methods of structuring these terms into a tree structure of subtopics. In particular, it focuses on two different algorithms. For each of these algorithms, subtopic trees are generated and evaluated. For the evaluation, a metric measuring the efficiency of the tree structure is introduced. Finally, different large language models are compared regarding their ability to structure physics terms into a tree.

## 2.1. Collecting physics terms

The first required step is to create a list of physics terms that can be used to structure the different subject areas of physics. This section discusses the three approaches to collecting physics terms: collecting them from the Oxford Dictionary of Physics, obtaining them from Wikipedia articles, and generating them using a large language model.

### 2.1.1. Using physics terms from the Oxford Dictionary of Physics

Several online resources provide lists of physics terms. For example, the Oxford Dictionary of Physics [19] contains a list of 3767 physics terms. These terms were extracted from the online version of the dictionary [20]. The list contains physics terms from different subject areas. The following excerption contains five randomly selected terms from the Oxford Dictionary of Physics:

- "*induced fission*"
- "*solar energy*"
- "*Biot–Savart law*"
- "*subsonic speed*"
- "*digital display*"

As this example shows, not all terms are completely physics-related. For example, "*digital display*" is not a physics term but a term from computer science. Additionally, the list contains some abbreviations, like "*AU*" or "*LEP*", that might require additional explanation to be clear.

### 2.1.2. Extracting physics terms from Wikipedia articles

Another approach is to extract the physics terms from Wikipedia articles. Within a Wikipedia article, the terms that represent relevant concepts are mainly those marked as links. These terms were considered by the Wikipedia authors as important enough to have their own article. To extract these terms, an algorithm was implemented that collects all the titles of the linked Wikipedia articles of a given Wikipedia article. This algorithm is based on the python wikipediaapi library [21]. In the next step, the titles of those Wikipedia articles that are physics-related have to be identified. This has been done using the large language model `gpt-3.5-turbo` [22] to classify the Wikipedia article titles into physics and non-physics-related articles. The classification was done by using the following query:

**Prompt for the classification of Wikipedia article titles**

```
Is <title of Wikipedia article> a technical term in
physics?  (answer y or n)
```

The resulting list of physics-related Wikipedia article titles was additionally filtered manually, mainly to remove all terms that don't relate to physics itself but to the scientific community or infrastructure, such as "*Max Planck Institute for Solid State Research*" or "*Physical Review Letters*". The whole extraction process was done on the seven Wikipedia articles "*Physics*", "*Astronomy*", "*Classical Mechanics*", "*Condensed Matter Physics*", "*Particle Physics*", "*Quantum Mechanics*" and "*Thermodynamics*". The resulting list of physics terms has 1038 entries. It contains a large variety of physics terms from different subject areas of physics. Here is a small excerpt of five randomly selected terms from the list:

- "*Maxwell relations*"
- "*Carnot's theorem (thermodynamics)*"
- "*Jupiter trojan*"
- "*Optical telescope*"
- "*Mathematical physics*"

Even if this approach leads to a list that does not contain many duplicates, some exceptions of similar terms occur multiple times. For example, the terms "*Black Hole*" and "*Black Holes*" are both on the list. This is because the Wikipedia article titles are not always consistent in their naming conventions, and there is such thing as redirected articles that have their own title.

### 2.1.3. Generating physics terms using a large language model

The question arises of whether there is any more scalable and automatable way of creating a list of physics terms than extracting them from the Oxford Dictionary or Wikipedia

topics. It seems plausible to use large language models for this task. This approach was tested on the `gpt-4-turbo` model [23]. The model was asked to generate a list of physics terms.

The result is terms like "*Condensed Matter Physics*", "*Quantum entanglement*", "*Higgs boson*", "*Hooke's Law*", "*Photonics*", "*Fluid dynamics*", and "*Angular momentum*". It is interesting to note that `gpt-4-turbo` tends to return the same terms when asked repetitively. After 35 Iterations, the term "*Superconductivity*" was returned 30 times, "*Thermodynamics*" 26 times, and "*Black holes*" 21 times. This effect can be mitigated by raising the temperature value of the `gpt-4-turbo` model, which indicates how much the generated results are subject to chance. Figure 2 shows the number of different terms dependent on the number of generated terms for the different temperatures 1, 1.2, and 1.4. In all three

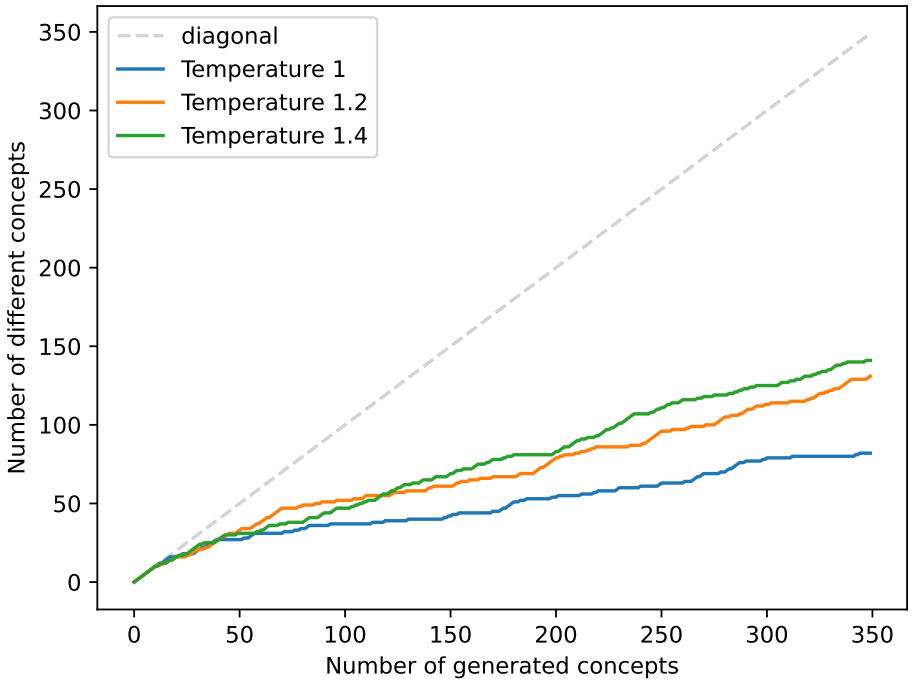



Figure 2: Number of different terms dependent on the number of generated terms for the different temperatures 1, 1.2 and 1.4.

cases, the curve rises depending on the number of generated terms. The efficiency improves for higher temperatures but is still not very high.

An alternative approach to generating new physics terms is to first generate a tree structure of the subtopics of physics, which is called a subtopic tree (see section 2.2 for more details). The model is then asked to generate physics terms that belong to randomly selected branches of this subtopic tree at a given depth. Figure 3 compares the new approach with the best-performing standard approach.

The subtopic tree approach is much more efficient than the standard approach. However, the generated physics terms are much more specific. This applies especially if they

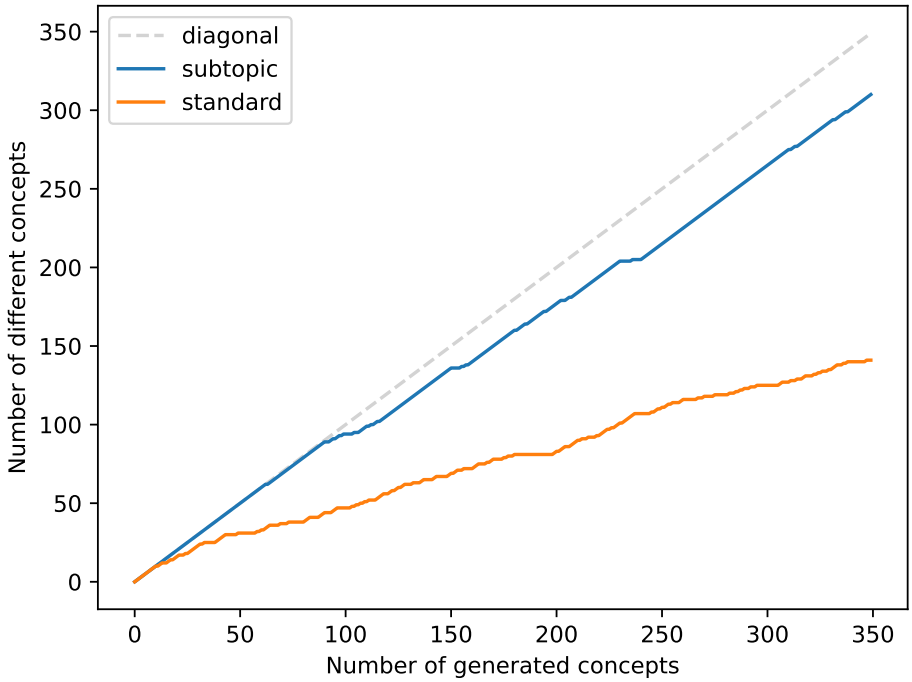



Figure 3: Number of different concepts dependent on the number of generated concepts for the subtopic tree approach and the standard approach.

are selected deeper within the subtopic tree.

Some example terms selected from a depth of 2 in the subtopic tree are "*Atmospheric neutrinos*", "*Kerr metric*", "*Fission*", "*Neutron capture*", "*Magnetic resonance*", "*Hadronization*", "*Multiverse Theory*", and "*Cosmological Redshift*". Here are some example terms selected from a depth of 4 in the subtopic tree: "*Galaxy clustering statistics*", "*Partial Polarization*", "*Solar wind contributions to dust charging*", "*Scaling transformations*", "*Photovoltaic cells*", "*Quantum entanglement in cosmology*" and "*Half-life of Uranium-238*".

## 2.2. Structuring physics terms into a subtopic tree

The term subtopic tree is used here for a tree structure of subtopics that start from a root node. In this case, the root node has the topic "*Physics*". The root node has several child nodes that represent the different subtopics of physics. Each of these child nodes can have further child nodes that represent subtopics of the subtopics. This structure can be continued to an arbitrary depth. Within the tree structure, two types of nodes can be distinguished: The leaf nodes and the junction nodes. The leaf nodes resemble the physics terms that should be structured. They do not have any further child nodes. The junction nodes are the nodes that are used to structure the leaf nodes. They can have further child nodes that are either leaf nodes or junction nodes. We call a junction node a supporting junction node if it has at least one child leaf node as a descendant.

A subtopic tree can be created manually by a human expert or automatically by a machine learning model. This work focuses on the automatic creation because it makes it possible to investigate the ability of large language models to structure and represent physics in an organized manner. To make sure that the created subtopic trees cover a broad range of physics topics, the list of 1038 physics terms extracted from Wikipedia articles that was mentioned in section 2.1.2 was used. The subtopic trees were then generated in a way that they contain all of these terms as leaf nodes. Because of the limited context window of the large language models and the high complexity of the task, it was not possible to generate the subtopic trees using a single prompt. Instead, the subtopic trees were constructed in multiple steps by inserting the physics terms one by one. Two different algorithms were used that can achieve this: the free association algorithm and the subdivision algorithm.

### 2.2.1. Description of the algorithms used to create the subtopic trees

The two algorithms pursue different strategies to structure the physics terms into subtopic trees. The free association algorithm freely associates possible subtopics of a given topic without considering the terms that are already in the tree. The subdivision algorithm, on the other hand, takes the existing terms and topics of the subtopic tree and divides them into more specific subtopic categories. Both algorithms start with a single root node that has the topic "*Physics*" and no further child nodes. In both cases, the terms are inserted into the existing tree structure and trigger the creation of new junction nodes if the number of child nodes of a given junction node exceeds a certain threshold. This threshold is a variable parameter of the algorithms, which is called `maxChildNumber`. The place where the terms are inserted into the tree is determined by a search mechanism that starts at the root node and asks `gpt-3.5-turbo` at each junction node which child junction node should be chosen to insert the term. In the case of the free association algorithm, the search is continued until there are no further child junction nodes. In the case of the subdivision algorithm, `gpt-3.5-turbo` has the additional option to decide that none of the existing child junction nodes is suitable for the term, in which case the term is directly inserted at the last matching junction node.

When the number of child nodes exceeds the `maxChildNumber` at a given junction node, the free association algorithm asks `gpt-3.5-turbo` to generate a total of `maxChildNumber` subtopics for the topic of the junction node which are then inserted as new child junction nodes. The previous child leaf nodes are then distributed among the new junction nodes by the insertion mechanism. These two steps are illustrated in figure 4.

The subdivision algorithm uses a different mechanism to create new junction nodes. When the number of child nodes exceeds the `maxChildNumber` at a given junction node, the subdivision algorithm asks `gpt-4-turbo` to subdivide the existing child nodes into categories. These categories are then inserted between the junction node and the child

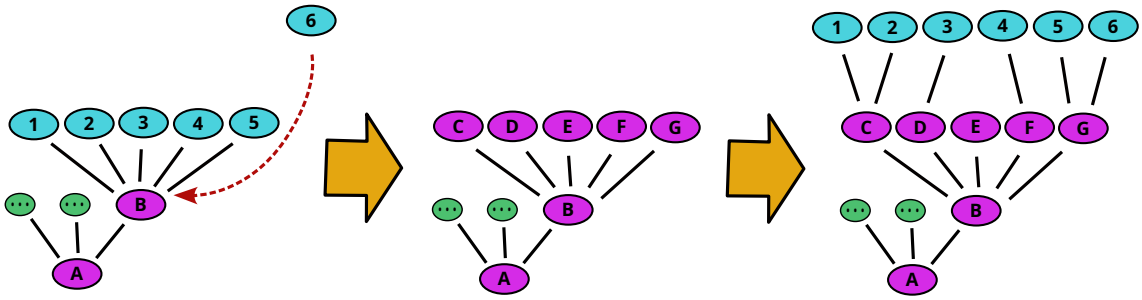



Figure 4: Illustration of the mechanism of the free association algorithm that creates new junction nodes when the maximum number of child leaf nodes is reached. The leaf nodes are colored in turquoise, and the junction nodes are colored in purple. The green node with the triple dots symbolizes an arbitrary number of further child nodes. In this example, the `maxChildNumber` is set to 5. When leaf node 6 is inserted at junction node B, the mechanism is triggered to create the new junction nodes C, D, E, F, and G. The leaf nodes 1 - 6 are then distributed among the new junction nodes.

nodes as a new subdivision layer. This mechanism is illustrated in figure 5. It was necessary to use `gpt-4-turbo` for the complex task of subdividing the child nodes because `gpt-3.5-turbo` was not able to reliably assign all child nodes to a subcategory.

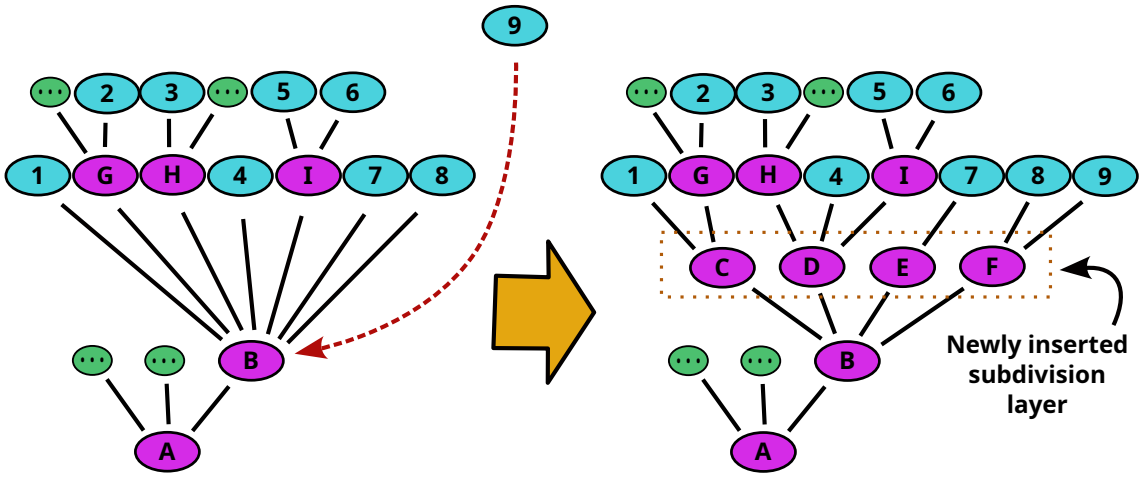


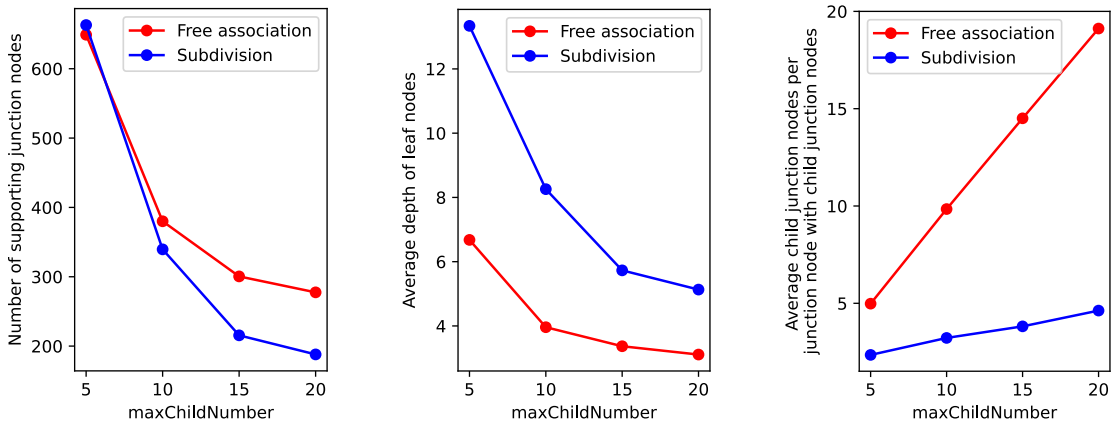

Figure 5: Illustration of the mechanism of the subdivision algorithm that subdivides child nodes into categories when the maximum number of child leaf nodes is reached. In this example, the `maxChildNumber` is set to 7. When the new leaf node 9 is inserted at the junction node B, the mechanism is triggered to subdivide the existing child nodes into the categories C, D, E, and F.

The creation process of a subtopic tree created by the free association algorithm differs from the one created by the subdivision algorithm. While a tree created by the former algorithm grows only at the tips of its branches, a tree created by the latter algorithm can grow at any point in the tree structure. Besides that, all leaf nodes of a tree created by the free association algorithm are growing on junction nodes with no further child junction nodes. This is not the case for a tree created by the subdivision algorithm.

### 2.2.2. Creating subtopic trees using the free association algorithm and the subdivision algorithm

Because the creation of subtopic trees is computationally expensive, only a limited number of subtopic trees could be created in this work. To enable a good comparison of the two algorithms with different parameter settings, a total of 16 subtopic trees were generated. Eight of them were created using the free association algorithm, and eight of them were created using the subdivision algorithm. For both algorithms, subtopic trees were generated with `maxChildNumber` set to 5, 10, 15, and 20. For each setting of an algorithm with a specific `maxChildNumber`, two subtopic trees were generated to test how much the properties of the subtopic trees depend on the random initialization of the large language models.

The subtopic trees contain the 1038 physics terms extracted from Wikipedia articles as leaf nodes. Figure 6 shows the properties of the subtopic trees created by the free association algorithm and the subdivision algorithm in dependence on the `maxChildNumber`.




(a) (b) (c)

Figure 6: Different properties of the subtopic trees created by the free association algorithm and the subdivision algorithm in dependence on the `maxChildNumber`. The three subplots show the number of supporting junction nodes (a), the average depth of the leaf nodes (b), and the average number of child junction nodes per junction node with child junction nodes (c). The plotted properties are averaged over the two subtopic trees created with the same parameter settings. The error bars are not shown because the low number of data points does not allow for reliable error statistics for the different values of the `maxChildNumber` parameter.

Subplot (a) shows that both algorithms have a similar behavior of creating new junction nodes. The number of supporting junction nodes increases in both cases with a lower `maxChildNumber` because the mechanism to create new junction nodes is triggered more often. For an `maxChildNumber` of 5, the free association algorithm and the subdivision algorithm create both approximately 650 supporting junction nodes. For an

`maxChildNumber` of 20, the free association algorithm creates approximately 50% more supporting junction nodes than the subdivision algorithm.

The average depth of the leaf nodes also increases with a lower `maxChildNumber` for both algorithms, as shown in subplot (b). This is expected because a low number of child nodes per junction node leads to a deeper tree structure. It is interesting to note that the subdivision algorithm creates trees that are approximately 2 to 3 levels deeper than the trees created by the free association algorithm. The main reason for this is that the structure of junction nodes created by the subdivision algorithm is slimmer than the structure of junction nodes created by the free association algorithm and, therefore, requires more depth to fit all leaf nodes. This can be seen in subplot (c), which shows the average number of child junction nodes for all junction nodes that have child junction nodes. For the free association algorithm, this number is approximately equal to the value of `maxChildNumber`. This is because in the free association algorithm `gpt-3.5-turbo` was asked to generate a total of `maxChildNumber` subtopics for the topic of each junction node. In the case of the subdivision algorithm, the average number of child junction nodes per junction node with child junction nodes also rises linearly with `maxChildNumber` but with a lower slope of 0.15. This is due to the fact that the subdivision algorithm splits the child nodes that exceed the `maxChildNumber` into a number of subcategories that is always smaller than the `maxChildNumber`.

Another difference between the two algorithms is that the free association algorithm creates subtopic trees whose branch lengths are more evenly distributed than those created by the subdivision algorithm. This can be seen in figure 7 and figure 8 that compare the distribution of the leaf nodes among different depth levels for both algorithms with a `maxChildNumber` of 10. As one can see, the subdivision algorithm produces a

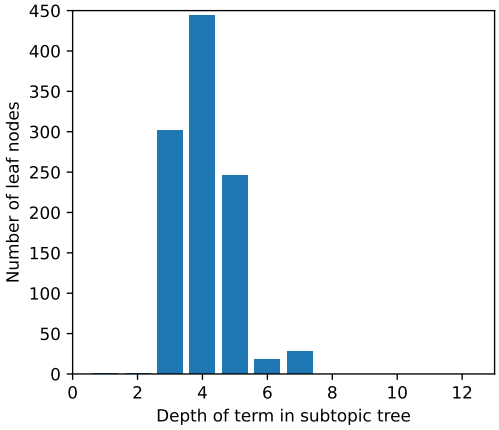



Figure 7: Distribution of the leaf nodes among the different depth levels of a subtopic tree created by the free association algorithm with `maxChildNumber` set to 10.

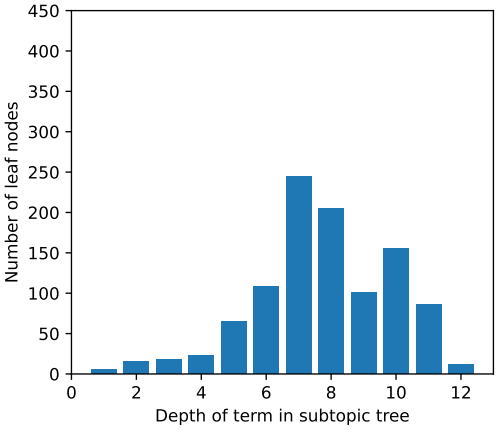



Figure 8: Distribution of the leaf nodes among the different depth levels of a subtopic tree created by the subdivision algorithm with `maxChildNumber` set to 10.

widespread distribution of the leaf nodes among different depth levels that starts at 1 and raises up to 12 with a maximum at 9. The free association algorithm, on the other hand, produces a narrow peak of leaf nodes with a depth level of 3 to 5 with some outliers at 6 and 7. The free association algorithm also shows higher stability of the average leaf node depth for different trees created with the same parameter settings. It varies only by 0.04 levels on average, while the subdivision algorithm shows a variation of the average leaf node depth by 0.9 levels on average. This is due to the less predictable behavior of the subdivision algorithm that can shift whole branches one level deeper in the tree structure when the number of child nodes exceeds the `maxChildNumber`.

### 2.2.3. Evaluating the content of the subtopic trees

Two subtopic trees that are typical representatives of the subtopic trees created by the free association algorithm and the subdivision algorithm were selected for a detailed evaluation of their content: The subtopic tree `frAss10`, which was created by the free association algorithm with a `maxChildNumber` of 10, and the subtopic tree `subd15`, which was created by the subdivision algorithm with a `maxChildNumber` of 15. Both subtopic trees choose different categories to structure the physics terms. The tree `frAss10` is visualized in figure 9 and the tree `subd15` is visualized in figure 10. The figures show the structure of the subtopic tree, which consists of small green lines linking the green junction nodes and the blue leaf nodes. The root node "*Physics*", in the figure's center, is marked with a black arrow. Groups of nodes that form a subject area are circled with colored lines and labeled with the name of the subject area in the same color. Junction nodes higher in the tree hierarchy are labeled in black with their topic.

