Creating a Unified Structure for Positive and Negative Common Sense Knowledge

Silvia Mokráňová
Supervisors: Gaole He, Ujwal Gadiraju
EEMCS, Delft University of Technology, The Netherlands
28 June 2022

A Dissertation Submitted to EEMCS faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering

Abstract

Common sense is knowledge that most humans have, but machines do not. Generally, computer knowledge bases make use of positive (known) knowledge. However, in addition to positive common sense knowledge, there is also negative. Negative knowledge represent facts that are known to be untrue, like "a cat does **not** have fins". This knowledge is important in order for a machine to make assumptions the same way a human does. This research proposes ways to combine and organize the positive and negative knowledge tuples in a unified way. The two main ways that are looked into are table-like and graph-like organizations. These are analyzed based on different requirements and important queries are defined. Based on the requirements a recommendation is made. The research also takes a look at a solution that makes use of complex number space and suggest how this solution could be further researched and improved in the future.

1 Introduction

Common sense knowledge is information that helps people make sense of every day situations [7]. This knowledge is assumed instead of being written down, therefore it isn't trivial to get it into a machine's knowledge base. Common sense knowledge is important for machine learning because with it machines can make assumptions closely to how a human can. Common sense knowledge is used for reasoning for machines and the necessity to have common sense knowledge is increasing as new machine learning techniques arise [5].

Knowledge can be of many types, from generative (one relation of a concept to another concept - *cat is a mammal*) to discriminative (the distinguishing of two concepts in relation to a third concept - *octopuses, unlike fish, do not have fins*) [3]. Generative knowledge can be of two types - positive and negative [1]. Positive knowledge represents facts that are known to be true, for example *"a cat **is** a mammal"*. Negative knowledge refers to facts that are known to be untrue, for example *"a cat does **not** have fins"*. If all of the world's positive knowledge and all of the world's negative knowledge was combined, this would create a complete knowledge base. However, currently existing knowledge bases only store positive knowledge [12].

In common sense research, negative knowledge is just as important as positive knowledge [1]. A machine can only make assumptions based on the information it has in its knowledge base and if it lacks some information, it cannot simply make an assumption from that, unlike humans. A human can asses if they do not have enough knowledge to make an assumption and they know what is enough in order to make an assumption. Machines have no way of knowing if something is actually not a fact or if it has simply not been documented [6]. From positive knowledge (*a cat is a mammal*) one cannot assume anything about whether cats have fins or not. Sure, generally mammals do not have fins, but this is not a fact, as there are exceptions (eg. whales), so this is not a fact a machine can simply assume, even though we as humans can. However, being explicitly told that something is indeed not true (*cats do **not** have fins*) minimizes this issue.

Adding negative knowledge into the existing knowledge base is not as trivial as it may seem, as it is a completely different type of knowledge and it is used in a different way. Negative knowledge needs to be used for negative reasoning, which is different from positive reasoning.

The main research question to be answered is whether positive and negative knowledge can be organized in a unified way. A unified organization for knowledge is desired for ease and effectiveness of retrieving the knowledge, which brings up the following subquestions:

1. What structures can be used for a unified organization of positive and negative knowledge?
2. How can different knowledge organizations be evaluated and how do they compare?

The first part of the research is exploring table-like organizations and how they can be applied to storing knowledge. Then graph-like organizations are explored. The solutions are then compared based on numerous criteria. From the four proposed solutions, the hypergraph solution appears to be the best fit.

This is not an exhaustive list of possible ways to organize knowledge. There are certainly many more ways to store knowledge that this research does not go into. However, apart from the table- and graph-based solutions, the paper also briefly looks into a complex number space solution and proposes ways this can be further investigated in future research.

A basis for combining positive and negative knowledge in a unified way is provided in this paper. Multiple organizations are compared and improvement possibilities for the proposed organizations are looked into.

First, related work is discussed in Section 2, After that, the methodology is discussed in Section 3. Section 4 discusses the different ways of organizing knowledge that were proposed during the research. The next section, Section 5, describes researched uses of common sense knowledge bases and discusses important criteria when organizing knowledge. The paper then compares the proposed knowledge organizations based on the criteria. Next a discussion is in Section 7 and a conclusion in Section 8. The paper ends with a reflection on responsible research in Section 6.