The root node of the subtopic tree `frAss10` has ten subtopics. Astrophysics (183 leaf nodes), particle physics (196 leaf nodes), and quantum mechanics (165 leaf nodes) are the most prominent ones. The least prominent subtopics are electromagnetism (47 leaf nodes), nuclear physics (45 leaf nodes), and fluid mechanics (27 leaf nodes). Most leaf nodes are filed under an appropriate subtopic path, such as "*Schwarzschild radius*" under the path "*Astrophysics*" -> "*Black holes*" -> "*Event horizon*". Some others are filed under an inappropriate subtopic path, such as "*Vacuum*" under the path "*Electromagnetism*" -> "*Electromagnetic spectrum*". There are a few junction nodes with names that can not be understood without the context, such as the nodes "*Definition*" and "*Applications*", which are registered as subtopics of the topic "*Quantum entanglement*". All in all, the subtopic tree `frAss10` is a compact and well-structured representation of the physics terms.

The root node of the subtopic tree `subd15` has five junction nodes as subtopics. These are "*Quantum Mechanics and Atomic Physics*" (419 leaf nodes), "*Astrophysics and Cosmology*" (252 leaf nodes), "*Fundamental Physics*" (142 leaf nodes), "*Statistical Physics and Thermodynamics*" (142 leaf nodes), and "*Applied Physics and Interdisciplinary Physics*" (74 leaf nodes). In contrast to the subtopic tree `frAss10`, several leaf nodes are attached directly to the root node. The categories "*fundamental physics*" and "*applied physics and interdisciplinary*

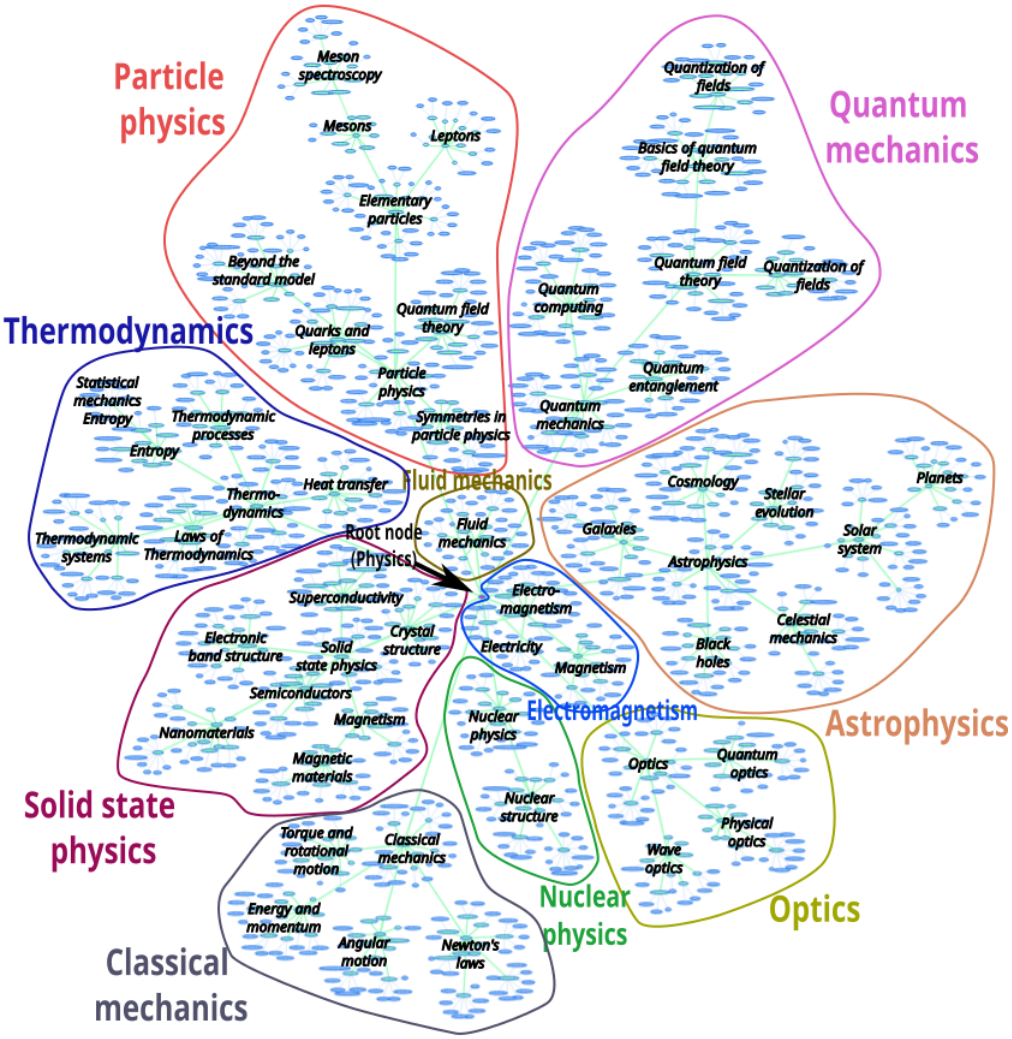



Figure 9: Labeled visualization of the subtopic tree `frAss10`.

*physics*" are not very well suited to structure the physics terms because they overlap with other categories and are not very specific. The subtopic tree also contains some wrongly assigned junction nodes, such as "*Classical mechanics*" and "*Electromagnetism and Phenomena*", which are registered as subtopics of the topic "*Quantum mechanics and atomic physics*". Furthermore, topics like "*Quantum mechanics*" and "*Particle physics*" appear in multiple places in the tree. Another problem is that the tree `subd15` contains some junction nodes with a single subbranch. This is not very efficient because the junction node does not add any structure to the tree. These cases are marked in figure 10 with yellow arrows. All in all, the subtopic tree `subd15` is a less compact and less well-structured representation of the physics terms than the subtopic tree `frAss10`.

### 2.2.4. A metric to evaluate the quality of the subtopic trees

To evaluate the quality of the subtopic trees, a metric measuring the efficiency of the tree structure is introduced. This is done by measuring how well terms can be found in the

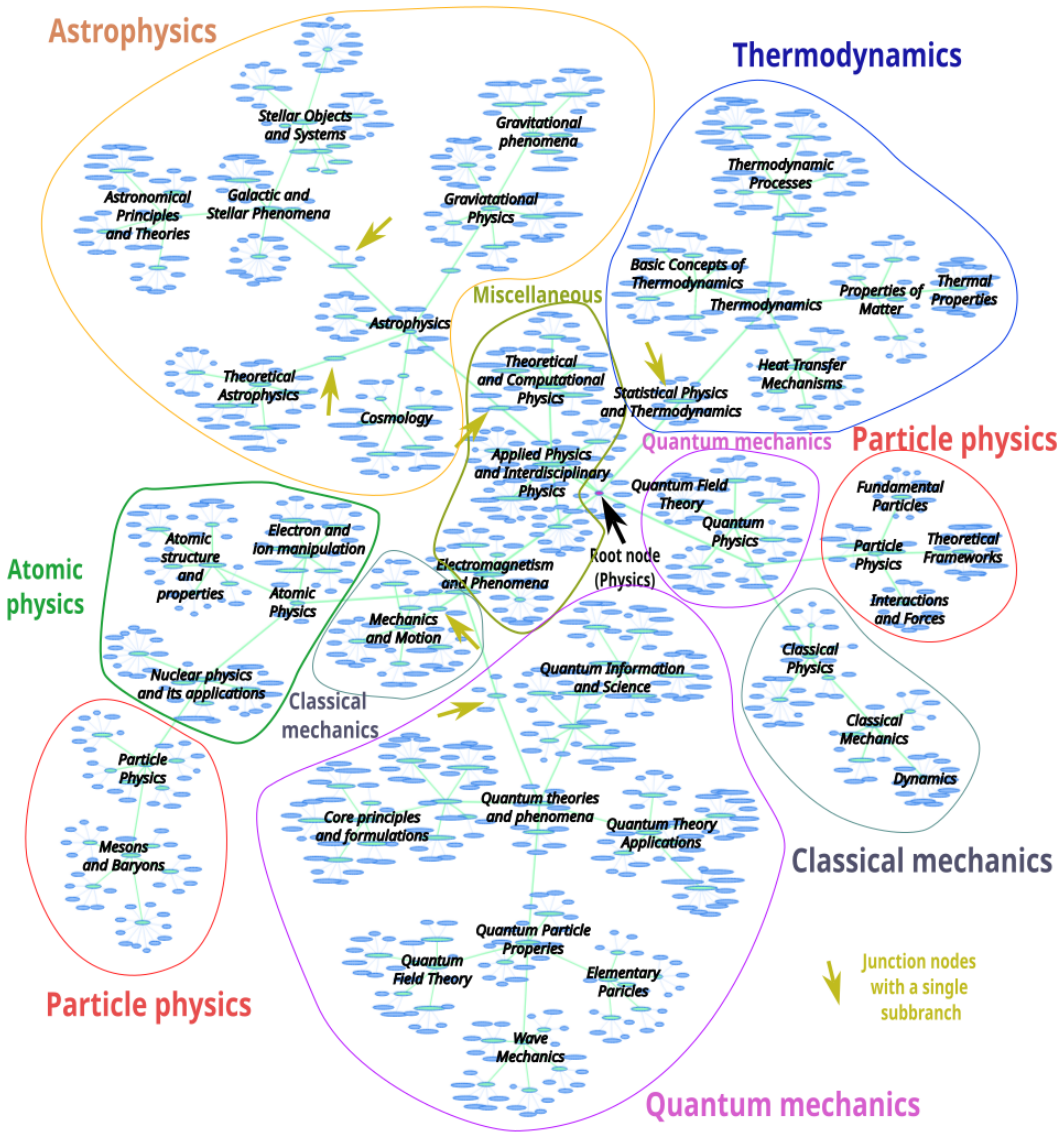



Figure 10: Labeled visualization of the subtopic tree `subd15`.

tree. For this, the assumption is made that the language model searching the subtopic tree for a desired term starts at the root note and looks at all available branches. It then chooses one based on the term it is searching for and repeats the process until it reaches the last node. If the search is terminated without success, it starts again at the root node without remembering its previous choices. This is repeated until the desired term is found. The quality of the subtopic tree is measured by counting the total number of terms the evaluating model has to look at during this process. If this number is lower than the total number of leaf terms in the subtopic tree, the classification is an improvement compared to a plain list of all available leaf terms. The average number of viewed terms $T_{view}$ is calculated in the following way:

$$T_{view} = (N_{rep} - 1)T_{fail} + T_{suc} = (\frac{1}{P_{suc}} - 1)T_{fail} + T_{suc}$$

Where $T_{suc}$ and $T_{fail}$ are the average number of viewed terms in a successful and failed search run, and $N_{rep}$ is the average number of search run repetitions that is necessary to find the desired term. $N_{rep}$ equals the inverse of $P_{suc}$, which is the probability that a search run is successful. For the calculation, the assumption is used that all terms are equally hard to find, which is a simplified model.

A modified version of this metric that penalizes longer term names is the total viewed bytes $B_{view}$, which is calculated analogously but with the number of bytes of the terms instead of the number of terms. This version of the metric is more oriented toward the time a person would need to find a term in the tree. It considers that a detailed description of a subtopic category takes more time to read than a short term name.

A good impression of the quality of a subtopic tree can be obtained by calculating the viewed terms fraction, which is the percentage ratio between $T_{view}$ and the total number of leaf terms $T_{leaf}$. Analogously, the viewed bytes fraction can be calculated by dividing $B_{view}$ by the total number of bytes of the leaf terms $B_{leaf}$. If these numbers are close to zero, the classification is efficient. If the numbers are close to 100% or higher, the classification is inefficient.

### 2.2.5. Using the metric to evaluate the subtopic trees

This metric was used to evaluate the 16 subtopic trees that were generated in the previous step. The results of the evaluation are shown in figure 11. The figure shows that all generated subtopic trees have a viewed bytes fraction that is smaller than 30% wich means that they are suitable for organizing the physics terms in a way that they are easier to find than in a plain list. The trees created by the free association algorithm have a lower viewed bytes fraction and are, therefore, more efficient in classifying the physics terms than the trees created by the subdivision algorithm. The subtopic tree with the highest efficiency was created by the free association algorithm with a `maxChildNumber` of 10. The subtopic trees with the lowest efficiency are created by the subdivision algorithm with a `maxChildNumber` of 5 and 10. The variation of the viewed bytes fraction within the pairs of subtopic trees that were created with the same algorithm and `maxChildNumber` goes beyond the error of the viewed bytes fraction of the individual subtopic trees. This indicates that the quality of the subtopic trees is not only dependent on the algorithm and the `maxChildNumber` but also depends on the random initialization of the large language models. The average depth of the leaf nodes that rises with a lower `maxChildNumber` shows a slight positive correlation with the viewed bytes fraction. This is reasonable because it requires more correct navigational choices to find a term that is hidden deeper in the tree structure. This effect can also be seen in table 1, which lists the properties of different subtopic trees. The subtopic trees `subd5`, `subd15`, and `subd20` have a slightly better average choice correctness percentage than the subtopic trees `frAss10` and `frAss15`. However, because their average term depth is much higher, they perform worse in the average viewed terms fraction and the average

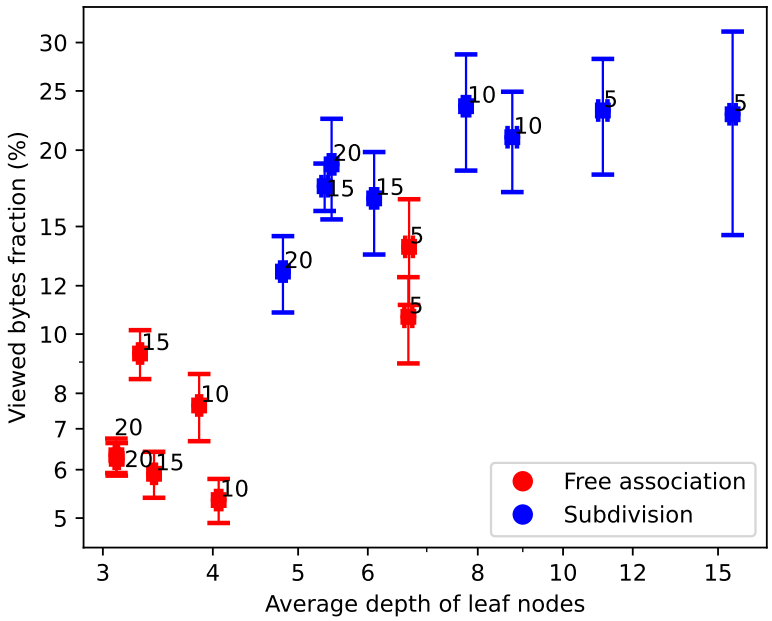



Figure 11: The viewed bytes fraction in dependence of the average depth of the leaf nodes for the subtopic trees created by the free association algorithm and the subdivision algorithm. The data points are labeled with the `maxChildNumber` that was used to create the subtopic trees. The axes are scaled logarithmically.

viewed bytes fraction.

The table also shows that the average viewed terms fraction and the average viewed bytes fraction are very similar for all subtopic trees. This means that the generated topic names of the junction nodes that are used to organize the physics terms have a similar length as the physics terms themselves.

### 2.2.6. Comparison of the ability of different models to create subtopic trees of physics

Next, it is investigated how well different large language models perform in creating subtopic trees of physics. For this the models `gpt-3.5-turbo`, `gpt-4-turbo`, `Mistral-7B-Instruct-v0.2` [24], `Cosmosage-V2` [9] and `Llama3.1-8B-Instruct` [25] are compared. These models are based on the transformer architecture. `gpt-3.5-turbo` and `gpt-4-turbo` are closed-source models from OpenAI, while the others are open-source models. `cosmosage` is a finetuned version of the `mistral` model that specializes in cosmology.

For comparison, the subtopic trees generated by the different models will be searched for a set of 50 physics terms. This dataset is a hand-selected subset of the 1038 physics terms used to create the 16 subtopic trees in the previous evaluation. The subset contains well-known physics terms like "*Atom*" and "*Quantum mechanics*" as well as more

| **Tree** | $T_{depth}$ | $T_{view}/T_{leaf}$ | $B_{view}/B_{leaf}$ | $choi_{corr}$ |
|---|---|---|---|---|
| frAss5 | $7.684 \pm 0.0668$ | $13.05 \pm 2.55\%$ | $13.89 \pm 2.73\%$ | $82.99 \pm 1.79\%$ |
| frAss10 | $4.8593 \pm 0.02$ | $7.697 \pm 0.962\%$ | $7.642 \pm 0.962\%$ | $76.62 \pm 2.99\%$ |
| frAss15 | $4.3064 \pm 0.0176$ | $10.189 \pm 0.929\%$ | $9.295 \pm 0.855\%$ | $74.36 \pm 2.47\%$ |
| frAss20 | $4.1108 \pm 0.0172$ | $6.425 \pm 0.388\%$ | $6.25 \pm 0.383\%$ | $88.94 \pm 2.13\%$ |
| subd5 | $12.097 \pm 0.135$ | $20.52 \pm 4.36\%$ | $23.22 \pm 4.98\%$ | $79.01 \pm 1.43\%$ |
| subd10 | $9.756 \pm 0.108$ | $18.49 \pm 3.46\%$ | $21 \pm 3.92\%$ | $63.33 \pm 2.23\%$ |
| subd15 | $7.1002 \pm 0.0489$ | $13.59 \pm 2.59\%$ | $16.68 \pm 3.18\%$ | $77.74 \pm 2.3\%$ |
| subd20 | $5.8054 \pm 0.0353$ | $12.4 \pm 1.74\%$ | $12.66 \pm 1.81\%$ | $80.38 \pm 2.23\%$ |

Table 1: Evaluation of different subtopic trees. The following properties are shown: The average term depth $T_{depth}$, the average viewed terms fraction $T_{view}/T_{leaf}$, the average viewed bytes fraction 6 and the average correctness percentage of the choices that were offered to the model during the search process $choi_{corr}$.

specialized terms like "*Casimir effect*" and "*Gauge symmetry*". In contrast to the previous evaluation, not the whole subtopic tree is generated but only the parts of the tree that are necessary to find the desired terms. The following algorithm to search for a term in the subtopic tree is used: It starts at the topic "*Physics*" and, from there on, performs a maximum of 10 search iterations. In each iteration, the tested model is asked to generate a list of subtopics that together cover the entire range of the current topic. The number of subtopics generated in each iteration is not restricted, and the model freely associates the subtopics without the information of the term that is searched for. While the generation of the subtopic trees is performed by the different tested models, the evaluation is performed by the model `gpt-3.5-turbo` to ensure consistency. Therefore, `gpt-3.5-turbo` is used to choose the subtopic that is most likely to contain the term that is searched for. It is then tested if the term that is searched for is contained in the selected subtopic string. If this is the case, the term has been found, and the search process is terminated. If this is not the case, the search will continue with the next iteration. If the term is not found after ten iterations, the search process is terminated and the term is marked as not found. This search is performed twice for all 50 terms for all models to also get an impression of their consistency.

As it turns out, the smaller open source models `mistral`, `llama`, and `cosmosage` are not able to return their generated subtopics in a machine-readable format. They alter the format or add a text description to the list of subtopics even if they are asked to return nothing but the list. For this reason, an additional step is necessary to extract the subtopics from the generated text. Therefore, the model `gpt-3.5-turbo` filters the generated text and brings it into the desired format. The results of the comparison are shown in table 2.

As one can see, the model `gpt-4-turbo` has the best performance regarding the percentage of terms found. The models `mistral_7b_instruct` and `llama_3_1_8B` have a similar performance that is slightly better than the performance of the model `gpt-3.5-turbo`. This is remarkable because both models could not generate the subtopics in a

| Model | Found | Iterations | Subtopics | Stability |
|---|---|---|---|---|
| gpt-4-turbo | 71±4.5% | 3.30±0.19 | 10.37±0.17 | 86±4.9% |
| mistral_7b_instruct | 63±4.8% | 3.24±0.23 | 9.34±0.14 | 78±5.9% |
| llama_3_1_8B | 62±4.9% | 3.42±0.27 | 9.49±0.19 | 68±6.6% |
| gpt-3.5-turbo | 56±5.0% | 3.25±0.25 | 6.68±0.11 | 80±5.7% |
| cosmosage | 48±5.0% | 3.92±0.35 | 14.97±0.52 | 64±6.7% |

Table 2: Comparison of the ability of different models to create subtopic trees of physics. The first column, "*Found*", shows the percentage of terms the model found. The second column, "*Iterations*", shows the average number of iterations that were necessary to find a term. The third column, "*Subtopics*", shows the average number of subtopics that were generated in each iteration. The fourth column, "*Stability*", shows the percentage of terms found in both search runs or none of the search runs.

machine-readable format and needed the help of the model `gpt-3.5-turbo` to filter the generated text. The model `cosmosage` has the worst performance regarding the percentage of terms found. The average number of iterations necessary to find a term is similar for all models. The number of subtopics, on the other hand, varies between the models. Most subtopics are generated by the model `cosmosage`, which produces more than twice as many subtopics per iteration than the model `gpt-3.5-turbo`, which generates the least subtopics. The stability of the models is highest for the model `gpt-4-turbo` and lowest for the model `cosmosage`.

# 3. Representing physical knowledge as semantic triples

This chapter explores how semantic triples can be used to represent physical knowledge. First, a network of correlated physics terms is created that is later transformed into a network of semantic triples by generating predicates for the connections. Next, a metric is introduced that measures the meaningfulness of the triples. Then, the triples are manually evaluated to determine their validity, truth, context dependency, and triviality. Finally, a method is introduced to visualize the network of semantic triples.

## 3.1. Creating a correlation network

A network of correlations was created by connecting closely related pairs from the list of 1038 physics terms described in section 2.1.2. The subtopic trees from the previous chapter were used to determine the relatedness of the terms. Two terms were considered related if they were connected to the same parent node in one of the subtopic trees. In this way a network of correlations was built from six of the subtopic trees that were created in the previous step. The graph has 10061 edges, an average of 19.39 connections per term, 39 connections for the most connected term, and 3 connections for the least connected term. It forms one connected unit with an average path length of 3.18 between two randomly selected terms.

To reduce the number of connections and create a more uniform network this graph was modified so that each term has as few connections as possible but still more than or equal to five. Therefore, a pool of potentially connected terms was created for each term that was constructed by extending the set of initially connected terms by randomly selected terms, if necessary, to reach a total of 39 terms. In the next step, `gpt-3.5-turbo` was asked to select the five strongest connections per term from the pool. Because `gpt-3.5-turbo` failed to provide structured output in a few cases, a small number of connections had to be manually added to the graph to ensure the minimum degree of five.

The resulting graph has 3247 edges, an average of 6.26 connections per term, 15 connections for the most connected term, and 5 connections for the least connected term. It forms one connected unit with an average path length between two randomly selected terms of 4.94. The five most connected terms within this graph are "*Magnetic fields*", "*Quark*", "*Ferromagnetism*", "*Internal energy*", and "*Volume (thermodynamics)*". A random

selection of three terms with their connections is shown in table 3.

| Term | Connections |
|---|---|
| Geophysics | "*Earth's magnetic field*", "*Outgassing*", "*Magnetosphere*", "*Van Allen radiation belt*", "*Solar wind*", "*Physical oceanography*", "*Proca action*" |
| Amorphous solid | "*Crystal, Crystallography*", "*Solid-state physics*", "*Lattice model (physics)*", "*Nucleation, Phase (matter)*", "*Ductility*" |
| Gamma-ray astronomy | "*Fermi Gamma-ray Space Telescope*", "*Compton Gamma Ray Observatory*", "*Gamma ray astronomy*", "*Gamma ray burst*", "*Synchrotron emission*", "*Supernova*" |

Table 3: Random selection of three terms from the correlation network with their connections.

Most of the connections in the correlation network, except some outliers, such as the connection between "*Geophysics*" and "*Proca action*", are reasonable and can be considered content-related.

## 3.2. Generating predicates for the semantic triples

Next, a semantic network was generated that has semantic triples for each connection in the correlation network. These triples have the connected terms as subject and object and a generated expression that describes the relationship between them as a predicate. For each pair of terms, there are two ways to order subject and object, which results in two triples. The `gpt-3.5-turbo` model was used with the following prompt to generate the predicates:

**Prompt for generating predicates**

```
Semantic triples such as ["Star", "emits", "Light"] and
["Rocket", "can bring cargo to", "Space"] consists of a sub-
ject, a predicate, and an object.  What is the predicate for
the triple [ <subj> , ???  , <obj> ]?  Return only the predi-
cate quoted by "" without explanation.
```

The names of the <subj> and <obj> placeholders are replaced by the two terms that are connected by the correlation network. The generated semantic network consists of 6494 triples. Table 4 shows a random selection of 10 triples from this network. Of these ten examples, the triple that states that Bose gas is equivalent to Fermi gas is clearly false.