2 Related Work and Preliminary Knowledge

The majority of currently existing knowledge bases (for example Atomic [10] or Cyc [8]) focus on positive knowledge, whereas this research is going to focus on negative knowledge and combining positive and negative knowledge in a unified way. The research done for the FindItOut game [2] focuses on acquiring both positive and negative knowledge. In their implementation, the acquired knowledge is stored in four separate tables, one for positive generative knowledge, one for negative generative knowledge, one for positive discriminative knowledge and one for negative discriminative knowl-

edge. Within the table about generative knowledge it is structured as a 3-tuple, with the two concepts and the relation of the first one to the second.

During this research multiple ways to organize knowledge are proposed. They make use of different data structures, which are introduced in this section.

2.1 Relational Databases

First, a look into using relational databases [4] to store the knowledge is needed. A relational database can be visualized as interconnected tables, however in the proposed solutions only one table is used, with connections only to itself. Each row in a database table has to have a unique key, which is what makes the databases more effective than a simple, array-like, table. Each row can be represented as a tuple of the information in the cells of the row. In databases, it is also possible to allow $null$ cells, in case some data is not available or relevant or needed. This option can be enabled or disabled for certain colums based on what is needed.

2.2 Graph Databases

Similarly to previous research, this one makes use of graphs for knowledge storing, similar to graph databases like Neo4J [9]. Graph databases consist of having nodes and their connection via edges. The edges are directed and labeled with a relation between the two nodes. The nodes can be of different types and one can define rules of what can be connected to what and in which way. It is also possible to add $null$ nodes or edges if necessary.

One downside of regular graphs is that only two nodes can be connected with an edge, which is why hypergraphs [11] may be more useful. In a hypergraph, an edge can connect as many nodes as needed. The edge can also be a directed edge, with a starting (head) and ending (tail) node. The edge still has the ability to have its own label as well. This scales down the necessary space needed to store all the information and, if implemented correctly, makes looking up many queries incredibly effective.

3 Methodology

The two important parts of this research were coming up with ways to organize knowledge and then comparing them and finding the ideal one. This paper is therefore also divided into two main parts - the knowledge organization methods themselves and the analysis of their performance in comparison with each other.

3.1 Knowledge Organizations

The research started with reviewing existing literature about knowledge bases. Previously done research focused primarily on positive knowledge and its organization, however the aim of this research is to add negative knowledge into the knowledge base, or to find a new organization altogether. The purpose is to ensure the usage of negative knowledge smooth and effective in the proposed organization.

Based on the previously reviewed work, multiple ways of combining positive and negative knowledge were proposed. The different organization proposals are described in Section 4. The two main categories that were looked into are a table-like organization and a graph-like organization. For each of the styles, two different solutions were proposed, and a trade-off between space and time complexity was made. A basis for further research is proposed for a solution that makes use of complex number space and mathematical operations that would speed up the organization significantly, however due to time and knowledge constraints, this was not further explored.

3.2 Evaluation Protocol

As there are many ways negative knowledge can be added into a primarily positive knowledge base, a way to compare these is needed. The question is, how to decide what the best method of organizing knowledge is? And is there one? It is important to know which criteria are important and which are not. To know this, one needs to know what the knowledge will be used for, as different ways of organization can work for different applications. A part of this research consisted of finding different applications and usages of common-sense knowledge in computer science, and specifically ways negative knowledge can affect these applications and when it is needed.

After finding applications of knowledge, it was explored how exactly it is used and what operations and queries are most important. The different ways of organizing knowledge were then analysed and categorized based on their effectiveness with the found applications. The criteria are described in Section 5.1.

It was expected that not all organizations would perform the same and that there would not be a clear "best" organization, but rather that different styles of organization are useful for different applications.

4 Ways to Organize Knowledge

The information that needed to be stored consists of a concept, a relation and a second concept. The information about whether the knowledge is positive or negative also needs to be available in the organization. After researching ways positive knowledge is organized, several methods for creating a unified organization for positive and negative knowledge were proposed.

4.1 Table Based Organizations

The way the knowledge is currently organized is simply in two separate tables, one with the positive knowledge and one with the negative, each with three columns, which can be represented as a 3-tuple that looks like this: $<$*Concept1, Relation, Concept2*$>$. This way, getting knowledge about a concept requires one to look into two different tables, which is why a unified structure would be preferred.