The triple $\langle$ *curvaton* | *drives* | *cosmic inflation* $\rangle$ is not generally true. Cosmic inflation is a theory that assumes the rapid expansion of space in the early universe. One candidate for the driving mechanism of cosmic inflation is the inflaton field [26]. However, it is not yet known if the inflation is driven by more than one field [27]. Some theories suggest the existence of a curvaton field that played a role in the early universe [28]. Therefore, it

| Subject | Predicate | Object |
|---|---|---|
| Impulse (physics) | is related to | Elasticity (physics) |
| Curvaton | drives | Cosmic inflation |
| Spectroscope | uses | Pinhole camera |
| Quantum nonlocality | relates to | Quantum dynamics |
| General relativity | describes | Spacetime topology |
| Bose gas | is equivalent to | Fermi gas |
| Chern class | is related to | Gauge symmetry |
| Circumstellar envelope | is located at | Frost line (astrophysics) |
| Cosmological constant | relates to | Vacuum energy |
| Gravitation | leads to | Gravitational singularity |

Table 4: Random selection of 10 triples from the semantic network.

can not be ruled out that a theory exists where the curvaton drives the cosmic inflation, which means that this triple may be true in some contexts.

The triple ⟨ *spectroscope* | *uses* | *pinhole camera* ⟩ is also not generally true because most spectroscope designs do not use a pinhole camera. However, it also can not be considered false because there are spectroscopes that use a pinhole as part of their design [29]. The triples ⟨ *impulse* | *is related to* | *elasticity* ⟩, ⟨ *quantum nonlocality* | *relates to* | *quantum dynamics* ⟩ and ⟨ *chern class* | *is related to* | *gauge symmetry* ⟩ are very unspecific but can not be considered as false.

The triple ⟨ *circumstellar envelope* | *is located at* | *frost line* ⟩ is also not false, even though the information about the location is too specific. The circumstellar envelope of a star is a region of gas and dust that surrounds the star, starting near its surface and extending outward, often blending into the interstellar medium. The frost line, on the other hand, is the distance from the central star where the temperature is low enough for volatile compounds such as water, ammonia, and methane to condense into solid ice grains. In our solar system, the frost line of water is located at a distance of a little less than 5 AU from the sun [30]. This is within the region where the circumstellar envelope can be found. The triples ⟨ *general relativity* | *describes* | *spacetime topology* ⟩, ⟨ *cosmological constant* | *relates to* | *vacuum energy* ⟩ and ⟨ *gravitation* | *leads to* | *gravitational singularity* ⟩ can be considered as true and contain meaningful information.

The most used predicates are plotted in figure 12. The list is dominated by the predicates "*is related to*" and "*relates to*", which together make up approximately 10% of all predicates. Together with predicates like "*describes*", "*studies*", and "*belongs to*", they form a group of relatively unspecific predicates. Predicates like "*is a*", "*contains*", and "*consists of*", on the other hand, are more specific but make up a smaller part of the network.

## 3.3. A metric to measure the meaningfulness of the triples

The frequency of nonspecific triples suggests that many generated triples are not particularly meaningful. To test this hypothesis, a metric that measures a triple's meaningfulness

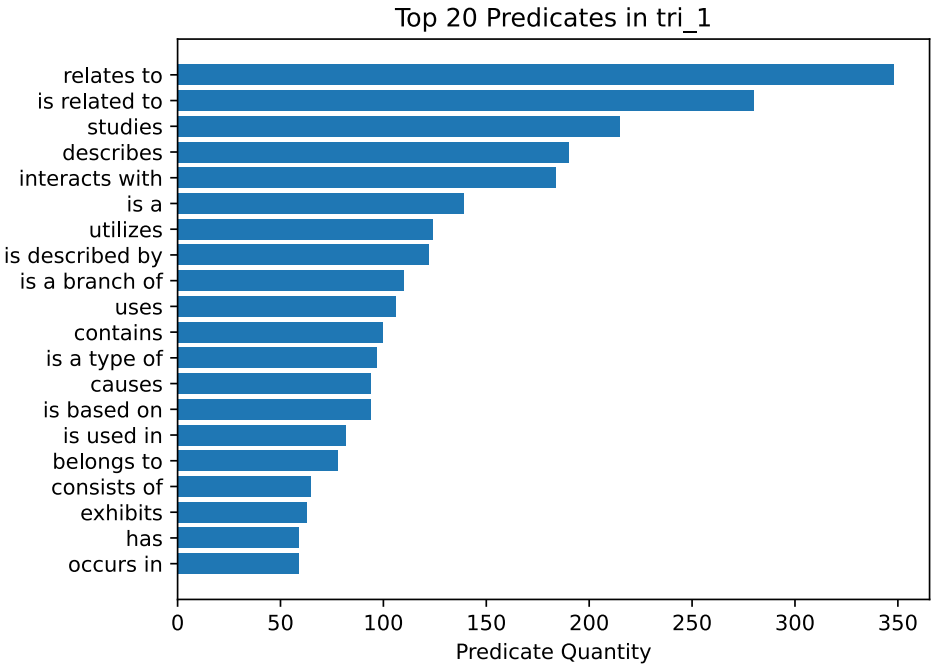



Figure 12: Most used predicates in the semantic network.

is introduced. A meaningful statement should connect two terms so that the predicate describes a relationship specific to the subject and object and not to most other terms. The triple is considered to be not meaningful if, for a given subject and predicate, the choice of the object is arbitrary and true in any case. To measure this, `gpt-3.5-turbo` was asked to identify the correct object for a given subject and predicate from a list of five terms that are connected to the subject in the correlation network. If `gpt-3.5-turbo` performs notably better than random guessing, the triples are considered to be meaningful. The metric for the ambiguity of the objects $A_{\text{obj}}$ is therefore defined as the ratio of the number of false answers to the number of true answers $\frac{\text{FA}}{\text{TA}}$ divided by the ratio of the number of false answers to the number of true answers that would be expected for random guessing $\frac{\text{FA}_{\text{rand}}}{\text{TA}_{\text{rand}}}$:

$$A_{\text{obj}} = \frac{\frac{\text{FA}}{\text{TA}}}{\frac{\text{FA}_{\text{rand}}}{\text{TA}_{\text{rand}}}} \tag{3.1}$$

A high value of $A_{\text{obj}}$ close to 1 indicates that the triples are not meaningful, while a low value close to 0 indicates that the triples are meaningful.

This metric is not perfect because it is also possible that two meaningful triples share the same subject and predicate. For example, the triples $\langle$ *electron* | *is attracted by* | *proton* $\rangle$ and $\langle$ *electron* | *is attracted by* | *positron* $\rangle$ are both meaningful. Still, the model would have problems deciding which object belongs to which of the two identical subject and predicate pairs. However, this can only have the effect of shifting $A_{\text{obj}}$ towards 1. Therefore, a low value of $A_{\text{obj}}$ is still a good indicator for meaningful triples.

For the calculation of the metric, a total of 500 randomly selected triples were checked for the correct object. The calculated value of $A_{\text{obj}}$ for the semantic network is $0.630 \pm 0.063$. Therefore, the identification of the objects works clearly better than random guessing, but the meaningfulness of the triples is still not very high. One reason for this could be that the model struggles to come up with very specific semantic predicates for some pairs of terms because this requires a lot of creativity, and it is easier to choose a more general predicate. For example, `gpt-3.5-turbo` connects the terms "*Electrical conduction*" and "*Charge (Physics)*" with the predicate "*is related to*" even though a more specific predicate like "*is caused by the transport of*" would be more appropriate. Another example is the connection between "*Frequency*" and "*Ultrasonics*" with the predicate "*relates to*" even though it would be possible to use a more specific predicate like "*is greater than 20 kHz for sound waves of*".

Next, it should be investigated if the meaningfulness is higher when the model has more freedom to freely associate the triple by choosing the predicate and the object. These triples are called two-thirds generated triples in contrast to the one-third generated triples where only the predicate is generated. To test if the two-thirds generated triples are more meaningful than the one-third generated triples, the $A_{\text{obj}}$ metric was calculated for 500 two-thirds generated triples that were created by `gpt-3.5-turbo`. The resulting value of $A_{\text{obj}}$ is $0.429 \pm 0.040$. This is a significant improvement compared to the previous value of $A_{\text{obj}}$. The reason for this could be that the model has more freedom to choose the triple and can therefore come up with more specific and meaningful triples such as $\langle$ *Solar corona* | *emits* | *Solar wind* $\rangle$ and $\langle$ *Cosmic inflation* | *is a theory that describes* | *the rapid expansion of space in the early universe* $\rangle$. Of the 500 two-thirds generated triples, the most used predicates are "involves" with 15 occurrences, "is" with 11 occurrences, and "describes" with 9 occurrences.

## 3.4. Manual evaluation of the semantic triples

A manual evaluation was carried out to rate the truth of the physics-related semantic triples generated by the models `gpt-4-turbo` and `gpt-3.5-turbo`. Therefore, a total number of 500 triples were assessed manually. Half of these triples were generated by the models `gpt-4-turbo` and the other half by the models `gpt-3.5-turbo`. Both models created 125 one-third generated triples (with generated predicates) and 125 two-thirds generated triples (with generated predicates and objects).

### 3.4.1. Evaluation criteria

For the evaluation process, four different aspects of the triples were investigated.

**Validness:** Do the three terms fulfill all criteria of being a valid semantic triple? These criteria are:

- The subject and object have to be concepts that contain a noun. For example, "*Fast particle*" is a valid concept while "*fast*" by itself is not a valid concept.
- Subject and object have to stand by themself. For example, the term "*Extra-solar planet*" is a valid concept while "*Planet of that star*" is not valid because it contains a reference to a star mentioned in the context.
- The predicate has to be a text that connects two concepts by creating an English sentence when placed between them. It must not be dependent on a specific predicate and object.

**Truth:** Is the statement of the semantic triple true? This can only be checked if the semantic triple is valid. The truth value is not always clearly defined since some statements depend on the context. For example, the triple $\langle$ *Electron* | *is accelerated by* | *Particle accelerator* $\rangle$ is neither clearly true nor clearly false because not all electrons are accelerated by a particle accelerator. To evaluate whether this statement is true, it must first be determined which electron is precisely meant. Also, some statements can not be determined as true or false because they are the subject of the current debate in science, and multiple theories exist about them. Therefore, a statement can also be true in the context of one theory but false in the context of another theory. For the evaluation, each triple got one of the tree values `true`, `false`, or `unknown` assigned to it. The category `true` is for all triples that are true in some contexts. Only if there is no commonly used context where the triple is true it gets assigned to the category `false`. The category `unknown` is for all those triples where the human evaluator does not know the truth value of the triple.

**Context dependency:** How context-dependent is the statement of the semantic triple? This can only be evaluated if the semantic triple is valid and true. A context dependency score value was assigned to each triple in the range from 0 to 10. The context dependency determines how well-defined the truth value of the triple is. Only the true triples with a context dependency score of 0 can be considered true in any scenario. A true triple with a context dependency score of 10 is only true in very specific contexts. If the context dependency score of a true triple is 5, it is approximately true in 50% of the commonly used contexts.

**Triviality:** How trivial is the statement of the semantic triple? This can only be evaluated if the semantic triple is valid. The triviality score assigned to every triple ranges from 0 to 10. A low triviality score means that the evaluation of the triple requires expertise in the respective field. In contrast, a high triviality score means that the truth value of the triple is obvious when considering general knowledge. In particular, predicates like "*Is related to*" lead to a high triviality score because they are very general and apply to most pairs or subjects and objects.

### 3.4.2. Results of the manual evaluation

Of the 500 evaluated triples, 461 were classified as valid triples. Of those valid triples, 382 were marked as true, 63 as false, and 16 as unknown to the human evaluator.

The percentage of valid triples is shown in table 5. The table shows that the triples

| Valid triples | | | |
|---|---|---|---|
| **`gpt-3.5-turbo` one-third** | **`gpt-4-turbo` one-third** | **`gpt-3.5-turbo` two-thirds** | **`gpt-4-turbo` two-thirds** |
| 97.6 ± 1.3% | 100% | 77.6± 3.7% | 93.6± 2.2% |

Table 5: Percentage of valid triples out of all evaluated triples.

generated by picking a predicate for a pair of correlated subjects and objects have a higher chance of being valid than the two-thirds generated triples. Furthermore, the model `gpt-4-turbo` is better at generating valid triples than the model `gpt-3.5-turbo`. This is especially true for the freely associated triples.

An example of an invalid one-third generated triple that was created by `gpt-3.5-turbo` is ⟨ *Elementary particle* | *has a negative charge* | *Electron* ⟩. The predicate "*has a negative charge*" is not valid because it does not link two concepts with each other. A valid triple would be ⟨ *Elementary particle* | *has a negatively charged instance* | *Electron* ⟩.

An example of an invalid two-third generated triple that was created by `gpt-3.5-turbo` is ⟨ *Canonical quantum gravity* | *is a theoretical framework* | *in the field of quantum gravity* ⟩. This is because the term "*in the field of quantum gravity*" is not a standalone concept but part of a sentence. The valid version of this triple is: ⟨ *Canonical quantum gravity* | *is a theoretical framework in* | *the field of quantum gravity* ⟩.

The percentage of the true triples out of all valid triples without unknown truth values is shown in table 6. A clear difference exists between the true triple share of the one-third

| True triples | | | |
|---|---|---|---|
| **`gpt-3.5-turbo` one-third** | **`gpt-4-turbo` one-third** | **`gpt-3.5-turbo` two-thirds** | **`gpt-4-turbo` two-thirds** |
| 67.5 ± 4.3% | 80.5 ± 3.6% | 98.9 ± 1.0% | 99.1 ± 0.8% |

Table 6: Percentage of true triples out of all valid triples without unknown truth value.

generated triples and the two-thirds generated triples. The former contain many more false triples than the latter. This is because the models have more difficulty finding an appropriate predicate for an existing pair of terms than creating both predicate and object from scratch. The valid two-thirds generated triples of both models `gpt-4-turbo` and `gpt-3.5-turbo` have such a high quality that only one false triple could be identified each in both datasets. For all four categories, an example of an incorrect triple is given in appendix A.

The average context dependency score of all true triples is shown in table 7. The table shows that the two-thirds generated triples are less dependent on the context than

| Average context dependency score | | | |
|---|---|---|---|
| **gpt-3.5-turbo one-third** | **gpt-4-turbo one-third** | **gpt-3.5-turbo two-thirds** | **gpt-4-turbo two-thirds** |
| $3.82 \pm 0.39$ | $3.56 \pm 0.34$ | $2.65 \pm 0.33$ | $2.27 \pm 0.24$ |

Table 7: Average context dependency score of all true triples.

the one-third generated triples. Also, the model `gpt-4-turbo` produces slightly less context-dependent triples than the model `gpt-3.5-turbo`.

An example of a triple with a low context dependency score is ⟨ *Electron shell* | *contains* | *Electrons* ⟩. This statement is true in all contexts. There is no situation where an electron shell does not contain any electrons. On the other hand, the triple ⟨ *Ring system* | *surrounds* | *Asteroid* ⟩ with a context dependency score of 10 is not true for most asteroids. But indeed, a ring system has been detected around the asteroid `(10199) Chariklo` [31].

The average triviality score of all valid triples is shown in table 8. It does not differ

| Average triviality score | | | |
|---|---|---|---|
| **gpt-3.5-turbo one-third** | **gpt-4-turbo one-third** | **gpt-3.5-turbo two-thirds** | **gpt-4-turbo two-thirds** |
| $5.70 \pm 0.26$ | $5.75 \pm 0.25$ | $5.61 \pm 0.21$ | $5.93 \pm 0.20$ |

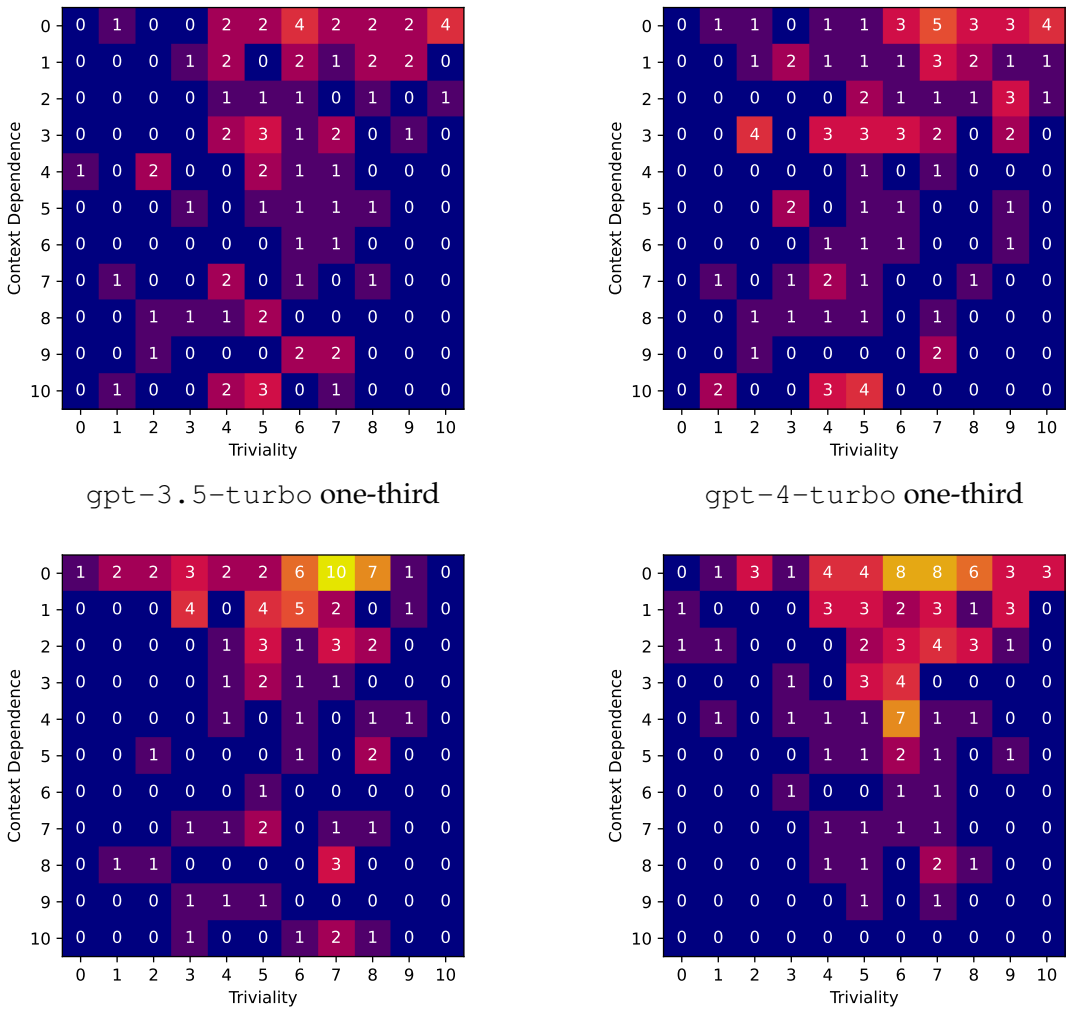
Table 8: Average triviality score of all valid triples.

much outside the error margin between the four datasets. They all contain trivial triples like ⟨ *Dark matter halo* | *is composed of* | *Dark matter* ⟩, which has a triviality score of 10, and less trivial triples like ⟨ *Hubbard model* | *predicts* | *Mott insulator phase* ⟩ which has a triviality score of 0.

Figure 13 shows a heat map of the context dependence plotted against the triviality for the true triples of all four datasets. All four data sets contain a broad spectrum of triples with different triviality and context dependency scores. In addition, all have a high concentration of triples with a context dependency score of 0 and a triviality value between 6 and 10. This is especially true for the two-thirds generated triples. Triples with very high and very low triviality are rare in all four cases. The distribution of the context dependency and triviality scores is very similar for the two models `gpt-4-turbo` and `gpt-3.5-turbo`.

### 3.4.3. Conclusion of the manual evaluation

In conclusion, the valid two-thirds generated triples of both models `gpt-4-turbo` and `gpt-3.5-turbo` have a very high quality and a higher than 98% chance of being true. The one-third generated triples of both models have a higher chance of validity but a significantly smaller chance of being true. This means that the correctness of the triple's statement tends to be higher when the models have more freedom to freely associate the triple by choosing the predicate and the object. In both categories, validness and truth, the `gpt-4-turbo` model performs better than the `gpt-3.5-turbo` model.




`gpt-3.5-turbo` one-third

`gpt-4-turbo` one-third

`gpt-3.5-turbo` two-thirds

`gpt-4-turbo` two-thirds

Figure 13: Heat maps of the context dependency plotted against the triviality of the true triples. The number of triples that end up in each bin is shown.

## 3.5. Visualizing the network of semantic triples

To visualize the generated semantic network, the `vis.js` Javascript library [32] has been used. The concepts (subjects and objects) were plotted as blue elipses and the predicates as green rectangles that connect the concepts with an arrow indicating the order in which the triples have to be read. Even though `vis.js` struggles to handle the whole dataset of 6494 triples at once, it can visualize small excerpts of 120 nodes just fine. Figure 14 shows an excerpt of the semantic network.

Because the network has an average of 12.5 connections per node, it is so dense that it is impossible to visualize without overlapping edges. To make it easier to get an overview of the semantic network, an algorithm was written that reduces the complexity of the semantic network by leaving out some of the connections. This is done so that it becomes possible to plot the graph without overlapping edges. The algorithm works by internally arranging the nodes in a hexagonal pattern where connected nodes are as close to each

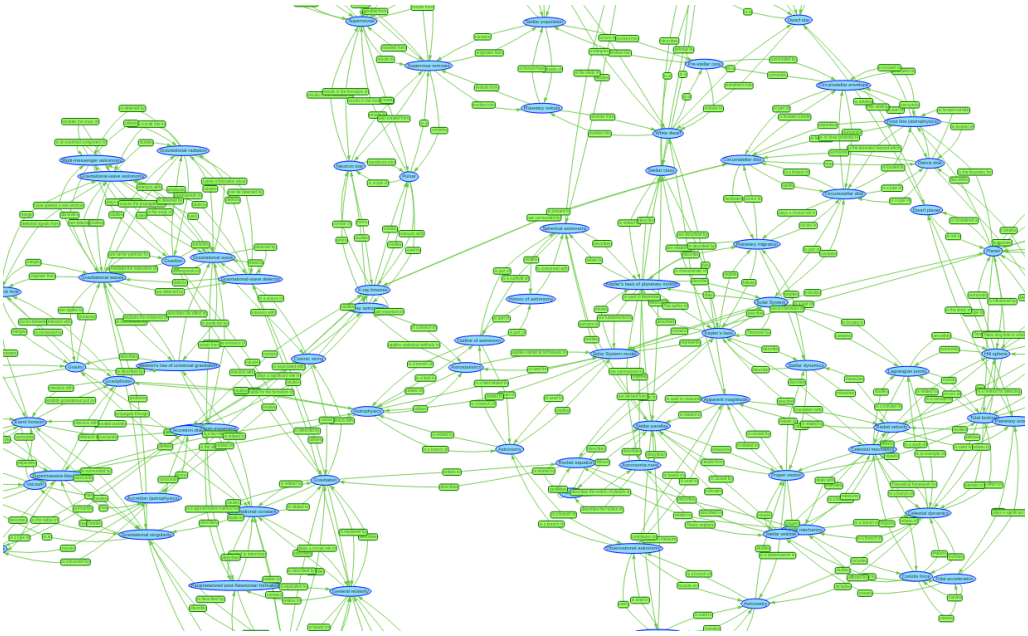


Figure 14: Excerpt of the fully connected semantic network of physics-related semantic triples.

other as possible. The algorithm then allows only connections between direct neighbors within the hexagonal pattern. In addition, the double-connected edges get replaced by single-connected edges. Figure 15 shows an excerpt of the semantic network that was flattened using this algorithm.

This customized view makes it much easier to read and understand the contents of the semantic network, even if some of the information from the original network is missing. Figure 16 shows a zoomed-in version of the semantic network that focuses on dark and exotic matter.

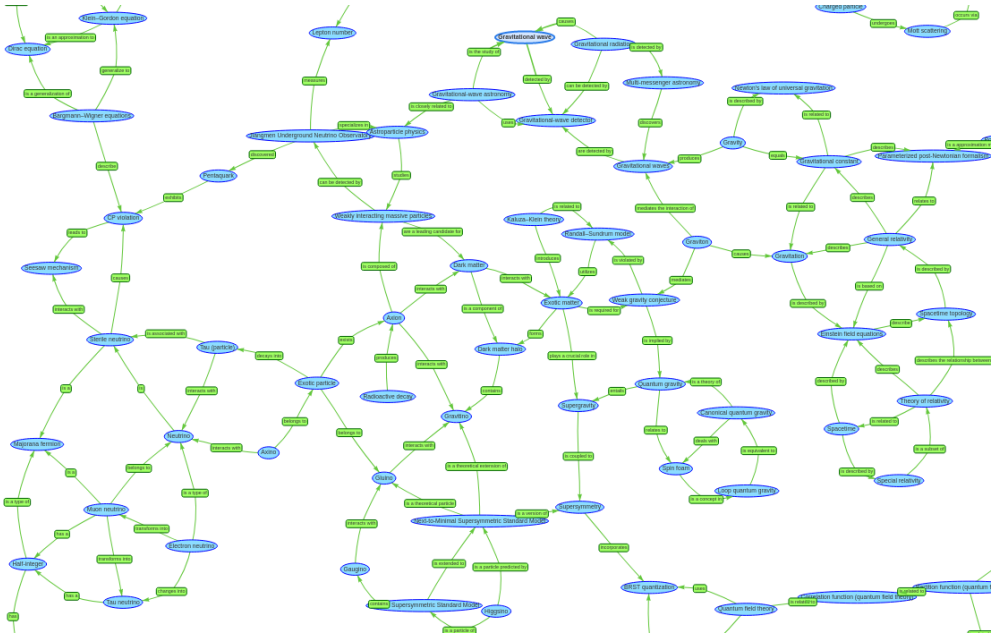


Figure 15: Excerpt of the flattened semantic network of physics-related semantic triples.

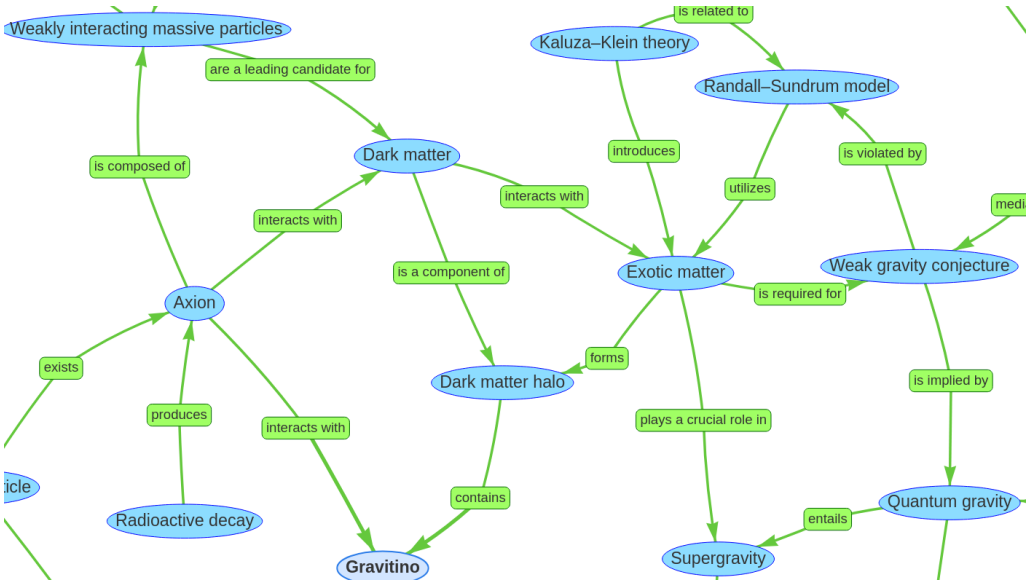



Figure 16: Zoomed in version of the flattened semantic network of physics-related semantic triples.

# 4. Answering physical questions using semantic networks

This chapter discusses the possibility of answering physics questions using semantic networks. In order to investigate if the answer to a specific question can be found in an knowledge base of semantic triples, a search algorithm is implemented. Next, it is tested if the relationships between questions and answers about physics topics can be formalized in a semantic network. As an use case of this approach, the physics knowledge of large language models is tested for consistency by letting them answer their own multiple-choice questions.

## 4.1. Searching a semantic network for answers to physics questions

How well the answer to a specific question can be found in a generated knowledge base of semantic triples is determined by two factors: the quality of the knowledge base and the quality of the search algorithm. The search algorithm tries to answer a question like "What do particle accelerators produce?" by navigating through the semantic network until it finds a matching semantic triple like $\langle$ *Particle accelerators* | *produce* | *High-energy particles* $\rangle$.

### 4.1.1. A setup of three agents for testing the search algorithm

For testing the knowledge base and the search algorithm, a test scenario of the following three agents as sketched in figure 17 was used: One generating agent who provides a knowledge base by generating semantic triples, one navigating agent who navigates through the knowledge of the generating agent to find the answer to a question, and finally, one examiner agent who asks the questions and decides whether the semantic triples answer them. The generating agent and the navigating agent play in a team and try to find the correct answer to the question of the examiner agent. The generating agent who provides the knowledge does not know the question. In contrast, the navigating agent does know the question but is only allowed to use the knowledge of the generating agent to answer it. The generating agent has to offer the navigating agent a range of options for continuing the navigation through its knowledge, and the navigating agent

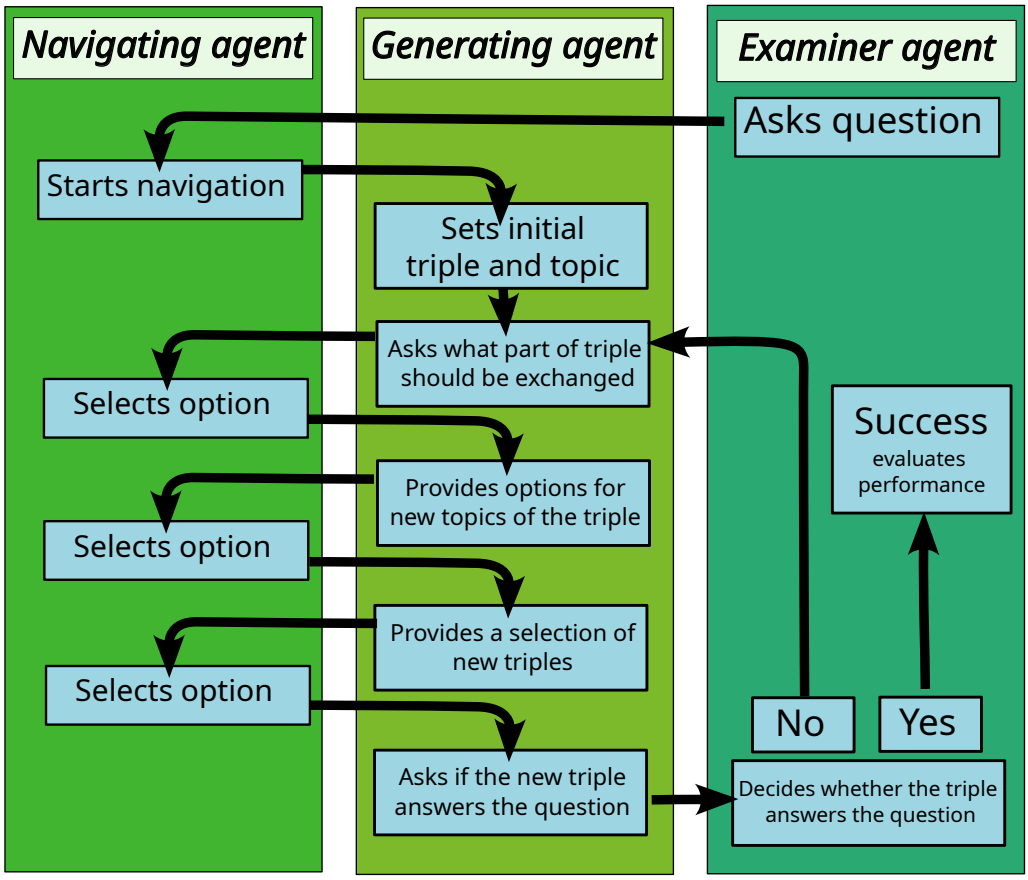



Figure 17: The three agents that are used to search the knowledge base of semantic triples for answers to physics questions.

is only allowed to communicate with the generating agent by telling it which of these options it wants to choose.

The type of navigation options that the generating agent offers determines the search algorithm that is used. For this investigation, the following approach was chosen to implement the search algorithm: It uses two variables that change during the navigation process: the current triple with the initial value $\langle$ *Physics* | *is a branch of* | *Science* $\rangle$ and the current topic with the initial value "*Physics*". A navigation iteration is performed in the following steps: The generating agent asks the navigating agent which part of the semantic triple should be changed. Six options are available for changing one or two of the three components of the triple. Next, the generating agent asks the navigating agent to choose a new topic from a list of ten topics. This list contains, besides the current topic, the last topic, and the topic "*Physics*" also five subtopics and two similar topics of the current topic that are generated by the generating agent. Finally, the generating agent generates ten new triples and lets the navigating agent choose from that list. These triples are generated by changing the chosen part of the old triple. The generating agent is also prompted to generate them in a way that they are related to the new topic.

### 4.1.2. Evaluating the search results

The evaluation of the triple search was carried out on a list of 100 questions generated by `gpt-4-turbo`. The requirement for these questions was that they could be answered by providing a semantic triple. For each question, the navigating agent and the generating agent performed a search limited to ten iterations.

The whole process was conducted twice. Once with `gpt-3.5-turbo` as navigating and generating agent and once with `gpt-4-turbo` as navigating and generating agent. The examining agent was in both cases `gpt-4-turbo` to ensure that the evaluation of the search results is comparable.

Out of the 100 triple search runs conducted for each model, `gpt-3.5-turbo` completed 27 successfully, while `gpt-4-turbo` completed 43 successfully. Table 9 shows the results of the `gpt-4-turbo` search for the question "What protects Earth from solar radiation?". It can be seen how the navigation process starts from the initial triple and

| Iteration | Triple | Topic |
|---|---|---|
| 0 | ⟨ *Physics* \| *is a branch of* \| *Science* ⟩ | Physics |
| 1 | ⟨ *Magnetism* \| *is studied as a branch of* \| *Science* ⟩ | Electromagnetism |
| 2 | ⟨ *Magnetism* \| *is a key component of* \| *Electromagnetism* ⟩ | Electromagnetism |
| 3 | ⟨ *Dynamo theory* \| *explains creation of* \| *Electromagnetism* ⟩ | Magnetic fields |
| 4 | ⟨ *Protective shield around Earth* \| *is created by* \| *Electromagnetism* ⟩ | Earth's magnetic field |

Table 9: The different iterations of the triple search for the question "What protects Earth from solar radiation?"

navigates through the knowledge base in four iterations until it reaches the triple that fulfills the condition of answering the question. This example also shows that the examiner agent is not completely strict in the requirements a tripe must fulfill to answer a question. The final triple ⟨ *Protective shield around earth* | *is created by* | *Electromagnetism* ⟩ does not explicitly state that the shield protects the earth from solar radiation. Another thing this example shows is that the triples generated by the generating agent are not always entirely correct. The triple ⟨ *Dynamo theory* | *explains creation of* | *Electromagnetism* ⟩ is not correct because the dynamo theory explains the creation of magnetic fields by the motion of conductive fluids and not the creation of electromagnetism itself [33].

Table 10 shows a list of five typical questions from the same dataset and the according triples that the navigation process has found. These examples show that the search algorithm was able to successfully navigate through the knowledge base and find the correct answer to the questions.

### 4.1.3. The decision cost metric

A metric to evaluate the quality of the search algorithm should be a measure of how much effort the navigating agent has to put into the search process to find the correct

| Question | Triple |
|---|---|
| What does the Tesla unit measure? | ⟨ *Magnetic flux density* \| *is quantified in* \| *Tesla* ⟩ |
| What does a laser emit? | ⟨ *Laser* \| *utilizes* \| *Electromagnetic waves in Science* ⟩ |
| What allows particles to overcome potential barriers in quantum mechanics? | ⟨ *Quantum tunneling* \| *exploits* \| *wave-particle duality* ⟩ |
| Which planet has an extensive ring system? | ⟨ *Saturn* \| *has the most extensive* \| *Planetary rings* ⟩ |
| What state of matter is composed of charged particles? | ⟨ *Ionosphirical plasma* \| *yields* \| *highly ionized particles* ⟩ |

Table 10: A list of five typical questions and the according triples that the navigation process has found.

answer. Therefore, it should count the number of combinations of decisions that would have been possible for the options that the generating agent offers during an average successful search process. For a successful search, that includes $N_{dec}$ decisions where each decision with the index $i$ selects one of $c_i$ options, the number of possible choice combinations $N_{com}$ is calculated as:

$$N_{com} = \prod_{i=1}^{N_{dec}} c_i \tag{4.1}$$

For convenience, an additive quantity is used to represent the number of choice combinations.

$$D = \log_{10}(N_{com}) = \sum_{i=1}^{N_{dec}} \log_{10}(c_i) \tag{4.2}$$

This quantity $D$ is called the decision cost of the search process. It corresponds to the number of navigation decisions the navigation agent has to make to find a question answering triple if ten options are available per decision.

The dataset of the 100 triple searches was used to estimate the search algorithm's average decision cost. For the searches that were interrupted after ten unsuccessful iterations it needs to be assessed what the average decision cost would be if the search had been continued until a successful triple was found. Therefore, the approximation is made that the probability of finding a triple within a single search iteration is a constant independent of the previous search iterations which means that it makes no difference if one continues after ten search iterations or starts again from the beginning. This is a necessary approximation that neglects the progress that has been made in the previous search iterations and therefore leads to an overestimation of the average decision cost. Now it is assumed that in average $N_{rep}$ search runs of a maximum of ten iterations are needed to find a triple. $N_{rep}$ is the inverse of $P_{suc}$ the probability that the triple can be found within ten iterations. The average decision cost of an uninterrupted search $D_{avg}$ can then

be calculated from the average decision cost of an unsuccessful interrupted search $D_{fail}$ and the average decision cost of a successful interrupted search $D_{suc}$ as:

$$D_{avg} = (N_{rep} - 1)D_{fail} + D_{suc} = (\frac{1}{P_{suc}} - 1)D_{fail} + D_{suc} \tag{4.3}$$

The calculated decision cost for the implemented search algorithm of `gpt-3.5-turbo` and `gpt-4-turbo` is shown in table 11. The model `gpt-4-turbo` performed signifi-

| **decision cost** | |
|---|---|
| **gpt-3.5-turbo** | **gpt-4-turbo** |
| $91.1 \pm 17.0$ | $50.7 \pm 7.1$ |

Table 11: The calculated decision cost.

cantly better than the model `gpt-3.5-turbo`.

The questions used for the evaluation contain, on average, 51 characters. An alternative setup with a generating agent that cheats by asking the navigating agent letter by letter what the question is would have a decision cost of 73 when the generating agent offers the 26 letters of the English alphabet plus a space as options.

$$\log_{10}(27^{51}) \approx 73 \tag{4.4}$$

The fact that the decision cost of `gpt-4-turbo` is lower than this value indicates that the search algorithm is quite efficient.

## 4.2. A semantic network of questions and answers

A semantic network of physics questions and answers is introduced that contains three types of nodes: questions, statements, and topics. Questions can be answered by statements, indicated by the relation `answers`. Statements can support or contradict each other, indicated by the relation `supports` or `is contradiction to`. They also can contain other statements as substatements, which is indicated by the relation `has substatement`. A statement is a substatement of another statement if the truth of the second statement implies the truth of the first statement. Both statements and questions can be related to topics, which is indicated by the relation `has topic`. Statements can be either true or false. This information is not included in the network but has to be determined by the external observer. Figure 18 shows an example of such a network. In this network, different aspects of the questions "What is the speed of light?" and "What quantity is independent of the inertial frame of reference?" are discussed. The network contains five different statements that are connected to the questions. The statement "The speed of light is 299,792,458 meters per second in a vacuum" answers the first question and contains the substatement "Light waves can travel through the vacuum." The statement, "The speed of light is a natural constant that is independent of the inertial frame of ref-

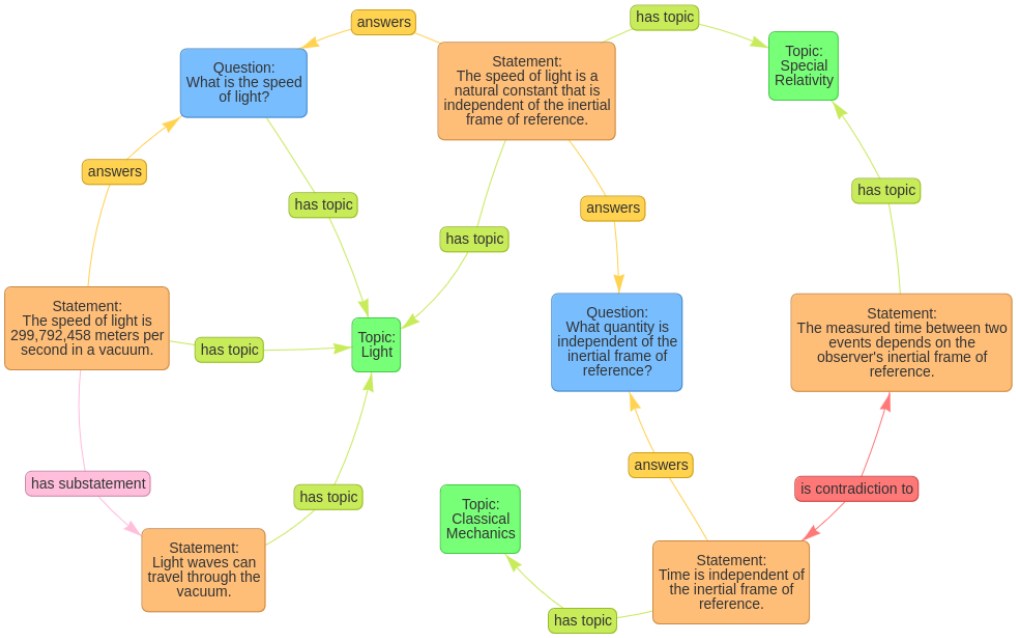



Figure 18: A semantic network of questions and answers about the speed of light.

erence, " answers both questions. Additionally, the second question is answered by the statement, "Time is independent of the inertial frame of reference.". This statement is only true for non-relativistic physics and, therefore, contradicts the statement, "The measured time between two events depends on the observer's inertial frame of reference.". The example demonstrates how a semantic network can represent the relations between different questions, statements, and topics.

### 4.2.1. Generating semantic networks of questions and answers with a single prompt

In the next step, large language models will be used to generate a semantic network of physics questions and answers. Because the structure of the network is quite complex and the resulting graph can contain circular dependencies, it is pretty challenging to develop an algorithm that assembles the network step by step. Therefore, an approach was used where `gpt-4-turbo` was asked to generate the whole network in a single prompt. This makes it possible for the model to not only generate the network's text content but also its general structure. To do this, a text representation format of the network that the model should use was defined first. Therefore, a JSON format [34] was chosen that contains the questions, statements, topics, and the relations between them. This format needed to be explained to the model in the prompt to generate the network. To make the model understand which type of network it should generate, the example from figure 18 was converted into the JSON format and inserted in the prompt. Additionally, the model was provided with a topic for which it should generate the network. This topic was also included in the prompt. Finally, some tests were performed on the generated networks

to check if their format was correct. The generation process was repeated until the format of the generated network contained no errors.

For the evaluation, 20 networks were generated and analyzed for different topics. The generated networks have between 11 and 22 nodes and between 9 and 22 edges. They contain an average of 6.1 statements, 4.5 questions, and 4.4 topics. The average number of "answers" relations is 4.9, the average number of "has topic" relations is 8.4, and the average number of "has substatement" relations is 1.5. In all 20 networks, only one occurrence of the "is contradiction to" relation exists. The topologies of the generated networks are diverse. They vary from star-shaped networks with a central topic node to more complex, highly interconnected networks and even disconnected networks with multiple components.

The connections "answers" and "has topic" are mainly used correctly, but the connection "has substatement" is often misused. Instead of connecting a statement with a partial statement, the model often uses this connection to connect a statement with a completely different statement. The connection "is contradiction to" is barely used, probably because the model tries to avoid contradictions and generate only correct statements. In the one example where this connection is used, the two statements "Reversible processes are idealized processes where the system remains in thermodynamic equilibrium" and "In practical situations, true reversibility is never achieved because it requires an infinite amount of time" are not contradictory, but rather complementary because they distinguish between idealized and practical situations.

The content of the generated statements is primarily correct but sometimes not very precise. The second statement in the example above could be less ambiguous by stating that this is the case for thermodynamics because, in classical mechanics, all processes are reversible without any time constraints. In another example, the model describes the right-hand rule for the Lorentz force without mentioning that it is only valid for positive charges. In general, the model seems to understand the topics well and can generate correct statements about them.

Figure 19 shows the network generated for the topic "*Bell's theorem*". This network consists of four statements, three questions, and four topics. The relation "answers" connects three of the statements with the questions. Two of these connections are correct, while in the third case, the statement about quantum entanglement does not mention Bell's theorem and, therefore, does not answer the question, "How does quantum entanglement relate to Bell's theorem?". The relation "has topic" connects all statements and questions with the four topics. The assignment of the topics to the statements and questions is correct, even if the tags "quantum mechanics" and "Bell's theorem" would fit all statements and questions. The one "has substatement" relation is not used quite correctly because the statement about non-local hidden variables does not contain the statement that Bell's inequality is used to measure correlations between entangled particles.

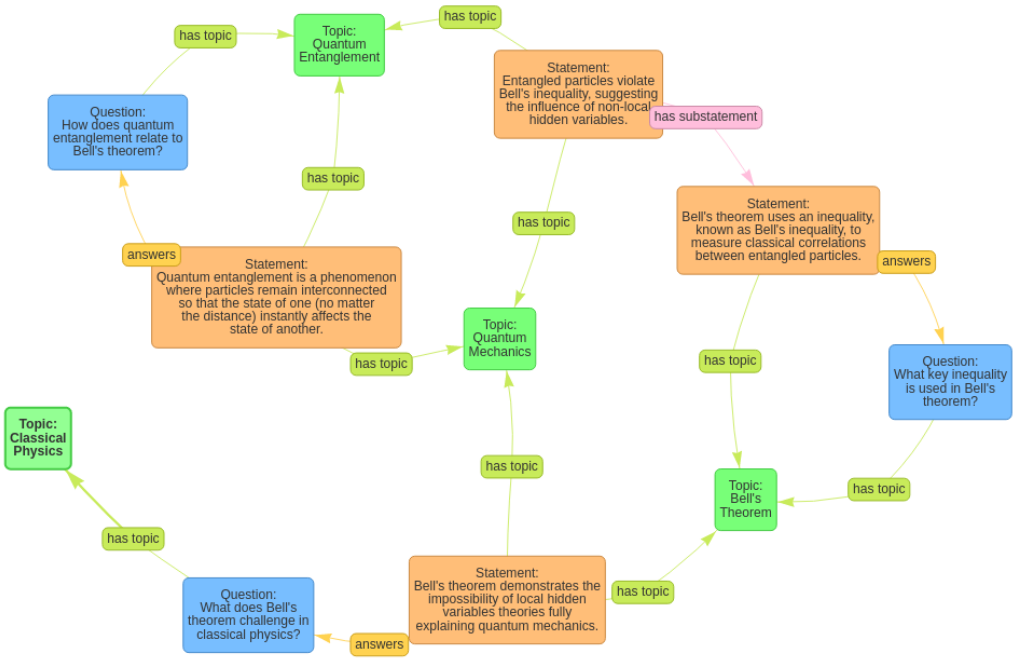



Figure 19: A semantic network of questions and answers about Bell's theorem that was generated in a single prompt.

## 4.3. Testing the consistency of the physics knowledge of large language models

The principle of questions and answers assigned to specific topics can be used to test the consistency of the physics knowledge of large language models and evaluate the extent to which their understanding of different physics terms contains contradictions. To do this, an excerpt of 100 handpicked terms from the Oxford Dictionary's list of physics terms was used. For each of these terms, `gpt-3.5-turbo` was asked to generate a list of 30 questions about this term. In the next step, `gpt-3.5-turbo` was asked to provide a list of 5 answers to each of these questions. One of these answers should be correct, and the other four should be wrong. However, they should all sound plausible to the layman. Next, `gpt-3.5-turbo` was asked to identify the correct answer to each question. If the model is able to do this with a high accuracy, it can be assumed that the knowledge about this term is consistent.

From the total of 3000 multiple choice questions generated, $83.7 \pm 0.7\%$ were answered correctly by `gpt-3.5-turbo`. The percentage of correct answers was calculated for each term from the list. The five terms with the highest percentage of correct answers are "*active galactic nucleus*" (100%), "*Archimedes' principle*" (100%), "*Heisenberg uncertainty principle*" (100%), "*Newton's law of gravitation*" (97%) and "*Higgs boson*" (97%). The five terms with the lowest percentage of correct answers are "*Copenhagen interpretation*" (37%), "*ideal gas*" (53%), "*Compton effect*" (57%), "*relativistic mass*" (60%) and "*de Broglie wavelength*" (67%). Note that the statistical error of this evaluation is up to 9%.

It is interesting to note that the term "*Copenhagen interpretation*" with the lowest percentage of correct answers is closely related to the term "*Heisenberg uncertainty principle*" with one of the highest percentages of correct answers. This might be because the Heisenberg uncertainty principle is a well-defined mathematical concept, while the Copenhagen interpretation is a more philosophical concept that can be interpreted in different ways. In addition, the Copenhagen interpretation is a collective term for several different interpretations of quantum mechanics, which might lead to contradictions in the answers.

The term "*ideal gas*" with a low percentage of correct answers is also interesting, as it is a well-defined concept in classical physics. The main problem here seems to be that the model is unsure whether the particles in an ideal gas interact. This is a typical example of a contradiction in the model's knowledge, which might be because this contradiction also appears in the training data. For example, the German Wikipedia article about the ideal gas states that there can be elastic collisions between the particles [35], while the English Wikipedia article states that the particles are not subject to interparticle interactions even if this requirement of zero interactions can often be relaxed [36]. The possibility of elastic collisions between the particles also seems to contradict the fact that the particles are assumed to be point-like with no volume and, therefore, have an infinitely small cross-section. On the other hand, the possibility of elastic collisions is compatible with most of the ideal gas's derived properties, such as the ideal gas law, so it would make sense to include it in the definition of the ideal gas.