The first organization idea was to merge the tables into one and distinguish the relations as either positive or negative. This means that, for example, the relations isA becomes $isNotA$ in a negative tuple, just like in natural language. However, in order to query the table-based database more easily, it would be better to not use natural language when distinguishing the positive and negative relations, so a

A	isA+	B
A	isA-	C
B	hasA+	D

+	A	isA	B
-	A	isA	C
+	B	hasA	D

Figure 1: In this figure the two ways to store knowledge in a table can be seen. The capital letters represent concepts, the plus and minus sign represent positive and negative knowledge respectively and the remaining column in the relation between the concepts. On the left, the knowledge is represented as a 3-tuple with a relation that indicates whether it is positive or negative. On the right, the distinction is in a separate column, making it a 4-tuple.

decision was made to use $isA+$ and $isA-$. This raised a second idea, which was to add a separate column to represent the difference between positive and negative. This is visualized in Figure 1.

4.2 Graph Based Organizations

A significant amount of the previously done research stores knowledge in a knowledge graph, so this was the next explored idea. In Figure 2 a graphical representation of what a simple knowledge graph would look like can be seen. This solution builds on top of the 3-tuple solution. The nodes in the graph represent the concepts and the directed edges represent relations between them. This way, if a concept is present in multiple tuples, which is should be, as a concept can have many relations to many subjects, it only needs to be stored once. It can then have edges connected to all the concepts it is related to. In this solution, the relations are either positive or negative, but the distinction is only expressed on the edge itself, by the relation being either positive (isA) or negative ($isNotA$).

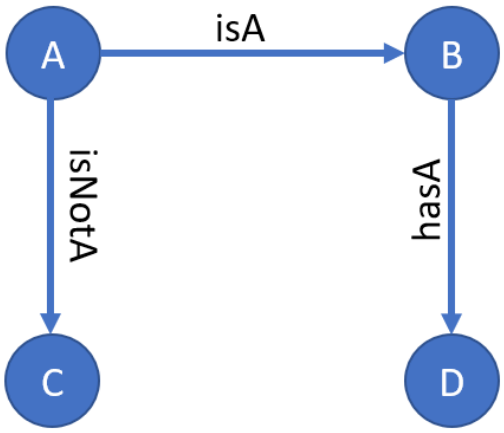

Figure 2: In this figure, a graph representation is visualized. The nodes represent concepts and the edges represent the relations between them.

The next idea was to express the distinction between positive and negative knowledge explicitly, so the idea of using a hypergraph came. In a regular graph, an edge connects two nodes, but in a hypergraph, a so-called hyperedge can connect multiple nodes. A hypergraph can contain directed hyperedges, where the direction is indicated by having a head node, which connects to the next node and the next. The final node is the tail node. In the proposed knowledge hypergraph, there are two types of nodes, concept nodes and relation nodes. A directed hyperedge connects together the first concept, the relation and the second concept. The value of the edge represents whether the knowledge is positive or negative. This can be seen in Figure 3.

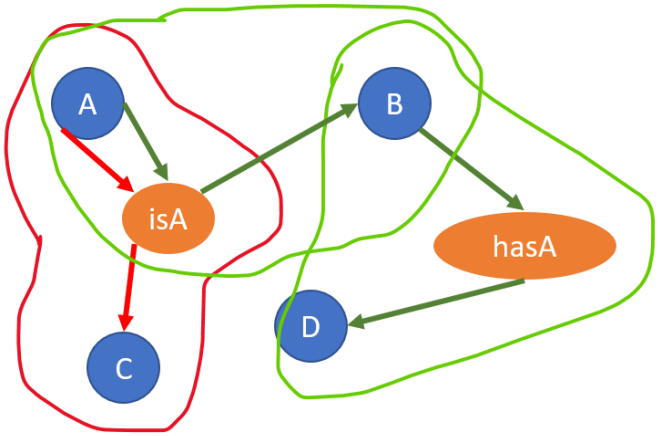

Figure 3: In this figure, a hypergraph representation is visualized. There are two types of nodes, concepts and relations. In blue, concepts are shown and in orange relations. The hyperedges connect two concepts and a relation. A green hyperedge (visualized as circling) represents positive knowledge and red represents negative.

4.3 Complex Space Based Solutions

Inspired by the way complex numbers are represented as $A * r + B * i$ where r represents the real part of the number, i represents the imaginary part and $A, B \in R$, a solution making use of complex number space was proposed. The research very briefly looked into a solution that makes use of this. This solution was not fully explored, as there was not enough time to get the required background knowledge. However a basis can be provided on how this idea can be developed further. A couple ways were looked into and some were quickly dismissed, due to obvious flaws, but some are worth exploring further. This section will briefly mention four of them.