In the case of the term "*Compton effect*", the model seems to have problems deciding whether the wavelength of the scattered photon is longer or shorter than the wavelength of the incident photon. This ambiguity seems to be a general problem for the model in understanding the Compton effect. The Compton effect describes the scattering of a photon by a charged particle, which leads to a change in the wavelength of the photon. When the charged particle is at rest before the scattering, the wavelength of the scattered photon is longer than that of the incident photon because the photon loses energy to the charged particle. Only if the charged particle moves before the collision and transfers kinetic energy to the photon, the wavelength of the scattered photon will get shorter than the wavelength of the incident photon. This case is referred to as the inverse Compton effect. It seems that `gpt-3.5-turbo` is confused by this fact and, therefore, gives contradictory answers to questions about the Compton effect.

Besides the consistency of the knowledge about the terms, it can also be analyzed how diverse the knowledge about the terms is. If the model has extensive knowledge about a term, it should be able to generate a wide variety of diverse questions about this term. On the other hand, if the model only has superficial knowledge of a term, all questions about this term should be very similar. For this reason `gpt-3.5-turbo` was asked to determine the diversity of the questions about each term by assigning a diversity score to each group of 30 questions. This score should range from 0 to 100, where 0 means all questions are identical and 100 means all questions are entirely different. For more

consistent results, the model was asked five times, and the average of the diversity scores was calculated.

The five terms with the highest diversity of questions are: "*Copenhagen interpretation*" (92), "*conservation law*" (91), "*uncertainty principle*" (91), "*Yang–Mills theory*" (91) and "*big-bang theory*" (90). The five terms with the lowest diversity of questions are: "*BCS theory*" (22), "*Gibbs free energy*" (32), "*pressure*" (33), "*Higgs boson*" (33) and "*Compton effect*" (42).

The term with the lowest diversity of questions "*BCS theory*" is a complex theory in condensed matter physics. Most of the questions about this term are about the basic principles of the theory, such as the formation of Cooper pairs and the superconducting state. A typical question about the BCS theory is: "What is the BCS theory and how does it explain superconductivity?". This question is very similar to many other questions about the BCS theory, which leads to a low diversity score.

The term with the highest diversity of questions "*Copenhagen interpretation*" is at the same time the term with the lowest percentage of correct answers. It seems that the model has a lot of different ideas about the Copenhagen interpretation, which leads to a high diversity of questions. Typical questions about the Copenhagen interpretation are: "What is the Copenhagen interpretation, and how does it explain the behavior of subatomic particles?" and "What is the role of the observer in the Copenhagen interpretation of quantum mechanics?"

Figure 20 shows the correlation between the percentage of correct answers and the diversity of the questions for a selection of terms:

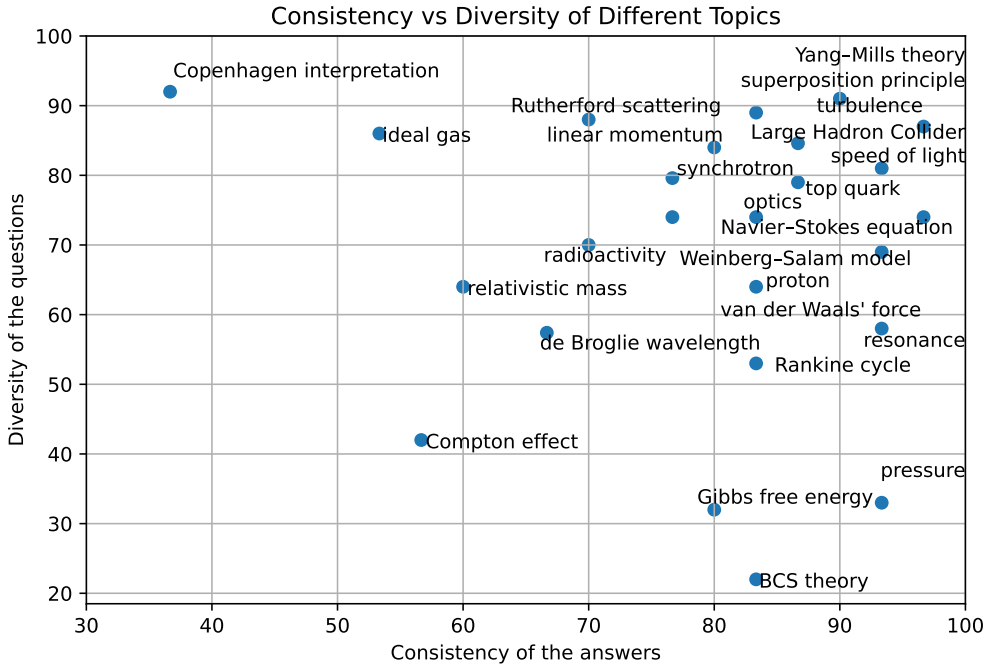



Figure 20: Correlation between the percentage of correct answers and the diversity of the questions for a selection of terms.

# 5. Extracting semantic network data from scientific texts

This chapter discusses different approaches to using external sources to obtain data for semantic networks. It will be shown how the citations in scientific papers can be used to create a citation graph. To investigate how specific information can be identified in those papers, an example from the subject area of top quark tagging is used to extract the properties of different machine learning models. Finally, it will be investigated how not only specific properties but the essence of a whole text can be extracted and represented in a semantic network. This will be tested using two different approaches: one that uses semantic triples and one that uses a network of sentences.

## 5.1. Creating a citation graph

Scientific papers published in scientific journals usually refer to each other using citations. It will now be explored how a citation graph can be created from the references in the papers. A citation graph is a semantic network that consists of papers as nodes and citations as directed edges between the nodes. There are two main challenges in creating a citation graph to identify each paper with a unique identifier and to extract the references from the papers. A typical solution to the first challenge is to use the digital object identifiers (DOI) of the papers as unique identifiers. A DOI is a string assigned to a document by the International DOI Foundation [37]. However, not all papers have a DOI, so in some cases, it may be necessary to use other identifiers. The second challenge is not easy to solve using an automated approach because not all papers provide metadata about the references. However, this challenge can be addressed by using neural networks that are trained to extract the references from the papers.

The semantic scholar project solved both challenges and created a citation graph containing more than 200 million papers [38]. The data was obtained from publisher partnerships, data providers, and web scraping. The semantic scholar project uses its own unique identifiers to identify the papers. The references were extracted using the "*ScienceParse*" system based on recurrent neural networks [39]. The project offers an API to access the data and query the citation graph.

The possibility of obtaining semantic network data from the semantic scholar API was tested by creating a citation graph for papers about machine learning in particle physics. This topic is a relatively new field of research that deals with the question of how machine

learning methods can be used to tag, analyze, and generate data of particle collision events. As a basis for the citation graph, the *Living Review of Machine Learning for Particle Physics* [40] was used. This review is a web page that contains a list of papers about machine learning in particle physics. These papers were extracted from the page using the Python package `beautifulsoup` [41]. The extracted data consists of the title of the publication together with a link to the source. In some but not all cases, the DOI of the paper is also provided. In all cases where the living review did not provide the DOI, the link to the source was used to obtain information about the paper. This was done by adjusting a scraping algorithm to handle the different formats of the web pages of the sources like `arxiv.org` or `nature.com`. The obtained DOI or arXiv identifier was then used to query the semantic scholar API for their internal identifier of the paper. A total of 912 papers were extracted from the living review.

As an additional source of papers the website *INSPIRE-HEP* [42] was used. This website contains a database of scientific articles with more than 1 million entries [6]. It also provides an API that can be used to query the database. From this API, a total of 2223 papers were extracted by searching for the terms *"machine learning*" or *"deep learning"* or *"neural"* and the categories "`hep-ex`" or "`hep-ph`" or "`hep-th`" that stand for experimental, phenomenological and theoretical particle physics. The data extracted from the *living review* and *INSPIRE-HEP* have an overlap of 726 papers.

With the help of the semantic scholar API, a citation graph was created for the extracted papers. The data was stored in the graph database `Neo4j`. The resulting graph contains 2409 papers and 14982 citations. 53 papers from the two sources could not be included in the database because they contain errors, such as missing identifiers or citation information. On average, each paper has 6.22 references within the database. Two papers connected within the citation graph have an average distance of 3.64 citations. The most cited paper in the citation graph is the paper *Jet-images - deep learning edition* [43] with 175 citations of other papers also contained in the database. The paper that cites the most other documents within the database is the paper *A Living Review of Machine Learning for Particle Physics* [44] with 385 citations. The latter is the paper describing the living review used as a starting point for the creation of the citation graph. It is interesting to note that only two citations are listed in the pdf version of this paper. However, the paper contains a link to the living review that includes the full list of citations. It seems that semantic scholar was counting these indirect citations as well.

These results show that a citation graph can be a helpful tool to analyze the publications in a specific field of research that can also be used to identify outstanding papers.

## 5.2. Extracting the R30 value and parameter count from papers about top quark tagging

Next, a demonstration should show that it is possible to extract specific properties from scientific papers that can later be used to build a semantic network. For this purpose, the R30 value and the parameter count of different top tagging models were extracted from papers about top quark tagging.

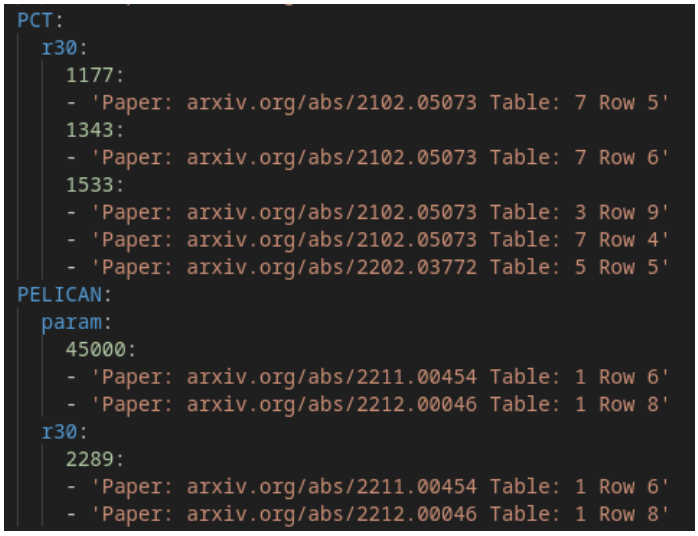
The tagging of top quarks refers to the process of identifying the decay products of top quarks in high-energy physics experiments. Top quarks are the heaviest known elementary particles and decay almost instantaneously into lighter particles. Typically, their decay products are a bottom quark and a W boson, which then decays into a lepton and a neutrino or a pair of quarks.

A top tagging algorithm recognizes the fingerprints of these decay products in the detector data and uses this to decide if a specific particle event contains a top quark. Usually, the sensitivity of such an algorithm can be adjusted to identify a higher percentage of the top quark events within the dataset. This also comes with the cost of producing a higher false positive rate of events without top quarks. The R30 value is the inverse of the false positive rate at a signal efficiency of 30%. A higher R30 value means a better top tagging performance. Another relevant quantity is the parameter count of different top tagging models. This is the number of trainable parameters in the neural network of the machine learning model used for top tagging. A higher number of parameters usually means that it needs more computing power to train and use the corresponding machine-learning model.

The papers "The Machine Learning Landscape of Top Taggers" [45] and "Feature Selection with Distance Correlation" [46] have been used as a starting point to obtain a list of papers that contain information about top tagging models. In the next step, an algorithm was applied to extract the papers' R30 value and parameter count. Even though it would have been faster to extract the information manually in this case, this approach was chosen to investigate the feasibility of automating the extraction process. In the selected papers, the relevant information is stored in tables. To obtain the tables in a machine-readable format, the algorithm uses the HTML versions of the papers available on the arXiv preprint server. The algorithm then searches for specific keywords in the tables that indicate that they contain the R30 value or the parameter count. During the creation of the algorithm, multiple iterations of adjusting the search terms and the search strategy were necessary to achieve the desired results.

39 papers were used for the extraction. In 29 of them, no information about the R30 value and the parameter count was found. From the papers, 124 tables were extracted. 13 of them were used to obtain R30 values, and 9 to obtain parameter counts. A total of 39 different machine-learning models were identified in the tables. The number of extracted R30 values is 54, and the number of extracted parameter counts is 48. Figure 21 shows an

excerpt of the extracted data. As one can see, some models have multiple different values for the R30 and the parameter count. This is due to different versions of the models that use different hyperparameters. The extracted data was later used to create a semantic network as discussed in section 6.3.



```
PCT:
  r30:
    1177:
    - 'Paper: arxiv.org/abs/2102.05073 Table: 7 Row 5'
    1343:
    - 'Paper: arxiv.org/abs/2102.05073 Table: 7 Row 6'
    1533:
    - 'Paper: arxiv.org/abs/2102.05073 Table: 3 Row 9'
    - 'Paper: arxiv.org/abs/2102.05073 Table: 7 Row 4'
    - 'Paper: arxiv.org/abs/2202.03772 Table: 5 Row 5'
PELICAN:
  param:
    45000:
    - 'Paper: arxiv.org/abs/2211.00454 Table: 1 Row 6'
    - 'Paper: arxiv.org/abs/2212.00046 Table: 1 Row 8'
  r30:
    2289:
    - 'Paper: arxiv.org/abs/2211.00454 Table: 1 Row 6'
    - 'Paper: arxiv.org/abs/2212.00046 Table: 1 Row 8'
```


Figure 21: Extracted R30 values and parameter counts from different papers. The different hierarchy levels of the data are the name of the model, the type of the extracted value (R30 or parameter count), the value itself, and the source of the value.

The results show that the method described in this section can be used to extract specific properties from scientific papers. Even though the algorithm works well for the given example, there is still a lot of room for improvement to make it more general and applicable to a broader range of papers and tasks. One feature that has proven to be useful and should be retained in future versions of the algorithm is that the data is stored together with references to the source which makes it easy to verify the correctness of the extracted values.

## 5.3. Converting sentences into semantic triples and back

Next, an experiment should test if sentences about physics can be converted into semantic triples and back without losing information. Therefore, the Wikipedia article about electrons [47] was used. The text was extracted from the website and split into sentences using the natural language toolkit [48]. Then, the dataset of sentences was filtered by the large language model to include only sentences that stand by themself and do not contain any references to the context. The resulting dataset contains 231 sentences. In the next step, `gpt-3.5-turbo` was used to convert the sentences into semantic triples using the following prompt:

**Prompt for converting sentences into semantic triples**

```
The sentence "The Earth orbits the sun at a distance of 1
AU." can be converted to the semantic triples:
[["Earth", "orbits", "Sun"],
["Earth", "has a distance from the sun of", "1 AU"]]
What are the semantic triples for the sentence:
" <sentence> "?  Return nothing but the semantic triples in
the format above.
```

As a result, an average of 2.99±0.11 triples were generated per sentence. Of the 690 triples, 84 contain the words "*electron*" or "*electrons*" as subject while 25 contain one of those words as object. The most frequently used predicates are "*is*", "*is a*", and "*are*", which together are used 41 times. The predicates "*has*" and "*have*" are used 37 times. The predicate "*include*" is used 27 times.

While the triples mostly contain the most essential elements of the sentences, some nuances of the sentence content are lost during the transformation. For example, the sentence "Laboratory instruments are capable of trapping individual electrons as well as electron plasma by the use of electromagnetic fields." is converted into the triples ⟨ *Laboratory instruments* | *are capable of trapping* | *individual electrons* ⟩, ⟨ *Laboratory instruments* | *are capable of trapping* | *electron plasma* ⟩ and ⟨ *Laboratory instruments* | *use* | *electromagnetic fields* ⟩. Therefore, the nuance is lost that the first two triples are explicitly achived by the use of the electromagnetic fields from the third triple.

To test how much information was lost by transforming the sentences into triples, the triples were transformed back into sentences, which were then compared to the original sentences. The back-transformed version of the triples from the previous example is "Laboratory instruments are capable of trapping individual electrons, electron plasma, and use electromagnetic fields.". The length of the back-transformed versions of the sentences is, on average, 10.6±1.6% shorter than the length of the original sentences. This indicates that approximately 10% of the information was lost during the transformation.

To analyze what kind of information was lost, it was calculated which the most common words that appeared in the original text but were missing in the transformed text were. Among those words are terms like "when" (left out 10 times), "by" (left out 10 times) and "because" (left out 9 times). These terms link different parts of the sentences and create causalities between them. They are essential to construct more complex statements that can not be expressed purely by the more compact semantic triples. This missing expressiveness illustrates the limitations of the approach to represent knowledge as semantic triples.

## 5.4. Semantic network of sentences

To overcome the limitations of the semantic triples, a different design for a semantic network, which we call a "semantic network of sentences," was tested. This semantic network consists of terms and sentence fragments linked with each other. Figure 22 shows a simple example of such a network. This network consists of four terms, illustrated by

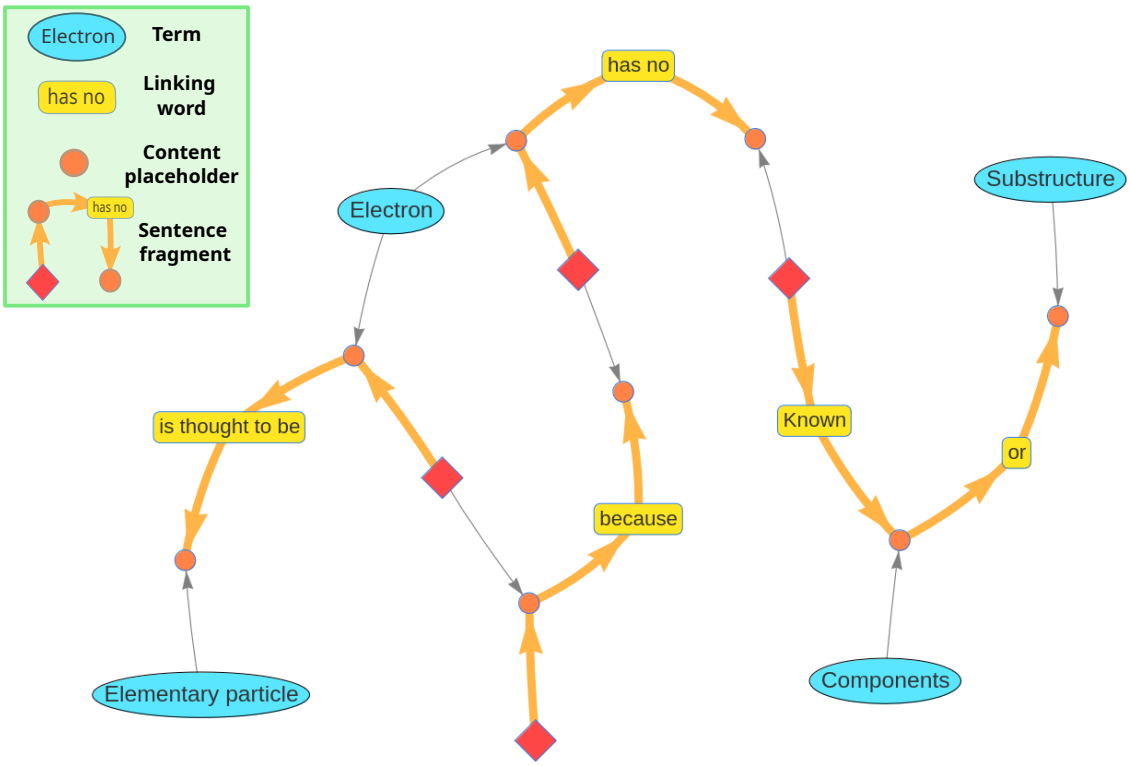



Figure 22: Simple example of a semantic network of sentences that represents the sentence "An electron is thought to be an elementary particle because it has no known components or substructure."

blue ellipses and four sentence fragments, depicted by red diamond shapes followed by a tail of orange arrows. The terms correspond to the subjects and objects of the network of semantic triples and represent concepts that stand by themselves. The example network uses the terms "*Electron*", "*Elementary particle*", "*Components*", and "*Substructure*". The sentence fragments are sequences of content placeholders and linking words that can be read like a natural language sentence in the order indicated by the orange arrows. The content placeholders are depicted by the little orange circles. For each content placeholder, a thin grey arrow shows which content has to be inserted at that place. For example, terms like "*Electron*" or "*Elementary particle*" can be inserted into the content placeholders. The linking words are depicted by the yellow boxes. Their role is to connect the concepts similar to the predicates in the network of semantic triples. The leftmost sentence fragment in the graphic represents the sub-sentence, "An electron is thought to be an elementary particle". It connects the concepts "*Electron*" and "*Elementary particle*" with the linking words "is thought to be". In this case, the content of the sentence fragment is a statement. A

sentence fragment can also describe a concept. For example, the rightmost sentence fragment in the graphic represents the concept of "Known components or substructure". The content placeholders can not only be placeholders for the terms but also for the content of other sentence fragments. In this case, the grey arrow points from the red diamond shape of the inserted sentence fragment to the content placeholder in which it is inserted. For example, the topmost sentence fragment in the graphic represents the sentence "Electron has no known components or substructure" by inserting the leftmost sentence fragment into its second content placeholder. The bottommost sentence fragment finally connects the leftmost and topmost sentence fragments and creates a causality between them.

### 5.4.1. Manual creation of a semantic network of sentences

This design was tested by manually building a semantic network of sentences for the first paragraph of the Wikipedia article about electrons. The resulting network is shown in figure 23. It consists of 33 sentence fragments and 28 terms. Among the sentence

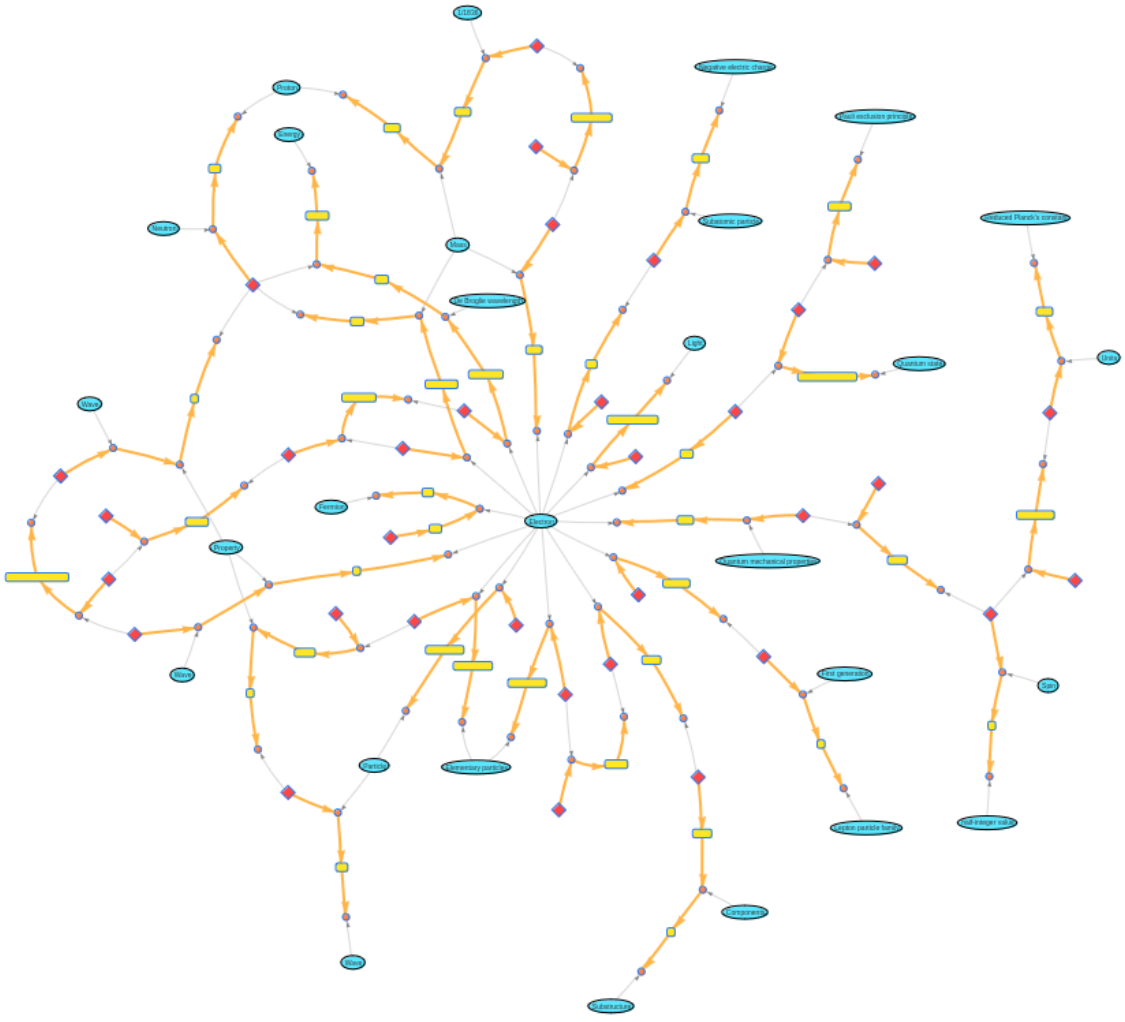


Figure 23: Semantic network of sentences for the first paragraph of the Wikipedia article about electrons.

fragments are 14 that describe concepts and 19 that describe statements. The sentence fragments have an average length of 3.4 components (content placeholders and linking words). Their content placeholders contain links to 0.67 other sentence fragments and to 1.5 terms on average. The most connected node is the term "*Electron*", connected to 14 sentence fragments. It is shown in the center of the figure. The network can store most nuances of the text used because it contains the different parts of the sentences that are nearly unchanged. It also extracts all concepts that occur in the text and links them with each other. The approach is scalable but requires a lot of curation work. The main challenge is linking cross-references between sentences and disassembling them into their atomic parts.

### 5.4.2. Automated extraction of a semantic network of sentences from a text

After manually creating the semantic sentence network, an attempt was made to automate converting a given text into this format. For this purpose, a pipeline was built that starts with the plain text and splits it into sentences using the natural language toolkit Python package. Next, the coreferences within the sentences get resolved by asking `gpt-4-turbo` to replace all occurrences of references like "it" or "this" with the actual words that they reference. Finally `gpt-4-turbo` is asked to mark the concepts within the resolved sentences. These concepts are then used as the terms, while the text between them is used as linking words. In contrast to the manual approach, this algorithm does not disassemble the sentences into their atomic parts. Instead, it creates one sentence fragment for each sentence. This also means it never uses other sentence fragments as content for the content placeholders, but only terms.

The algorithm was tested on the same paragraph used for the manual creation. The resulting network is shown in figure 24. The network consists of 6 sentence fragments and 29 terms which means that the network has a similar number of concepts but a much lower number of sentence fragments than the manually created one. The sentence fragments have an average length of 14.2 components and contain links to 5.8 terms on average which makes them longer with a higher number of links than the manually created sentence fragments. In contrast to the manually created network, the automatically created network is not fully connected but consists of two separate parts. This is because the algorithm distinguishes between the terms "*Electron*" and "*Electrons*". In the manual creation of the network, this distinction was not made. Even if the generated network is not as fine-grained as the manually created one, these results show that it is possible to convert a text into a semantic network of sentences using a large language model.

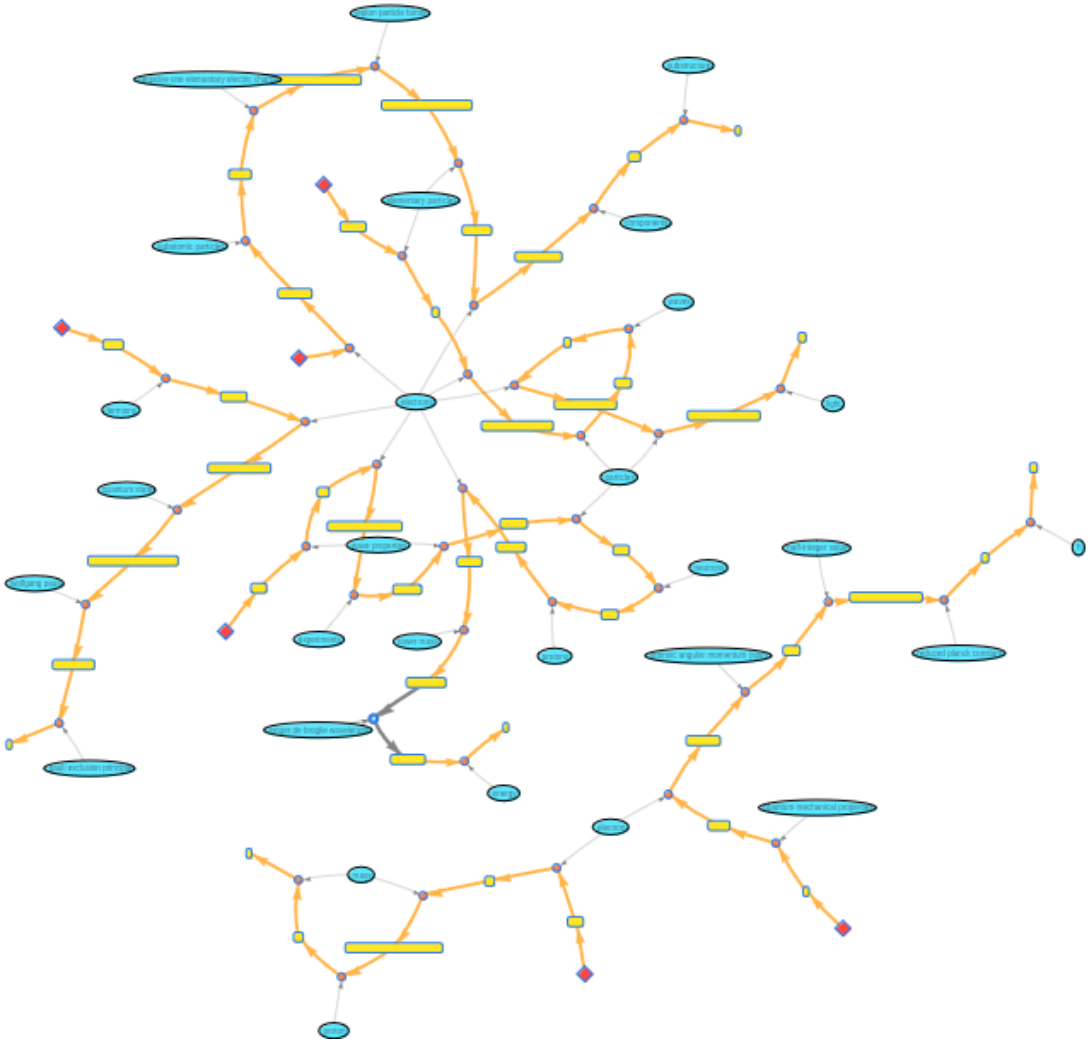


Figure 24: Automated semantic network of sentences for the first paragraph of the Wikipedia article about electrons.

# 6. Creating a physics ontology

This chapter analyzes the possibilities of using an ontology to structure the knowledge of physics. It starts with a general introduction to the concept of ontology and then introduces the Physci ontology [49] as an example of an ontology that was created to represent knowledge in the field of physics. Next, a customized ontology for machine learning in physics is introduced in order to demonstrate how an ontology can be adapted to a specific use case. Then, the question of whether the process of creating a physics ontology can be automated is investigated. An algorithmic approach is introduced that can generate classes and relations for the ontology. Finally, the chapter investigates whether the process of creating a knowledge graph that is based on the ontology can also be automated.

## 6.1. What is an ontology?

Using semantic triples without further specifications allows one to keep the structure of the semantic network as general as possible so that a wide range of information can be represented. However, great flexibility in structuring the data means that standardizing the information for automatic evaluation is more complicated. Precise specifications regarding the structure of the semantic network, on the other hand, can lead to a more explicit content structuring of the knowledge graph and enable targeted processing and evaluation of the physical data, which is, however, restricted to a smaller area of knowledge. Such a clearer structure of a semantic network can be achieved with the help of an ontology.

An ontology is a means to formally model the structure of a system [50]. This means that an ontology defines rules about which type of concept classes are allowed and which type of connections can exist between the concepts of the different classes. The most popular language for creating ontologies is the Web Ontology Language (OWL) [51]. OWL is based on the Resource Description Framework (RDF) and is used to create ontologies in a machine-readable format. The most important terms used in the context of an OWL ontology are classes, properties, and individuals. Classes define the types of objects that can exist in the ontology. They can be hierarchically structured into subclasses and superclasses. Properties are used to define relationships between objects. OWL distinguishes between object properties and data properties. Object properties define relationships between objects, while data properties define relationships between objects and data values. Individuals are used to define specific instances of classes. They can be connected to each

other via properties. Individuals are typically not part of the ontology itself but are used to represent the data stored based on it.

## 6.2. The Physci ontology

The question of how to use an ontology to bring physical research data into a machine-readable and understandable format has already been scientifically investigated [49]. In this publication, the authors propose the ontology "Physci," which focuses on various aspects of physics research. This OWL ontology, which was published under the Creative Commons license, demonstrates that physical knowledge can be organized according to a given scheme. Figure 25 shows the core classes of the Physci ontology with their relationships to each other.

The ontology consists of several classes that model the basic concepts of physics research. The class "ReaserchWork" stands for the systematic activity that a person does to acquire knowledge about the physical world. This class is connected to the class "Publication", which represents the published research results. A research work deals with a scientific problem represented by the class "ScientificProblem." This "ScientificProblem" is solved by a "Solution," which can be developed using a "ScientificModel." The class "ScientificModel" has a class "Equation" assigned to it, representing the mathematical equations used in the model. It also uses the "ScientificMethod" class, which represents the methods used to solve scientific problems. These methods use the "Observation" class, which represents observed findings. The "Observation" class is linked to the "Measurement" class on one side and the "Observer" class on the other. The "Observer" class represents any measurement device used to make the observation. The observer is connected to the "Formation" class, which represents the formation of the observed object.

The authors of the Physci ontology have also created a prototype of a knowledge graph based on the ontology. It is mainly used to demonstrate the capabilities of the ontology. This knowledge graph focuses on the field of astronomy with a particular focus on black holes. It contains 193 individuals and 179 connections between them. This knowledge graph demonstrates that the ontology can be used to represent complex physical knowledge in a machine-readable format.

Even if the ontology is very powerful, it has some limitations. The most significant limitation is that the class definitions are not completely clear and are sometimes subject to ambiguity. Terms like "ScientificProblem" or "ScientificMethod" are not defined in a way that precisely clarifies what they mean. It is always possible to interpret the terms differently, so whether a concept falls into one of these categories is not always easy to answer. This is not a problem if the ontology is used by humans, but it can lead to problems if it is used for automatic data processing.

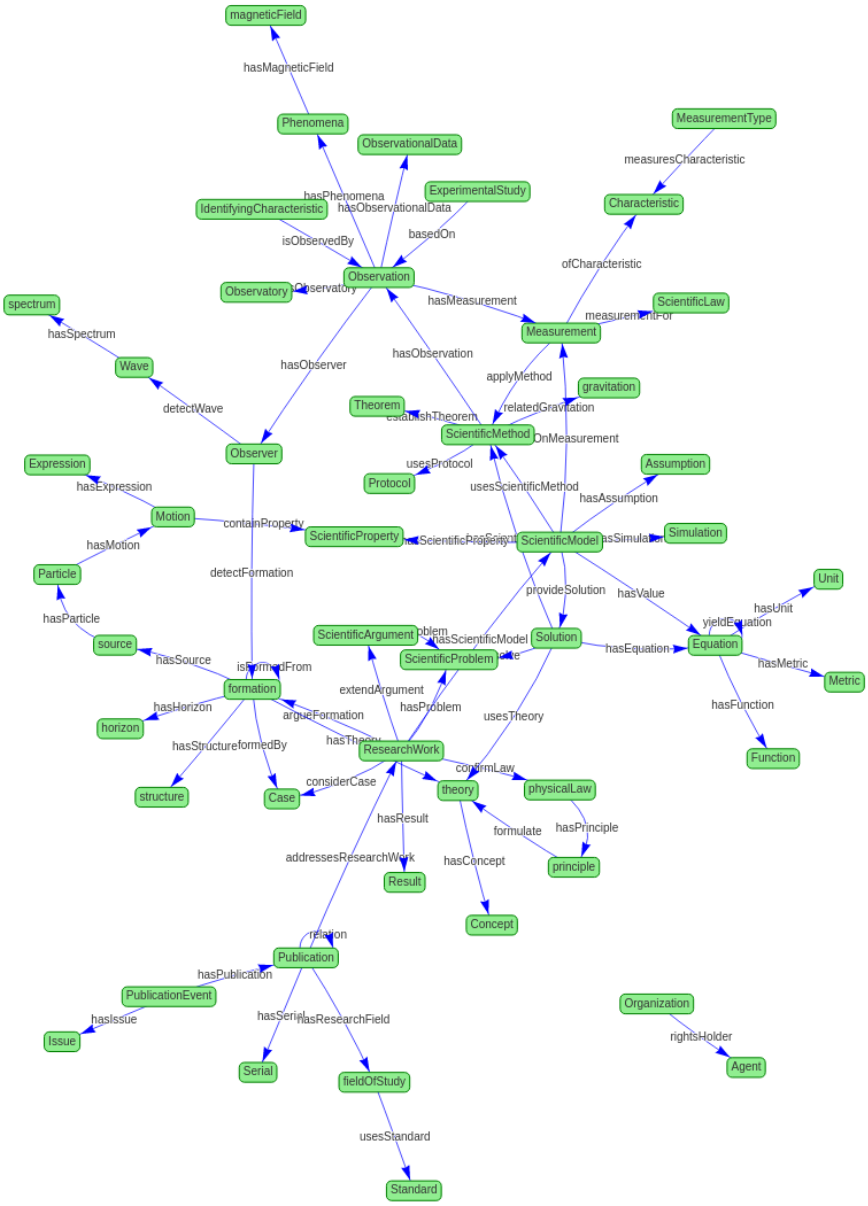



Figure 25: Core classes of the Physci ontology.

## 6.3. Building a customized ontology for machine learning in physics

The power of an ontology lies in its ability to specialize in a specific area of knowledge. By doing this, the ontology can be adapted to the requirements of the use case. This principle is tested on the area of top quark tagging with machine learning models. More specifically, the focus is on comparing the R30 value and the parameter count of different machine learning models extracted from scientific papers as described in section 5.2.

The resulting ontology is shown in the figure 26. The colored rectangles represent the classes of the ontology, while the black arrows represent the object properties that connect the classes. The orange rectangles are classes that have a data property assigned

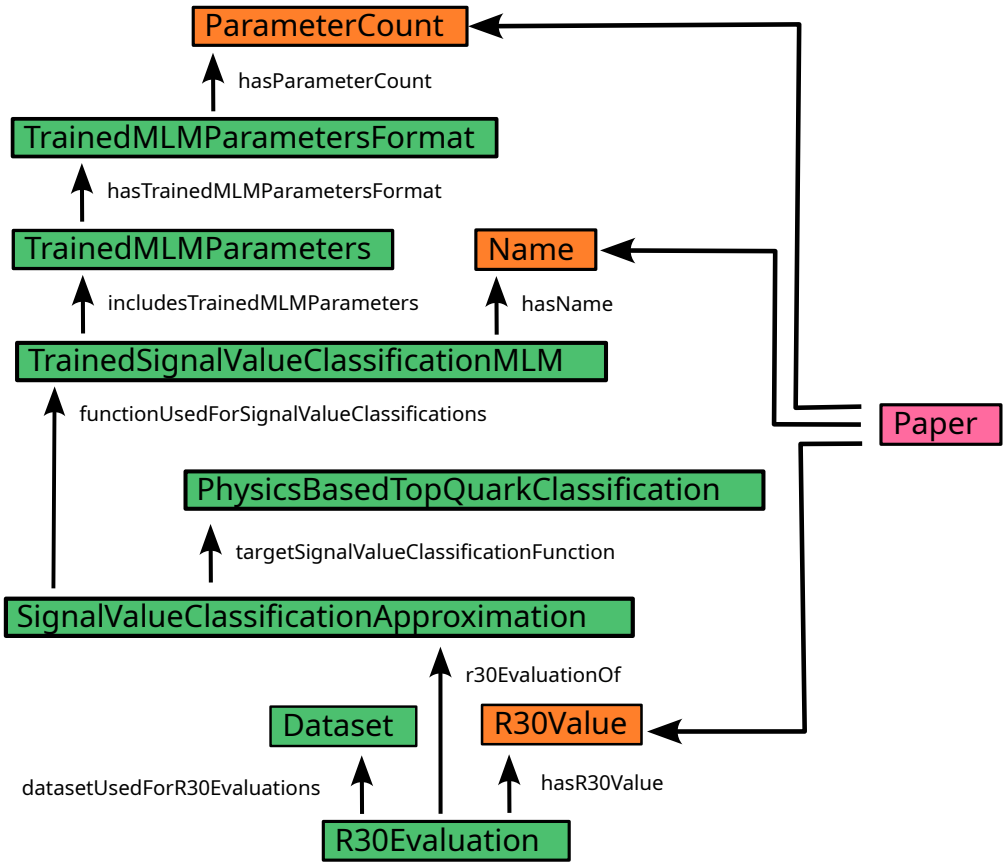



Figure 26: Ontology for the comparison of machine learning models in physics

to them. The different trained versions of the top tagging models are individuals of the class "*TrainedSignalValueClassificationMLM*". They can have a name assigned to it. To each of the models belongs a set of trained parameters that are represented by the class "*TrainedMLMParameters*". The trained parameters have a specific format in which the data is stored that has the parameter count as a property. Each trained signal value classification model is used as an approximation for a physics-based top quark classification. This is the classification that classifies all events based on physical considerations. The quality of this approximation is measured by a R30 evaluation performed on a test dataset. This evaluation results in a R30 value. The different properties of the machine learning models are mentioned in papers represented by the class "*Paper*".

The data of section 5.2 was converted into an RDF knowledge graph that is based on this ontology. This graph is composed of 1631 RDF triples, 386 individuals, and 603 connections between them. While in this simple case, storing the data as a semantic network does not bring a significant advantage over a simple table, the power of the semantic network becomes apparent when more complexity is added to the ontology. The ontology is designed to be extended with additional information about machine learning in physics. For example, the class "TrainedSignalValueClassificationMLM" could be seen as a subclass of the class "TrainedMachineLearningModel," which could be connected to the class "UntrainedMachineLearningModel" via a "MLMTrainingProcess" class. An ex-

tended version of the ontology could contain additional information about the training data, the training process, the hyperparameters of the machine learning models, and the scientists involved in the research. It could also link the datasets to the physical experiments that produced the data. In addition, different types of machine learning models could be included in the ontology, such as generative models, that can be used to simulate detector events. A knowledge graph based on such an ontology could be used for a comprehensive analysis of the performance of the machine learning models and the quality of the data they produce.

## 6.4. Automating the creation of a physics ontology

Next, it will be investigated whether the process of creating a physics ontology can also be automated using large language models. This is not trivial because designing appropriate classes and relations is a complex procedure that requires a lot of abstract thinking. However, it is possible to break down the process into smaller steps that can be automated. First, the language model `gpt-4-turbo` was queried to create a list of classes for the ontology. The model was not expected to create all classes at once. Instead, it was asked to extend an existing list of classes with new suggestions in multiple iterations. This approach potentially raises the quality of the generated classes because the model can concentrate on the classes one by one and use the existing classes as a reference. The query used to generate the classes for the ontology is shown in appendix B.

When no examples of classes were provided to the model, it generated classes like:

- "*QuantumMechanics*"
- "*ClassicalMechanics*"
- "*Thermodynamics*"
- "*Electromagnetism*"
- "*ParticlePhysics*"

These are not suitable classes for the ontology because they do not define a specific type of concept but only indicate general subject areas. Better results were achieved when the initial list of classes was seeded with a single class to show the model what kind of classes are expected. When the list is seeded with the class "ResearchArea", the following classes were generated:

- "*ResearchArea*"
- "*PhysicalTheory*"
- "*PhysicalLaw*"
- "*PhysicalExperiment*"

- "*MeasurementUnit*"
- "*PhysicalConcept*"

These classes are much better because they define specific types of concepts. The process of generating new classes was repeated multiple times until a list of 26 classes that could be used in the ontology was generated. The 26 classes are all good categories for individuals except for the class "*ElectromagneticSpectrum*", which can not have multiple instances because there exists only one electromagnetic spectrum.

The model was then asked to generate a list of relations that can be used to connect the classes. The query used to generate the relations is shown in appendix B. The following list shows an excerpt of the generated relations:

- $\langle$ *ResearchArea* | *focusesOnPhysicalConcept* | *PhysicalConcept* $\rangle$
- $\langle$ *PhysicalTheory* | *involvesPhysicalConstant* | *PhysicalConstant* $\rangle$
- $\langle$ *PhysicalTheory* | *describesPhysicalPhenomenon* | *PhysicalPhenomenon* $\rangle$
- $\langle$ *PhysicalExperiment* | *requiresScientificEquipment* | *ScientificEquipment* $\rangle$
- $\langle$ *PhysicalExperiment* | *verifiesPhysicalTheory* | *PhysicalTheory* $\rangle$

All relations in this example are valid object properties that can be used to connect the instances of the classes. The process of generating relations was repeated until a list of 39 relations that could be used in the ontology was generated. Of the 39 relations, 32 are useful, but 7 do not make much sense. For example, the relationship $\langle$ *PhysicalLaw* | *requiresMeasurementUnit* | *MeasurementUnit* $\rangle$ is not very useful because each physical law can be formulated in different units and is not dependent on a specific one. The generated ontology can be seen in the figure 27. Except for those 7 relations, it is a valid ontology that can be used to structure the knowledge of physics.

## 6.5. Automating the creation of a knowledge graph that is based on the ontology

Next, it was investigated whether the process of creating a knowledge graph that is based on the ontology can also be automated. To accomplish this, an algorithmic approach is introduced that can generate individuals of the classes and connect them via the allowed relations. Once again, the process is separated into two smaller steps that can be automated using the language model `gpt-4-turbo`. First, a list of named individuals is generated for each class. Then `gpt-4-turbo` was asked to connect the matching individuals by using the relations that are allowed by the ontology. The prompts that were used for these two steps are shown in appendix B.

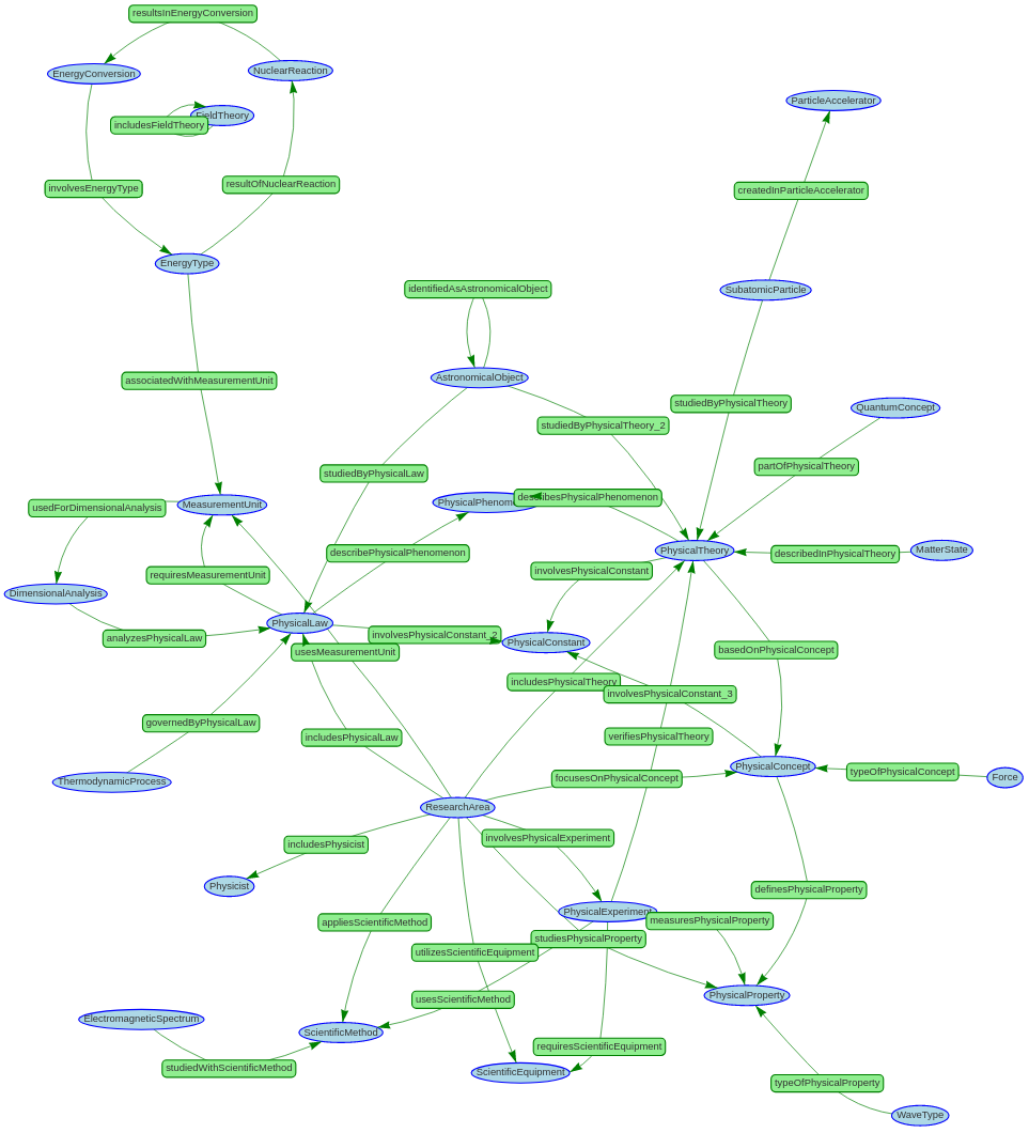



Figure 27: Ontology generated by `gpt-4-turbo`

Using this approach a knowledge graph was generated that is based on the ontology that was created in the previous step. The knowledge graph contains 156 individuals and 254 connections between them. The content of the knowledge graph is not very meaningful because the individuals available for connection were chosen arbitrarily. In some cases, the naming of the individuals that is used by `gpt-4-turbo` is unconventional. For example, the individuals of the class "PhysicalProperty" are named with the suffix "Measurement": "DensityMeasurement", "MassMeasurement", "PressureMeasurement" and so on. Also, not all parts of the knowledge graph are completely physics-related because some of the classes, like "ScientificEquipment" or "ResearchArea", are not directly related to physics.