The first method, which can graphically be seen in Figure 4, represents positive knowledge as the $real$ part of the tuple and negative knowledge as the $imaginary$ part of the number. The connection between them, represented as a point on a graph, can then be used to represent positive disriminative knowledge as well. In this solution, an obvious flaw is already visible, as there is no way to represent negative discriminative knowledge, and only $1/4$ of the complex space graph is in use.

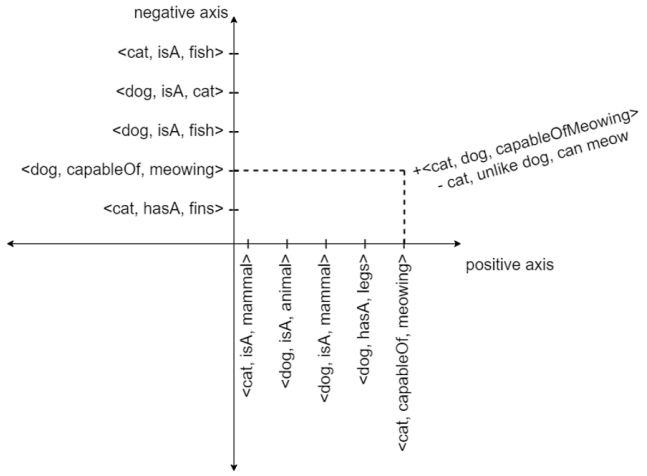

Figure 4: This figure graphically shows how positive and negative knowledge can be represented as the real and imaginary part of a number in complex space, with positive knowledge being on one axis and negative on the other axis. Their connection - points on the graph, represent positive discriminative knowledge.

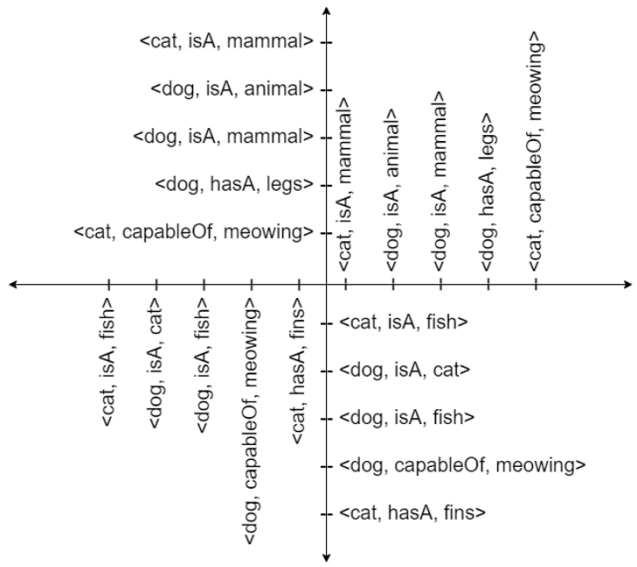

Figure 5: In this figure, positive knowledge is where positive numbers would be and negative knowledge is where negative numbers would be. A point on the graph represents their connection

The initial idea inspired the second version, which has two identical axes, as seen in Figure 5. The axes represent positive knowledge and negative knowledge the way positive and negative numbers are represented, so on the right positive and on the left negative (or with a vertical axis, top and bottom respectively for positive and negative). This means the axes are not symmetrical. This implementation allows for the representation of both positive and negative generative knowledge (using just one axis), as well as positive and negative discriminative knowledge (using two axes). There are still parts of the space that are not used, only half is used, but this is an improvement as compared to the first solution.

The third idea went in a slightly different direction, with the axes representing concepts instead of full pieces of knowledge and the points on the graph representing the pieces of knowledge. The axes then has a $positive$ concept on the right (or top) and $negative$ on the left (or bottom). This can be seen in Figure 6. A connection between them, represented as a point on the graph or a tuple, then represents a piece of knowledge. The point on the graph is annotated by a relation, to represent the type of relation between the two concepts. If the first concept is positive this represents positive knowledge and if it is negative it represents negative knowledge. As in the previous solution, only half of the space is in use. If this solution were to be expanded to include discriminative knowledge, as third dimension could be added.

The final idea also adds the relations into the space, instead of representing them on the points. The relations are represented as a separate dimension, as seen in Figure 7. In this case, only the positive part of the knowledge axes is used and knowledge is represented by a point on the graph, which combines two of the concepts and the relation. The relation can either be positive or negative (represented the same as positive and negative numbers on a graph), and this determines whether the piece of knowledge is positive or negative.

In this solution, positive and negative generative knowledge are represented in a way that is easiest to grasp and seems quite obvious. It also adds the possibility to expand the representation and add generative knowledge, by adding another dimension (or axis).

The complex space solutions require a mathematical background to be fully developed, but with an operation that makes storing complex space tuples easy, this can be done. This was not further researched and is only visited again in the section about future work, as it is certainly worth exploring further.

5 Organization Analysis

In order to decide which way of organizing is the best, criteria for comparison needed to be found. This included researching usage of knowledge and deciding on important ways the knowledge base will need to be queried. The different organizations are then compared, excluding the complex space solution as it is not mathematically finished and cannot be analyzed at the time of writing this research.

5.1 Evaluation Protocol

There are several things to consider when comparing knowledge bases, in order to accurately judge their performance. This section describes the criteria used for comparison within this research and their relevance to the overall conclusion.

The first thing to be considered is the space the knowledge base will take up. This means, the amount of space needed for a single piece of knowledge, but also the amount of times certain things need to be stored. So, is it needed to store a reference to a concept once? Or multiple times? This is very important for space related reasons, but also for when new

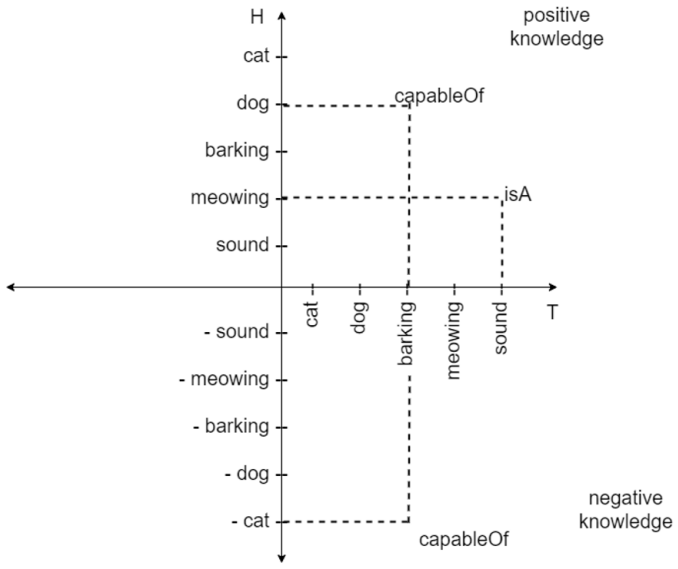

Figure 6: In this figure, concepts are represented on the axes, with the positive side representing a positive concept and the negative side a negative concept. A connection between them, or a point on the graph, represents the piece of knowledge.

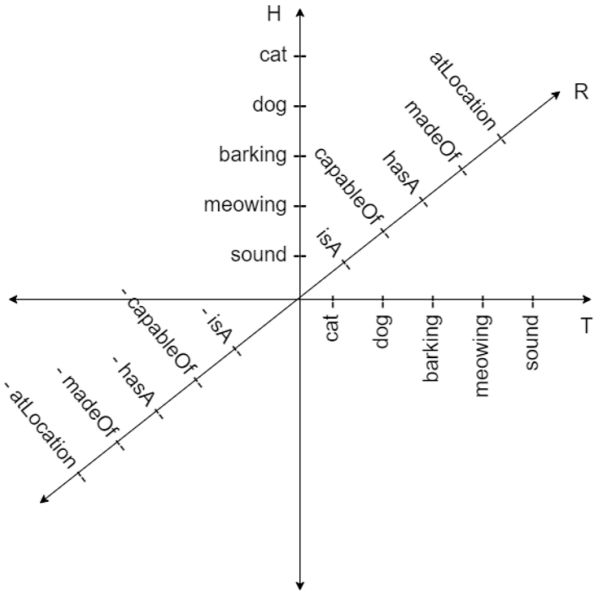

Figure 7: In this figure, concepts are represented on the axes and relations are also represented as one axis. A point on the graph represents a piece of knowledge.

information is being added to the knowledge base and when information is being searched for with a certain query. If a reference is stored multiple times, all of these need to be updated when something changes about the piece of knowledge or about the concept. This might sometimes be acceptable, but it is important to consider if this proneness to error is worth it for the given knowledge base organizations.