Some relations provide a better way of connecting concepts in a meaningful way than others. While the object property "identifiedAsAstronomicalObject" that connects the "AstronomicalObject" class with itself is not very useful, the object property "requires-

ScientificEquipment" that connects the "PhysicalExperiment" class with the "ScientificEquipment" class can be used to model useful information about the required equipment for a specific experiment. In some cases, `gpt-4-turbo` has chosen to connect all possible combinations of individuals of two classes with a given relation. This can lead to a large number of connections that are not meaningful. For example, the object property "definesPhysicalProperty" connects all individuals of the "PhysicalConcept" class with all individuals of the "PhysicalProperty" class.

Even if the knowledge graph has some limitations, it also contains some useful information. The following list shows examples of useful triples that are contained in the knowledge graph:

- $\langle$ *SpecialRelativity* | *basedOnPhysicalConcept* | *LorentzInvariance* $\rangle$
- $\langle$ *KeplersLaws* | *describePhysicalPhenomenon* | *OrbitalMotion* $\rangle$
- $\langle$ *solarPanelInstallation* | *involvesEnergyType* | *SolarEnergy* $\rangle$
- $\langle$ *QuantumMechanics* | *involvesPhysicalConstant* | *PlancksConstant* $\rangle$
- $\langle$ *entanglement* | *partOfPhysicalTheory* | *QuantumMechanics* $\rangle$

These results demonstrate that it is possible to generate a knowledge graph that is based on an ontology using the language model `gpt-4-turbo`. The quality of the generated knowledge graph is not perfect, but it could be potentially improved by refining the ontology and the process of generating the individuals and connections.

# 7. Handling equations in semantic networks

This chapter examines how semantic networks can accommodate the peculiarities of physical knowledge by integrating structures to represent equations. It starts by describing a network design that fulfills this purpose. Next, the capabilities of large language models to generate such networks are tested. Then, it is investigated how derivations can be represented as semantic networks and how they can be checked for correctness. The chapter concludes with a discussion of the limitations of this approach and possible ways to overcome them.

## 7.1. Network of equations

The semantic networks that have been used so far use natural language to represent physical knowledge. However, some aspects of physics are usually expressed in the form of equations. Even if there are approaches to represent complicated equations as natural language [52], it is more convenient for humans to read them represented by mathematical symbols.

In order to take this fact into account when designing the semantic network, a design is proposed that incorporates equations as nodes into the network. Figure 28 shows an example of the Planck relation displayed as a semantic network of equations.

The design of the network allows two types of nodes: Natural language concepts such as "Energy of the Photon" and mathematical terms such as "$\hbar\omega$." For these two types of nodes, two relations are possible:

- The "`equals`" relation that can connect two mathematical terms with each other and indicates the equality between them.
- The "`where <symbol> is`" relation which indicates that a specific symbol of the equation is representing the connected natural language concept.

## 7.2. Technical realization of using equations in semantic networks

Handling equations in semantic networks is not entirely trivial because the software tools that handle network data often support only plain `utf-8` text. To represent the formula internally as bytes, the LaTeX syntax [53] was used together with a marker that indicates

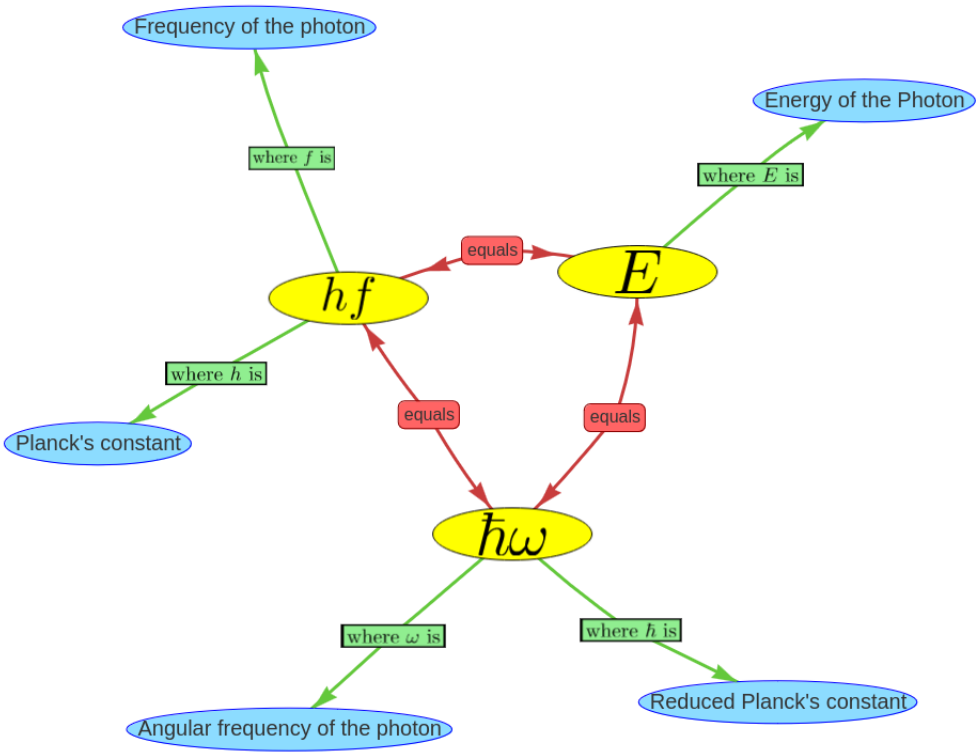



Figure 28: Example of a semantic network of equations

that the byte sequence should be interpreted as an equation. For example, the equation $E = \hbar\omega$ would be handled as "`f: E = \hbar \omega`."

For the visualization of the example above, the tool `vis.js` was used that can display interactive graphs in the web browser. This tool does not support inline equations. The solution for this problem was to convert the equation into `svg` [54] using the python library `latextools` [55]. This `svg` graphics could then be included in the `vis.js` visualization.

## 7.3. Generating networks of equations with large language models

The question arises of whether it is also possible to use large language models to generate such networks of equations for a specific physics topic. This would open up many possibilities to automatically generate physics knowledge with a stronger mathematical focus. To accomplish this, an approach was used to let `gpt-4-turbo` generate an equation together with the corresponding symbol explanations. The approach uses the following prompt that also contains the example of the formula for the energy of a photon:

**Prompt for generating a network of equations for a given topic**

```
Example of a formula for the Energy of a Photon:

[
 "$E = \hbar \omega$ = h f",
 {
 "$E$": "Energy of the Photon",
 "$\hbar$": "Reduced Planck's constant",
 "$\omega$": "Angular frequency of the photon",
 "$h$": "Planck's constant",
 "$f$": "Frequency of the photon"
 }
]

Give me a formula for <topic> with the explanations of all
symbols formatted in the same way.  The explanations of the
symbols should be concepts written in a compact form.  Re-
turn nothing but the result [formula, symbol explanations]
```

In this query, <topic> is replaced by a physics topic that should be displayed as a network of equations. The answer to this query is then broken down into its components. This is done using the Python JSON parser. The formula is then split up at the equals sign into its expressions. Next, an algorithm tailored to the LaTeX syntax determines which of the explained symbols appear in which expressions. It is not enough to just search in the expression for the LaTeX string of the symbol because this would lead to false positive results. For example in the formula for the acceleration "`a = \frac{\Delta v}{\Delta t}`" "a" appears only on the left side of the equation but would also be found in the right side of the equation by a naive algorithm that interprets the "a" in "`\frac`" as the symbol for acceleration. The algorithm used here may not correctly detect all possible cases, but it works for the most commonly used formula signs.

The implemented program can generate networks of equations for a given topic. Figure 29 shows a network of equations that is generated for the topic "Work done by a Carnot engine" by `gpt-4-turbo`. This is the correct formula for the ideal Carnot engine. But this example also shows that `gpt-4-turbo` is not always perfectly reliable. It made a typo in the explanation of $W$ and wrote **Carnof engine** instead of **Carnot engine**.

Not all the generated formulas of `gpt-4-turbo` are correct. An example of an incorrect formula is shown in the figure 30. In this example, the task was to generate a formula for the energy a black body radiates. The created formula is incorrect because the expression $\sigma A T^4$ is not equal to the energy $E$ but to the radiated power $\phi$. The correct formula would be $\sigma A T^4 t = E$ where $t$ is the time span for which the radiated energy of the black body is measured.

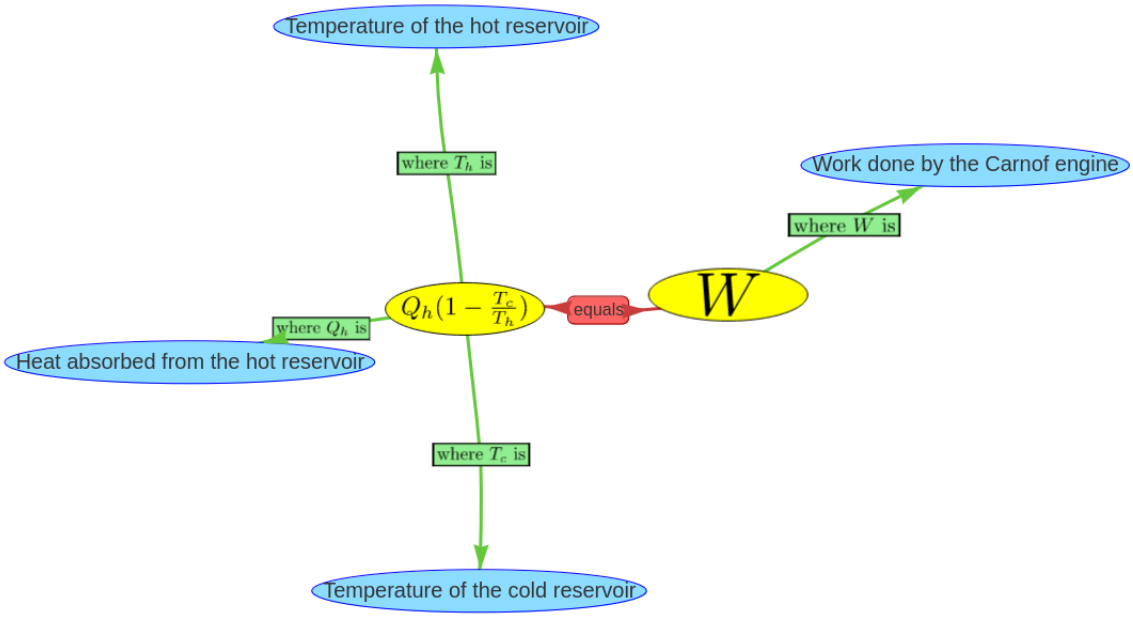



Figure 29: Network of equations for the topic "Work done by a Carnot engine".

Out of 20 tested equations from different areas of physics, four equations in total were incorrect. In addition to the errors in the black body radiation formula, `gpt-4-turbo` also forgot some brackets in the fourth Maxwell equation, did not produce the correct Tsiolkovsky rocket equation, and provided an incorrect explanation for the variable $c$ in the equation for the Doppler effect (the speed of light instead of the speed of sound).

## 7.4. Generation of networks containing multiple equations

The previous examples show isolated equations. In order to test whether generating networks that contain multiple equations is also possible, a new algorithm was developed. This algorithm starts with a physics term like "Elementary charge" and asks `gpt-4-turbo` to return a randomly selected physics equation that contains a symbol corresponding to that term. In the next step, one of the explanation terms of the returned formula is chosen, and `gpt-4-turbo` is asked to generate an equation that contains this new term as an explanation. This procedure is repeated, and identical formula explanation terms are merged.

Figure 31 shows a network of equations that was generated in this way. This network combines equations from different topics, such as gravitational force, Coulomb force, and kinetic energy, into one graph that is joined by common quantities like mass and energy. This demonstrates that in general, `gpt-4-turbo` can generate interdisciplinary equation networks, even if previous results have shown that the equations are not always completely reliable.

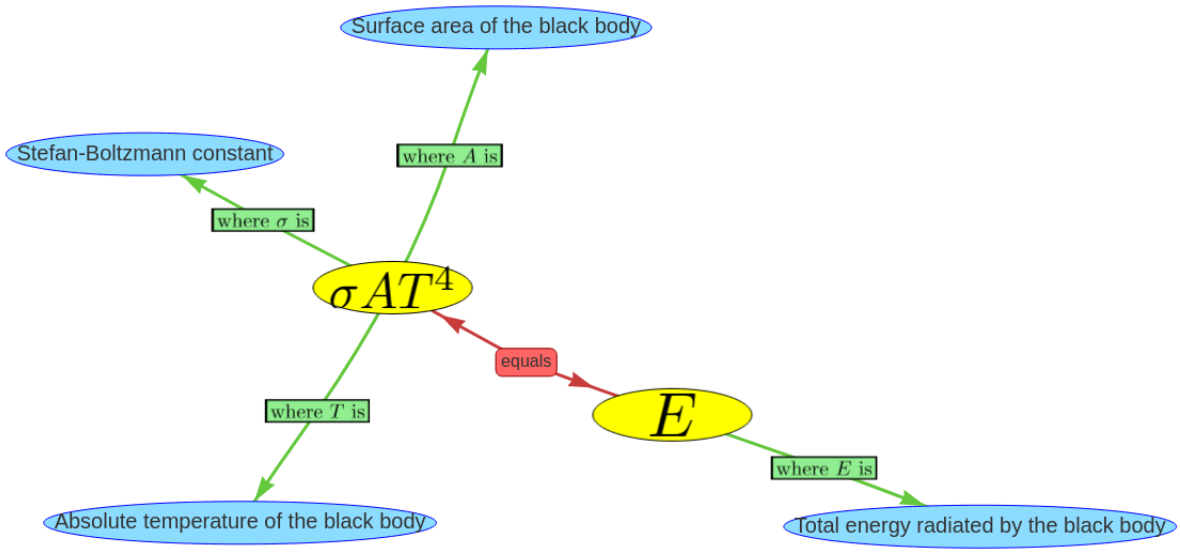



Figure 30: Network of equations for the topic "Energy radiated by a black body".

## 7.5. Representing a derivation as a semantic network

A central aspect of physical science is deriving new conclusions from certain basic assumptions using the rules of logic and mathematics. This section examines whether it is possible to represent such derivations as a semantic network and the difficulties associated with doing so.

### 7.5.1. Using a network of equations to represent derivations

The networks of equations from the previous section are now extended by a new connection type that represents the deduction of one equation from another. The "from this follows that" connection node can connect multiple input equations, which can be recognized by the arrows pointing from the equation to the node, and multiple output equations, which can be recognized by the arrows pointing from the node to the equation. This indicates that the output equations can be derived from the input equations.

Figure 32 shows the graph representation of the solution to the free fall differential equation. The symbol explanations have been omitted for clarity. The deduction process is organized in two steps. First, the two equations $F = gm$ and $F = m\frac{d^2x}{dt^2}$ are combined into one differential equation, and in the second step, this differential equation is solved by the solution $x(t) = x_0 + v_0 t + \frac{1}{2}gt^2$.

### 7.5.2. Representing a complex derivation as a network of equations

Figure 33 shows the derivation of the general solution for the one-dimensional Schrödinger equation of a particle in the time-independent potential. Again, the symbol explanations have been omitted. The derivation process starts with the Schrödinger equation.

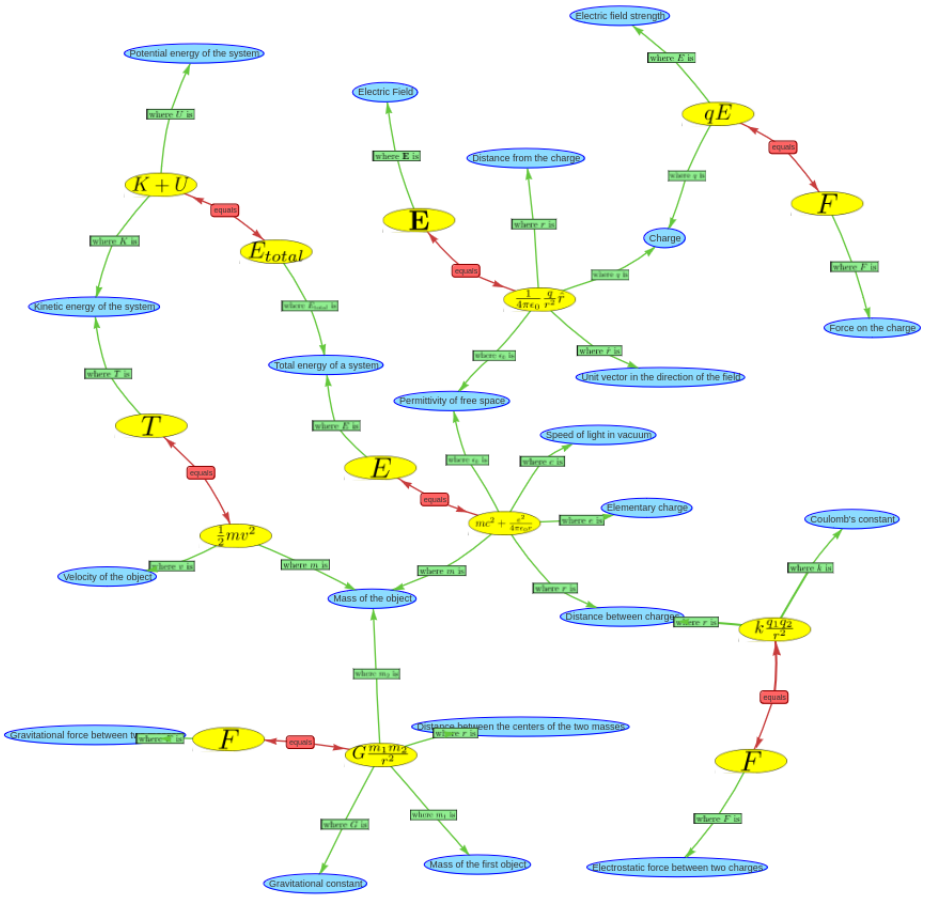



Figure 31: Network of equations generated by `gpt-4-turbo`.

$$i\hbar\frac{\partial}{\partial t}\psi(x,t) = -\frac{\hbar^2}{2m}\Delta\psi(x,t) + V(x)\psi(x,t) \tag{7.1}$$

It follows the approach of constructing the wave function as a product of two functions with separated variables for time and place:

$$\psi(x,t) = \varphi(x)\chi(t) \tag{7.2}$$

It then derives two equations for the space and the time, which are both equal to a newly defined constant $E$, and sets this constant to $\hbar\omega$:

$$\frac{i\hbar}{\chi(t)}\frac{d\chi(t)}{dt} = E = \hbar\omega = -\frac{\hbar^2}{2m\varphi(x)}\Delta\varphi(x) + V(x) \tag{7.3}$$

It then solves the differential equation of $\chi$ and introduces a new constant $A$:

$$\chi(t) = Ae^{-i\omega t} \tag{7.4}$$

It also observes the linearity of the Schrödinger equation and states that a sum of solu-

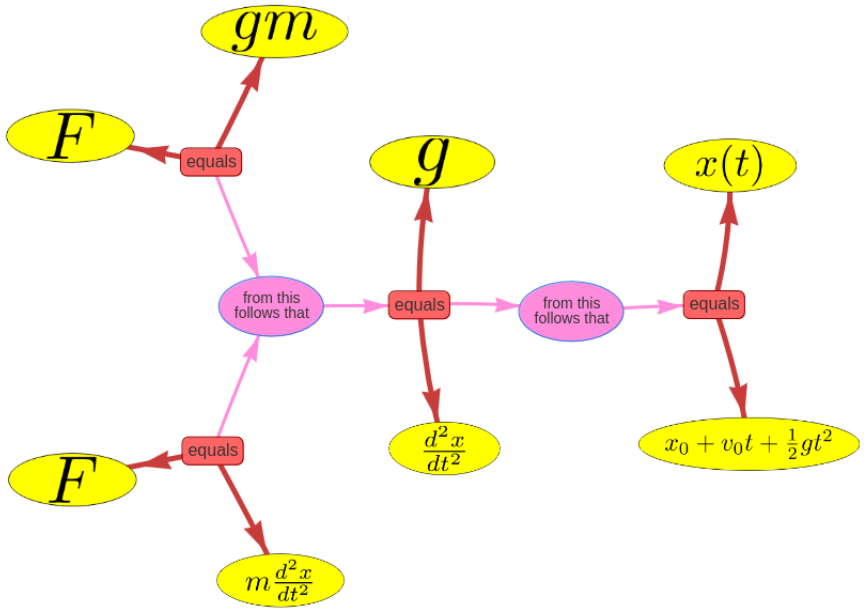



Figure 32: A semantic network representing the derivation of the solution to the free fall differential equation.

tions is also a solution of the Schrödinger equation:

$$\psi(x,t) = \sum_n \psi_n(x,t) \tag{7.5}$$

Finally, it summarizes the results into the general solution:

$$\psi(x,t) = \sum_n c_n \varphi_n(x) e^{-i\omega_n t} \tag{7.6}$$

### 7.5.3. Automated proof checking

Next, the question should be investigated whether it is possible to perform an automated proof-checking algorithm on the semantic network of equations. This can be done using a large language model like `gpt-4-turbo`. Testing the ability of large language models to solve mathematical equations is beyond the scope of this master's thesis. See [56] for a detailed analysis of the mathematical problem-solving capabilities of large language models. Because these models are not perfectly reliable, one cannot wholly trust their assessment of the correctness of derivations. However, the method can find potential errors in the derivation.

The network of equations from above was tested for correctness by asking `gpt-4-turbo` if the different reasoning steps are correct. For comparison, another version of the network with slightly altered equations was tested using the same method. The equations of the second network were changed so that the reasoning of the derivation is no longer correct.

From the five slightly altered reasoning steps, `gpt-4-turbo` correctly identified all

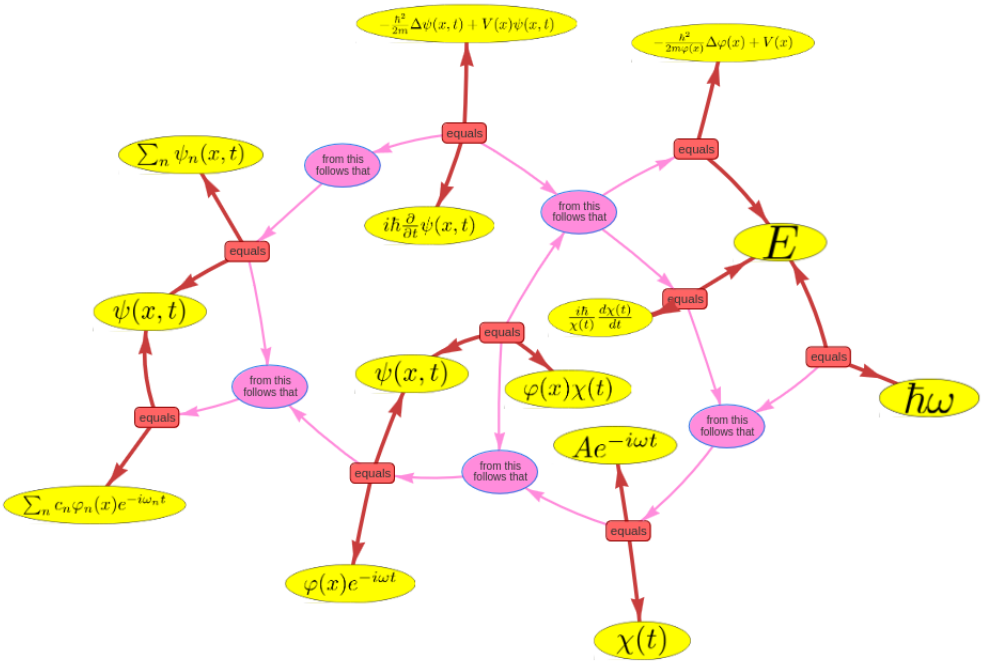



Figure 33: A semantic network representing the derivation of the general solution for the one-dimensional Schrödinger equation of a particle in a time-independent potential.

as false. From the five original reasoning steps of the derivation, `gpt-4-turbo` identified all except for one as correct. This one reasoning step is that a sum of solutions to the Schrödinger equation is also a solution which is derived from the linearity of the Schrödinger equation. The reason for this could be that the following notation is not completely clear because it does not explain that the $\psi_n$ must also be solutions to the Schrödinger equation.

$$i\hbar\frac{\partial}{\partial t}\psi(x,t) = -\frac{\hbar^2}{2m}\Delta\psi(x,t) + V(x)\psi(x,t) \Rightarrow \psi(x,t) = \sum_n \psi_n(x,t)$$

The information about the definition of the $\psi_n$ is encoded in the symbol explanations, which were not included in this evaluation process. This shows that the representation by mathematical symbols alone is often not sufficient to understand the derivation. Some of the information such as that a certain symbol is an arbitrarily selectable constant can only be understood by considering the symbol explanations. In the current implementation, the model guesses the meaning of the symbols. Some more advanced versions of the proof-checking algorithm could also take the symbol explanations into account.