This brings out the importance of querying the knowledge base. Naturally, the knowledge base needs to be easy and fast to query, otherwise it is kind of pointless to have it. The worst case time complexity of executing different queries is therefore considered. The knowledge consists of four pieces of information - H = the concept that the knowledge is about, R = the relation of H to T, T = the second concept and $+/-$ = the information about whether the knowledge is positive or negative. A query is represented as *<+/-, H, R, T>*, with a $*$ used in place of the information being searched. With this in mind, the following queries are the ones that are considered important for comparing the different knowledge bases:

- *<*, H, *, *>* - find all (positive **and** negative) knowledge about concept H
- *<+/-, H, *, *>* - find specific (positive **or** negative, based on input) information about concept H
- *<+/-, H, R, *>* - find which concepts T concept H relates to with relation R
- *<*, *, R, T>* - find all concepts H which relate in R to concept T. **Both** positive and negative.
- *<+/-, *, R, T>* - find all concepts H which relate in R to concept T. **Only** positive or **only** negative.

Apart from querying, the worst case time complexity of adding new knowledge into the knowledge base or removing knowledge from the knowledge base needs to be considered as well. This is less important, as the focus is for extracting the knowledge to be fast, however it is still something to be considered when comparing the different knowledge bases.

The last thing to be considered is the ease to expand the knowledge organization. This research focuses mainly on combining positive and negative generative knowledge, however future research should also focus on combining this representation with disriminative knowledge (which contains two main concepts, a relation and a third concept). This ability to expand is therefore not the most important criteria, but it is something to be considered if more types of knowledge were to be added into the knowledge base.

5.2 Evaluation and Analysis

When comparing the knowledge, some terminology needs to be defined first. A piece of knowledge is the combination of a concept, relation and second concept, or *<H, R, T>*. The amount of concepts (H and T) is represented by n and the amount of knowledge is m. In a table representation, m is the amount of entries, whereas in a graph solution this is the amount of edges. The amount of nodes in a graph solution is n, as they all represent concepts. In the hypergraph solution, the amount of nodes is $n + k$, where k is the amount of possible relations.

Table Based Solutions

When storing knowledge in a table within relational database, each line, or tuple, represents a piece of knowledge. If one wants to look at all knowledge about a concept (or *<H, *, *>*), every line needs to be looked at and the corresponding ones need to be returned. The same is the case for all the other considered queries. This means the worst case time complexity of the operation is $O(m)$, and that a reference to

the concepts needs to be made each time they are mentioned in a piece of knowledge. The table needs to have m entries, for each piece of knowledge. Adding knowledge is a $O(1)$ operation, because it simply needs to be added to the end, however if one wants to make sure there are no repetitions, it is an $O(m)$ operation, since all entries need to be checked. When deleting knowledge, all entries need to be checked as well, which means a worst case time complexity of $O(m)$.

The last factor to consider is the ability to expand the knowledge base to add different type of knowledge to it. In order to add additional knowledge to a table style knowledge base, an additional column could be created with the second concept, or a separate table that contains connections of different knowledge. The first solution would require a lot of repetitions in the table as well as a lot of $null$ entries. This is not something that has been researched further, but it could be if the knowledge base were to be expanded.

The difference between the 3- and 4-tuple solution is in the queries that require a distinction between positive and negative knowledge. In the 3-tuple solution the distinction between positive and negative knowledge is just within the relation and not obvious at first and can not easily be searched by. It is possible, but the 4-tuple solution makes the distinction a lot easier. In terms of time complexity, there is no difference, however a query would have to be a lot more complicated in the 3-tuple option, which could easily lead to bugs when it is being used in an implementation. This is a fixable issue, but it would be better to avoid it altogether.

Graph Based Solutions

In contrast to the table, the graph solution represents a piece of knowledge as an edge, connected to two concepts. This means each concepts is stored once, no matter if it is H or T and they each contain information about which concept they are connected to with which relation (= edge). Since there are n concepts, meaning n nodes, the worst case time complexity for the queries is $O(n)$. This is at least as good as $O(m)$, if there was one piece of knowledge per concept. However that is not the case, as each concept has significantly more knowledge about it than one piece. Therefore, this give that $O(n) \geq O(m)$. Adding knowledge requires looking up the concept and then adding an edge. This means a worst case time complexity is $O(n)$ and same goes for deletion.

The query time complexity could be improved by storing references to the concepts/nodes in a hash map data structure. This way, any concept look up would go down to $O(1)$ and a relation look up would require an average time complexity of $O(m/n)$ considering the concepts have about the same amount of knowledge. This can not be done in the table solution easily, but in a graph solution this is an easy improvement.

In the hypergraph solution, the worst case time complexity is $O(n)$ when dealing with queries that include information about a concept, and $O(k)$ when it is about a relation. The *<*, R, T>* query is significantly easier in this organization, as looking up the relation is $O(k)$ and then just the edges need to be looked it, which on average is $O(m/k)$.

In a graph vs. hypergraph, looking up positive vs. negative knowledge is quite similar, except again, just like in 3- vs. 4-tuple, the query will get more complicated in a regular graph, as opposed to a hypergraph. In a hypergraph, filtering needs to be done on one parameter - the edge type, whereas in a regular graph the filtering needs to be done by selecting all the positive (or negative) edges.

The hypergraph has the most potential for expansion. In order to expand it with disriminative knowledge, a node needs to be connected to the hyperedge. This provides the opportunity for more types of knowledge as well, if needed, as the hyperedge can contain as many nodes as necessary. This solution also provides the ability to easily create new relations without affecting the rest of the knowledge base. All the findings are summarized in Table 1.

In order to conclude that a graph solution is better than a table solution, $m < n$ must hold. With n being the amount of concepts and m being the amount of connections, one can assume that all concepts have at least one connection to another concept, otherwise the concept is irrelevant. This means that at least $m = n/2$, which would make $m > n$. However, this is not the case, as a knowledge base with the minimum possible amount of knowledge would be quite useless. So, considering there is at least one piece of information about a concept, it can be concluded that $m \leq n$. One should make sure of this for the knowledge base they are implementing, and decide based on their own values which organization to use.

	3-tuple	4-tuple	graph	hypergraph
<*, H, *, *>	O(m)	O(m)	O(n)	O(n)
<+/-, H, *, *>	O(m)	O(m)	O(n)	O(n)
<+/-, H, R, *>	O(m)	O(m)	O(n)	O(n)
<*, *, R, T>	O(m)	O(m)	O(n)	O(1)
<+/-, *, R, T>	O(m)	O(m)	O(n)	O(n)
adding	O(1)	O(1)	O(n)	O(n)
deleting	O(m)	O(m)	O(n)	O(n)

Table 1: The table shows the different worst-case time complexities for various organization solutions. The *n* value represents the amount of concepts and *m* represents the amount of knowledge, or lines in a table based solution.

Complex Space Based Solutions

Unfortunately, due to time constraints, this method of organization could not be analyzed further, as it would require a more extensive mathematical background. It is believed that this solution may have great benefits if explored fully and is therefore worth mentioning here and looking into further in future research. If done well, this solution may significantly improve the time complexity for the proposed queries.

6 Responsible Research

It is very important for all steps of the research to be reproducible and this was taken into account. Since the paper goes on to propose multiple ways common sense knowledge can be organized, it was important for these to be properly thought through and then documented. This was done by researching peer reviewed literature about knowledge bases and their structures. Based on these, new methods of organizing

knowledge were proposed, described in detail and reasoned about. Then, these methods were analyzed and compared based on criteria. These criteria came from peer reviewed work or, for the ones made up specifically for this researched, there is reasoning about them and their importance and relevance. All the steps taken are described in full detail and reasoned about, to ensure openness about the problem.

7 Discussion

The two main subquestions were about the different ways to organize knowledge and about how the different organizations can be analyzed and compared. Four different solutions were fully explored and compared. These solutions all either perform better than or at least as good as storing all the data in two separate tables.

In case of the table solutions, storing the data in one table instead of two does not have obvious benefits in terms of time complexity, but it is certainly better in terms of ease to search and proneness to error when implementing or when using queries.

The graph solutions, while more difficult to implement, provide a more effective way to browse knowledge and distinguish between positive and negative knowledge. Hypergraphs have shown to be the best solution out of the four proposed solutions, mainly because of its ability to expand and store more types of knowledge as well. Both graph solutions are also easier to visualize if a small sample is used, as looking through a table is less interesting than looking at connected nodes, but this is a purely subjective criteria.