### 7.5.4. The limitations of networks of equations

The results in the previous section show that it is possible to display some physical derivations as semantic networks. What remains to be answered is whether this approach

can represent all derivations in physics. What speaks against it is that many derivations involve not only formulas but also natural language reasoning or drawn sketches, which are used to clarify contextual relationships. These are both hard to incorporate into the semantic network. A simple example of a derivation that can not be easily converted into a semantic network of equations is the derivation of Snell's law from the properties of waves. Snell's law states that at a transition between two media, there is a relationship between the angle of incidence and emergence of a wave and the velocities of the wave, which is described by the following equation:

$$\frac{\sin \Theta_1}{\sin \Theta_2} = \frac{v_1}{v_2} \tag{7.7}$$

Here, $\Theta_1$ and $\Theta_2$ are the angles of incidence and emergence of the wave, while $v_1$ and $v_2$ are the speed of the wave in the first and the second medium. Snell's law can be derived from some geometrical calculations based on the assumption that a wave always moves at right angles to its wavefront. Figure 34 shows a sketch of the geometrical considerations that can be used to derive Snell's law. This consideration and geometrical

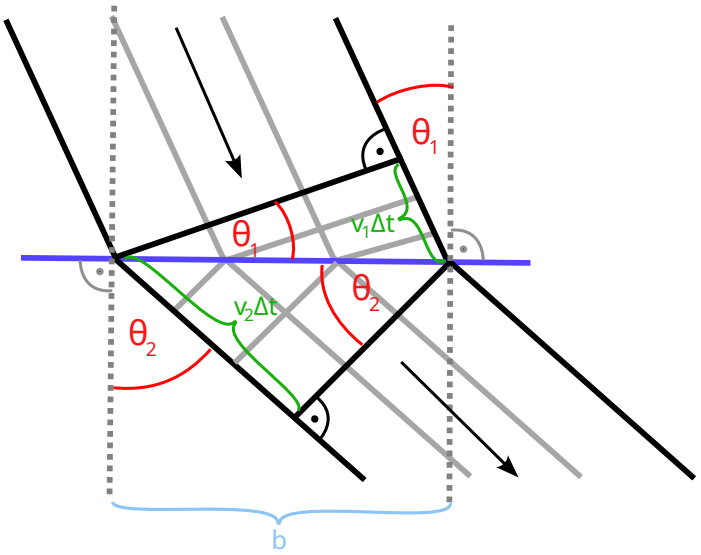



Figure 34: Geometrical considerations for the derivation of Snell's law. The light wave hits the interface between two media at an angle $\Theta_1$ and is refracted at an angle $\Theta_2$. $v_1$ and $v_2$ are the speeds of the wave in the first and the second medium. $b$ is the length of the section on which the wave crosses the interface, and $\Delta t$ is the length of the time interval between the wavefront hitting the beginning and the end of this section.

description of the problem can not be expressed solely by a semantic network of equations.

### 7.5.5. Combining a network of equations with a network of sentences

To solve the problem of representing derivations that can not be expressed solely by equations as a semantic network, one could use the expressiveness of the semantic network

of sentences described in section 5.4 to extend the network of equations. This requires describing the sketch using words. To test this approach a network of sentences with integrated equations was created that represents the derivation of Snell's law. This network describes the geometric properties by naming the catheti and hypotenuses of the triangles formed by the incoming and outgoing waves. It then derives the equations $\sin\Theta_1 = \frac{v_1\Delta t}{b}$ and $\sin\Theta_2 = \frac{v_2\Delta t}{b}$ from the description of the triangles. It therefore uses the hypotenuse $b$ and the opposite $d$ where $d = v\Delta t$. These results are used to derive the equation $\frac{\sin\Theta_1}{\sin\Theta_2} = \frac{v_1}{v_2}$ in a final step. The structure of this network is shown in figure 35. In this figure, the network of sentences that occupies the bigger left part of the image can

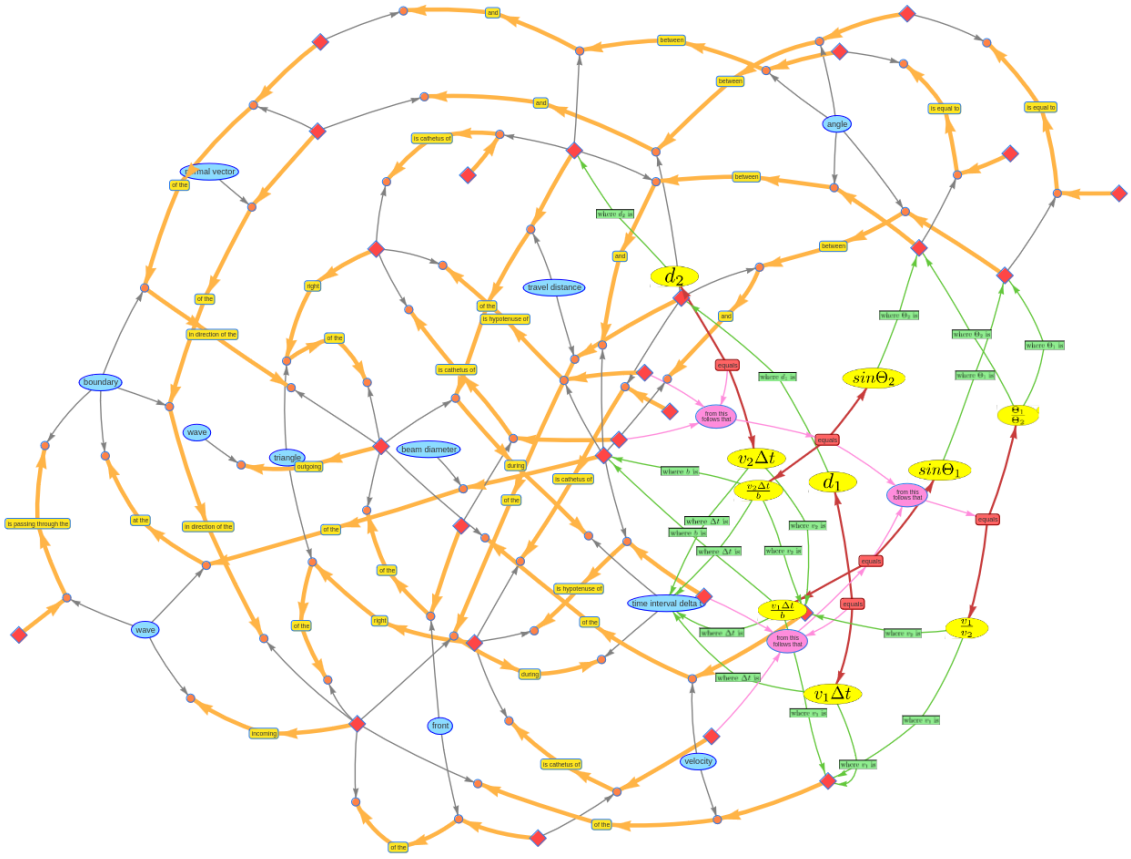



Figure 35: A semantic network representing the derivation of Snell's law.

be distinguished from the network of equations that occupies the smaller right part of the image. Both parts are connected to form a large network that consists of 26 sentence fragments, 11 terms and 5 equations. Even if the network is not easy to read without using further improved visualization tools, it contains all the information needed to understand the derivation of Snell's law. This example shows that a derivation that can be described by words and formulas can also be represented by a semantic network.

# 8. Summary and outlook

## 8.1. Summary

In this thesis, multiple different approaches to representing physics as a semantic network have been explored:

- Citation graphs were used to explore the landscape of scientific publications.
- Subtopic networks were used to structure the different subject areas of physics.
- Networks of semantic triples were used to represent general physical knowledge.
- Semantic networks of questions and answers were used to represent the process of answering physical questions.
- Semantic networks of sentences were used to capture the content of physical texts.
- Ontologies were used to represent the structure of physical knowledge.
- Semantic networks of equations were used to derive physical equations.

All of these approaches fulfill their respective purposes and can be used to represent physics in a structured way. It could be shown that large language models can generate semantic network data, which can provide insights into their internal understanding of physics. Different metrics were used to analyze the quality of the generated data and revealed that `gpt-4-turbo` outperforms the other tested models and can produce high-quality semantic network data. It was thus discovered that the meaningfulness of the generated physics facts is highest when the models can freely associate the content of the semantic network data and are not restricted by specific instructions. The free association algorithm also outperformed the subdivision algorithm in generating subtopic trees.

The best method to represent general physical knowledge was found to be the semantic network of sentences, which can capture more of the content of physical texts than the method of using semantic triples. To do justice to the representation of equations, a semantic network of equations can be integrated into the semantic network of sentences to further enhance its capabilities.

## 8.2. Outlook

The methods presented in this thesis can be further developed and used for a variety of applications:

**Physics education:** The benefits of semantic networks for organizing and structuring knowledge could be used to help students learn physics. A visual representation of subtopic networks could give students a better overview of the subject areas of physics and help them decide which topics to focus on. The development of interactive tools that allow students to navigate a physics knowledge base and explore the relationships between different concepts could also be helpful for learning physics. In contrast to traditional textbooks that present knowledge as a linear sequence of topics, this approach would allow students to focus on specific concepts and find all relevant information about them linked in one place.

**Scientific publication and data exchange:** Semantic networks could also be used to build a knowledge base maintained by the scientific community. This knowledge base could be used to store and exchange scientific data in a structured way. However, this would require further refinement and standardization of the methods used to represent physics data as semantic networks.

**Knowledge extraction from papers:** Large language models can help extract physics knowledge from scientific papers and represent it as semantic networks. Therefore, multiple papers could be combined to form a large knowledge base. The information contained in this knowledge base could then be queried for specific statements. For each of these statements, metadata could be available indicating the source of the statement.

**Reasoning and problem solving:** Semantic networks could also play a role in the development of new AI systems that can reason about physics problems. It has been shown that the semantic network of questions and answers can be used to isolate statements that are relevant to a specific question. The semantic network of equations has proven helpful for checking the consistency of derivations. These techniques could be used to create new AI algorithms that are able to derive new conclusions step by step from a given set of premises.

**Exploring the physics knowledge bases of large language models:** Letting large language models generate semantic network data could also be used to get an overview of their internal understanding of physics. The techniques described in this thesis could be used to search for contradictions in their knowledge base or to identify areas where their knowledge is incomplete. This could help to improve the performance of future models in physics-related tasks.

**Exploring the landscape of physics knowledge:** By using tools like citation graphs and subtopic networks, it is possible to explore the different subject areas of physics and the relationships between them. This could help to identify promising research directions and to guide future research efforts.

## 8.3. Conclusion

In conclusion, the methods presented in this thesis provide a powerful toolbox for representing physics as semantic networks. By using large language models to generate semantic network data, it is possible to get revealing insights into their internal understanding of physics. Semantic networks have the potential to fundamentally change the representation of physics knowledge and can be used to build knowledge bases that both humans and machines can understand.

# Bibliography


[1] Christopher Woods. "The earliest Mesopotamian writing". In: *Visible language. Inventions of writing in the ancient Middle East and beyond*. Ed. by Christopher Woods. Oriental Institute Museum Publications Number 32. Chicago, Illinois: The Oriental Institute of the University of Chicago, 2010, pp. 33–84.

[2] Johann Christoph Gottsched. *Gedächtnissrede auf den unsterblich verdienten Domherrn in Frauenberg Nicolaus Copernicus, als den Erfinder des wahren Weltbaues: welche... auf der Universitätsbibliothek zu Leipzig... zweyhundert Jahre nach seinem Tode, gehalten worden*. bey Bernhard Christoph Breitkopf, 1748.

[3] Aileen Fyfe et al. *A history of scientific journals*. UCL Press, 2022.

[4] Roberto Lalli. "A brief history of physics reviews". In: *Nature Reviews Physics* 1.1 (Jan. 2019), pp. 12–14.

[5] Tim Berners-Lee. "Long live the web". In: *Scientific American* 303.6 (2010), pp. 80–85.

[6] Stella Christodoulaki. "Rebuilding INSPIRE together with the HEP community". In: *EPJ Web of Conferences*. Vol. 245. EDP Sciences. 2020, p. 08012.

[7] Ashish Vaswani et al. *Attention Is All You Need*. June 2017. arXiv: 1706.03762 [cs.CL].

[8] Zhengde Zhang et al. *Xiwu: A Basis Flexible and Learnable LLM for High Energy Physics*. 2024. arXiv: 2404.08001 [hep-ph].

[9] Tijmen de Haan. *cosmosage: A Natural-Language Assistant for Cosmologists*. 2024. arXiv: 2407.04420 [astro-ph.IM].

[10] Matthew Hutson. "How does ChatGPT'think'? Psychology and neuroscience crack open AI large language models". In: *Nature* 629.8014 (2024), pp. 986–988.

[11] Fritz Lehmann. "Semantic networks". In: *Computers & Mathematics with Applications* 23.2-5 (1992), pp. 1–50.

[12] John F Sowa et al. "Semantic networks". In: *Encyclopedia of artificial intelligence* 2 (1992), pp. 1493–1511.

[13] Roger T Hartley and John A Barnden. "Semantic networks: visualizations of knowledge". In: *Trends in Cognitive Sciences* 1.5 (1997), pp. 169–175.

[14] *Neo4j*. Accessed: 2024-08-28. URL: https://neo4j.com/.

[15] Nadime Francis et al. “Cypher: An evolving query language for property graphs”. In: *Proceedings of the 2018 international conference on management of data*. 2018, pp. 1433–1445.

[16] Jeff Z. Pan. “Resource Description Framework”. In: *Handbook on Ontologies*. Ed. by Steffen Staab and Rudi Studer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 71–90.

[17] N Yadagiri and P Ramesh. “Semantic web and the libraries: an overview”. In: *International journal of library science* 7.1 (2013), pp. 80–94.

[18] James McCusker et al. *What is a Knowledge Graph?* Accessed: 2024-09-04. URL: https://www.authorea.com/users/6341/articles/107281.

[19] Alan Isaacs. *Oxford dictionary of physics*. 1996.

[20] *A Dictionary of Physics*. Accessed: 2024-08-08. URL: https://www.oxfordreference.com/display/10.1093/acref/9780199233991.001.0001/acref-9780199233991.

[21] Martin Majlis. *Wikipedia API*. Accessed: 2024-08-08. URL: https://wikipedia-api.readthedocs.io/en/latest/README.html.

[22] *GPT-3.5 Turbo*. Accessed: 2024-08-08. URL: https://platform.openai.com/docs/models/gpt-3-5-turbo.

[23] *GPT-4 Turbo*. Accessed: 2024-08-08. URL: https://platform.openai.com/docs/models/gpt-4-turbo-and-gpt-4.

[24] Albert Q. Jiang et al. *Mistral 7B*. 2023. arXiv: 2310.06825 [cs.CL].

[25] Abhimanyu Dubey et al. *The Llama 3 Herd of Models*. 2024. arXiv: 2407.21783 [cs.AI].

[26] Alan H. Guth. *The inflationary universe: The quest for a new theory of Cosmic Origins*. New York: Basic Books, 1997.

[27] David Wands. “Multiple Field Inflation”. In: *Inflationary Cosmology*. Ed. by Martin Lemoine, Jerome Martin, and Patrick Peter. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 275–304.

[28] David H Lyth and David Wands. “Generating the curvature perturbation without an inflaton”. In: *Physics Letters B* 524.1–2 (Jan. 2002), pp. 5–14.

[29] Chuangchuang Chen, Honggang Gu, and Shiyuan Liu. “Ultra-simplified diffraction-based computational spectrometer”. In: *Light: Science & Applications* 13.1 (Jan. 2024), p. 9.

[30] Bernard Henin. “The Frost Line”. In: *Exploring the Ocean Worlds of Our Solar System*. Cham: Springer International Publishing, 2018, pp. 21–31.

[31] F. Braga-Ribas et al. “A ring system detected around the Centaur (10199) Chariklo”. In: *Nature* 508.7494 (Apr. 2014), pp. 72–75.

[32] *vis.js*. Accessed: 2024-08-15. URL: https://visjs.org/.

[33] Andrew D. Gilbert. "Chapter 9 - Dynamo Theory". In: ed. by S. Friedlander and D. Serre. Vol. 2. Handbook of Mathematical Fluid Dynamics. North-Holland, 2003, pp. 355–441.

[34] Tim Bray. *The javascript object notation (json) data interchange format*. Tech. rep. 2014.

[35] *Ideales Gas — Wikipedia, die freie Enzyklopädie*. Accessed: 2024-07-24. URL: https://de.wikipedia.org/w/index.php?title=Ideales_Gas&oldid=241403574.

[36] *Ideal gas — Wikipedia, The Free Encyclopedia*. Accessed: 2024-07-24. URL: https://en.wikipedia.org/w/index.php?title=Ideal_gas&oldid=1225161812.

[37] Rajesh Chandrakar. "Digital object identifier system: an overview". In: *The Electronic Library* 24.4 (2006), pp. 445–452.

[38] David Matthews. "Drowning in the literature? These smart software tools can help." In: *Nature* 597.7874 (2021), pp. 141–142.

[39] Waleed Ammar et al. *Construction of the literature graph in semantic scholar*. 2018. arXiv: 1805.02262.

[40] HEP ML Community. *A Living Review of Machine Learning for Particle Physics*. Accessed: 2023-06-16. URL: https://iml-wg.github.io/HEPML-LivingReview/.

[41] *Beautiful Soup*. Accessed: 2024-08-28. URL: https://beautiful-soup-4.readthedocs.io/en/latest/#.

[42] *Inspire HEP*. Accessed: 2024-08-28. URL: https://inspirehep.net/.

[43] Luke de Oliveira et al. "Jet-images—deep learning edition". In: *Journal of High Energy Physics* 2016.7 (2016), pp. 1–32.

[44] Matthew Feickert and Benjamin Nachman. *A living review of machine learning for particle physics*. 2021. arXiv: 2102.02770.

[45] Gregor Kasieczka et al. "The Machine Learning landscape of top taggers". In: *SciPost Physics* 7.1 (July 2019).

[46] Ranit Das, Gregor Kasieczka, and David Shih. *Feature Selection with Distance Correlation*. 2022. arXiv: 2212.00046 [hep-ph].

[47] *Electron — Wikipedia, The Free Encyclopedia*. Accessed: 2024-08-23. URL: https://en.wikipedia.org/w/index.php?title=Electron&oldid=1235974061.

[48] *Natural Language Toolkit*. Accessed: 2024-09-09. URL: https://www.nltk.org/.

[49] Aysegul Say et al. *Semantic representation of physics research data*. Vol. 2. [Setúbal]: SCITEPRESS-Science and Technology Publications, Lda., 2020.

[50] Nicola Guarino, Daniel Oberle, and Steffen Staab. "What Is an Ontology?" In: *Handbook on Ontologies*. Ed. by Steffen Staab and Rudi Studer. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 1–17.

[51] Liyang Yu. "OWL: Web Ontology Language". In: *A Developer's Guide to the Semantic Web*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 169–263.

[52] Seon Yang and Youngjoong Ko. "Mathematical formula search using natural language queries". In: *Advances in Electrical and Computer Engineering* 14.4 (2014), pp. 99–104.

[53] *LaTeX*. Accessed: 2024-10-03. URL: https://www.latex-project.org/.

[54] *Scalable Vector Graphics*. Accessed: 2024-09-10. URL: https://www.w3.org/Graphics/SVG/.

[55] *LaTeX Tools*. Accessed: 2024-09-10. URL: https://github.com/cduck/latextools.

[56] Meng Fang et al. *MathOdyssey: Benchmarking Mathematical Problem-Solving Skills in Large Language Models Using Odyssey Math Data*. 2024. arXiv: 2406.18321 [cs.CL]. URL: https://arxiv.org/abs/2406.18321.

# Appendices

# A. Examples of incorrect triples from the four different categories

**`gpt-3.5-turbo` one-third:** An incorrect triple of this dataset is ⟨ *Elementary particle* | *is a* | *D meson* ⟩. A D-meson is not an elementary (indivisible) particle because it is composed of a charm quark and a lighter quark. A better choice for a predicate would have been "*is contained in*" instead of "*is a*".

**`gpt-4-turbo` one-third:** An incorrect triple of this dataset is ⟨ *Centripetal force* | *is considered a* | *Fictitious force* ⟩. In fact, the centripetal force, which causes a mass to move in a curved path around a central point, is a real force. Instead, the centrifugal force that seems to accelerate an object away from the center of a rotated frame is a fictitious force. A correct predicate choice would have been "*is not a*" instead of "*is considered a*".

**`gpt-3.5-turbo` two-thirds:** The incorrect triple of this dataset is ⟨ *Harmonic oscillator* | *is a type of* | *oscillatory motion* ⟩. This triple is false because the harmonic oscillator is not itself the motion. Instead, it is the structure that performs it. For this reason, a true version of this triple would have been ⟨ *Harmonic oscillator* | *performes a type of* | *oscillatory motion* ⟩.

**`gpt-4-turbo` two-thirds:** The incorrect triple of this dataset is ⟨ *Stack effect* | *increases* | *indoor air pressure at higher elevations* ⟩. When a building has no stack, the indoor pressure equalizes to the external atmospheric pressure through the leaks in windows, doors, and walls. When a chimney is added to the building, a column of warm air connects the indoor volume with the higher elevated outdoor air, which has a lower pressure than on the ground level. In hydrostatics, the change of the pressure $p$ by the height $\mathsf{h}$ can be calculated from the density $\rho$ and the gravitational acceleration $g$.

$$\frac{\partial p}{\partial \mathsf{h}} = -\rho g \tag{A.1}$$

Because the density of the warm air in the chimney is lower than that of the cold air outside, the pressure distance between the upper and lower end of the chimney is smaller than for the same vertical distance outside. This results in the indoor air being sucked in by the chimney. Thereby, the indoor pressure decreases. A correct triple would be ⟨ *Stack effect* | *decreases* | *indoor air pressure* ⟩.

# B. Prompts for generating ontologies and individuals

**Prompt for generating ontology classes**

I want to build an ontology about physics. Therefore, I need to create a list of owl classes. I already have the owl classes <list of existing classes>. What could be 5 additional owl classes that I could use in my ontology? Return a list of 5 owl classes formatted as follows: ["name of the first class", "name of the second class", ...] Write the names in camel case and return nothing but this list.

**Prompt for generating relations between ontology classes**

I want to build an ontology. I am using the following owl classes: <numbered classes list>. What relations could exist between instances of these classes? Return a list of possible relations in the following format: [[<number of the subject class>, "name of the first relation", <number of the object class>], [<number of the subject class>, "name of the second relation", <number of the object class>], ... ] The relation names should be in camel case. Return nothing but this list.

**Prompt for generating individuals of ontology classes**

I want to build a knowledge graph, that is based on an ontology. Therefore, I want to collect instances of the class <class name>. I want to add <number of instances> new instances of this class. The instances should have unique names that are written in camel case. Return the names of the new instances in the format ["first instance name", "second instance name", ...]. Return nothing but this list.

**Prompt for generating connections between individuals of ontology classes**

I want to build a knowledge graph, that is based on an ontology. Therefore I want to connect instances of the classes `<subject class>` and `<object class>` with the relation `<relation name>`. The individuals of the domain class `<subject class>` are `<list of subject class individuals>`. The individuals of the range class `<object class>` are `<list of object class individuals>`. Which of these individuals should be connected with each other using the relation `<relation name>`? Return the connections in the format [["first subject name", "first object name"], ["second subject name", "second object name"], ...]. If none of the listed instances should be connected, you can introduce new instance names to connect them with the already existing ones. They should be written in camel case. Return all possible connections. Return nothing but this list.

# Acknowledgements

First and foremost, I would like to thank Dr. Anna Hallin for supervising this thesis and for her many helpful comments and suggestions, which have greatly improved the quality of this work. Thanks to Prof. Dr. Gregor Kasieczka for giving me the opportunity to work on this fascinating topic. I also want to thank the members of the Kasieczka Working Group for the friendly working atmosphere and for the interesting weekly meetings. Finally, I want to thank my family and friends for their support and encouragement while I wrote this thesis. Thanks to Martin and Matthias for their comments on my thesis. Thanks to Hannes, Tetiana, Claas, and Moal for offering me help with the correction of the manuscript.

# Affidavit

I confirm that I have completed this work independently and without outside help. I have not used any resources other than those listed in the bibliography. All passages that have been taken verbatim or in essence from publications are marked as such. I have not previously submitted this work in any other examination procedure.

During the writing process of this thesis, I used GitHub Copilot, an AI-based auto-completion tool, to complete sentences. This tool was only used to speed up typing and not for independent generation of content. My own work was always the value-adding contribution.

I agree that this work may be published.

Hamburg, _____ Signature: _____