The four fully explored solutions were compared to each other, however they were not compared to the complex space solutions or to other possible solutions that one could come up with. This means that even though the hypergraph may seem like the best solution, and it certainly is the best of the four proposed solutions, there may be a better solution that has not been thought of for this research.

With that in mind, it is important to remember that the four proposed solutions are not the only way data and knowledge can be stored. The research did not have time to look into it fully, but there were some ideas of how to store knowledge using complex number space. These ideas were described and visualized, but not mathematically supported as that would require further mathematical knowledge. The complex space solution is something that certainly needs more research, however implementing it is not entirely trivial. It may have great benefits in terms of time and space complexity, as well as options for expansion to add different knowledge types.

8 Conclusions and Future Work

The research focused on combining positive and negative common sense knowledge in different ways. This is important, as, unlike humans, machines do not have the ability to assume things the way people do, because people do this based on knowledge that they have. This knowledge is so-called negative knowledge, which, in combination with positive knowledge, creates a complete knowledge base, in which assumptions are easily made, as all is known.

The researched looked into two main ways of storing knowledge. Namely, it looked into table and graph solutions. It also briefly discussed a complex number space solution, however not in full detail and without a further analysis, due to time and knowledge constraints.

The table and graph solutions were compared based on various criteria and it was found that a graph solution is generally better. In case the knowledge were to be expanded (like adding discriminative knowledge, or even other types), it was found that a hypergraph solution fits best. This research does not represent an exhaustive look into knowledge storing, as there are certainly many ways knowledge can be stored, but it provides a basis for understanding how this can be done and what could be improved. Mainly, the complex space solution is worth researching further, as it has potential to be far more effective than the other proposed solutions. However, it may also be more difficult to implement, which is also a factor when deciding what knowledge organization to use.

If one were to use any of the proposed solutions, it may be worth looking into different queries and their complexity as well, depending on the application of the knowledge base. The queries discussed in this research are what was most important according to the research team, but this may be different for each user and application of a knowledge base.

None of the organization methods were implemented, this is just a theoretical basis for further research, so more issues may occur while implementing the proposed solutions. This may either lead to further development of the ideas or may lead to completely new ideas for knowledge organization. The question about which way to store knowledge remains an open question, as more ways can be explored.

References

[1] Hiba Arnaout, Simon Razniewski, Gerhard Weikum, and Jeff Z Pan. Negative knowledge for open-world wikidata. In *Companion Proceedings of the Web Conference 2021*, pages 544–551, 2021.

[2] Agathe Balayn, Gaole He, Andrea Hu, Jie Yang, and Ujwal Gadiraju. Finditout: A multiplayer gwap for collecting plural knowledge. 2021.

[3] Agathe Balayn, Gaole He, Andrea Hu, Jie Yang, and Ujwal Gadiraju. Ready player one! eliciting diverse knowledge using a configurable game. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 1709–1719, New York, NY, USA, 2022. Association for Computing Machinery.

[4] Edgar F Codd. A relational model of data for large shared data banks. In *Software pioneers*, pages 263–294. Springer, 2002.

[5] Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103, 2015.

[6] Vladimir Adamovich Golovko, Alexsander Alexsandrovich Kroshchanka, Vladimir Vasilyevich Golenkov, Valerian Petrovich Ivashenko, Mikhail Vladimirovich Kovalev, Valery Vasilyevich Taberko, and

Dzmitry Sergeevich Ivaniuk. Integration of artificial neural networks and knowledge bases. , (8):133–147, 2018.

[7] Filip Ilievski, Alessandro Oltramari, Kaixin Ma, Bin Zhang, Deborah L. McGuinness, and Pedro Szekely. Dimensions of commonsense knowledge. *Knowledge-Based Systems*, 229:107347, 2021.

[8] Cynthia Matuszek, Michael Witbrock, Robert C Kahlert, John Cabral, Dave Schneider, Purvesh Shah, and Doug Lenat. Searching for common sense: Populating cyc from the web. *UMBC Computer Science and Electrical Engineering Department Collection*, 2005.

[9] Ian Robinson, Jim Webber, and Emil Eifrem. *Graph databases: new opportunities for connected data*. " O'Reilly Media, Inc.", 2015.

[10] Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035, 2019.

[11] Bryce Merkl Sasaki, Joy Chao, and Rachel Howard. Graph databases for beginners. *Neo4j*, 2018.

[12] Charles Yang. Negative knowledge from positive evidence. *Language*, pages 938–953, 2015.