Learning platform for training critical thinking

Manfred Wedemalm
Abstract

Learning platform for training critical thinking

Manfred Wedemalm

Media appear in several forms in today's society and play a big role in the typical adult's day. During exposure to media it is impossible to avoid being exposed to special interest media strategies. It is the schools' responsibility to teach students how to see through these strategies. Today there are several online learning platforms that deliver solutions for managing administration, documentation and evaluation of teaching. Platforms focusing on learning a specific objective are however more rare. This thesis describes the development of a prototype intended to support learning of critical thinking: a web-based platform with a collaborative learning method that provides basis for discussion and gives the students a firsthand experience of how impactful media can be.

Teachers get support creating, editing and comparing text with a diff engine made with dynamic programming. Concluding that making a full scale platform would be possible to ease the learning and teaching experience for one important objective of the Swedish school, critical thinking.

Handledare: Mikael Laaksoharju
Amnesgranskare: Iordanis Kavathazopoulos
Examinator: Johannes Borgström
IT 20 023
Tryckt av: Reprocentralen ITC
1 Introduction

The typical adult spends close to ten hours per day using some form of media, which is almost double the time compared to 1945 [1]. During these hours, the exposure from special interests is unavoidable, and not always easy to detect. Knowledge about how the human brain functions during these exposures, has opened up for special interests to disguise their attempts to use media for their own benefits. For instance trying to nudge people in one way or another [2].

1.1 Background

The human brain uses two systems when thinking, the automatic system and the reflective system [2]. The Automatic system is the system being used when ducking for a ball flying towards the head, or performing routine tasks. It is fast, uncontrolled and effortless, often referred to as one’s unconscious. The Reflective system is the system being used when solving maths problems, it is controlled, slow and effortful. It is the system people often associate with the word thinking. These systems are known and can be used by companies, media or other special interests, that might try to influence people’s actions in the direction they desire. There are several techniques of doing this: one is nudging. One example of the effect of nudging is the number of registred organ donors per million population in countries with an opt-in consent (citizens have to consent with their organs to be donated), compared to countries with an opt-out consent (citizens have to inform that they don’t consent with their organs to be donated). Spain has opt-out consent and a number of 35.3 of deceased organ donors per million of population, while Germany, that has opt-in consent, only has a number of 10.7 [3] [4].

Nudging is only one of many ways for special interests to redirect people’s actions. Closure is one that is being used in many social sites like Facebook. By claiming that the user’s ”...profile is 80% complete”, users are more likely to fill in more information about themselves [5], information that for instance can be used for selling directed ads.

One substantial example of how different wordings can affect people’s opinion is the following example from Daniel Kahneman and Amos Tversky’s ”Choices, Values, and Frames” [6].

Imagine that the United States is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume that the exact scientific estimates of the consequences of the programs are as follows:
If Program A is adopted, 200 people will be saved.

If Program B is adopted, there is a one-third probability that 600 people will be saved and a two-thirds probability that no people will be saved.

When choosing between Program A and B most people will probably pick Program A, since it will guarantee saving 200 lives. Program B has a clear gamble aspect that does not favour the desirable results, and therefore will most likely be neglected. Now consider changing the Programs A and B to the following:

If Program A’ is adopted, 400 people will die.

If Program B’ is adopted, there is a one-third probability that nobody will die and a two-thirds probability that 600 people will die.

When choosing between Program A’ and B’ most people will probably pick Program B’, since Program A’ will take 400 lives. Program B’ still has a clear gamble aspect that does not favour the desirable results, but the risks seem lower in comparison to taking 400 lives. A closer look at all four programs reveals that there are no difference in the outcome of program A contra A’ or program B contra B’. Out of 600 people, 200 will survive if choosing A or A’. Program B and B’ has the same probability (one-third) to save all 600 lives, and the same probability (two-thirds) to fail saving any life. Even though the outcome is the same the most likely choice between the programs diverge. They diverge due to clever choice of wording. This example shows how easy it is to steer people’s points of view.

With all the media exposure and it is extremely important to think critically. The Swedish Agency for Education states in their curriculum, that students should learn to think critically, to examine facts and relations and realise the consequences of different options; in other words approaching a scientific way of thinking and working [7].

1.2 Project

Teachers today can choose between several digital platforms that deliver solutions for managing administration, documentation and evaluation, for example Moodle [9] and Sakai [8]. The number of tools designed to reach specific objectives are more limited. If special groups can use technology to manipulate consumer opinion, why not use technology to teach consumers
how they do it? A platform that focuses on reaching this objective could help the school in a different way than existing platforms of today.

The goal is to create a prototype of a platform that helps teachers teach their students to think critically, and to evaluate the possibility to creating a full scale version. Studies have shown that collaborative learning is beneficial when learning critical thinking [10], and will therefore be the backbone of the learning method used in the prototype. The Kahneman example from section 1.1 shows that opinions from text with the same factual outcome can diverge just based on the choice of words. By using this type of presentments and splitting a class in two, where one group reads a different version (with the same factual outcome) than the other there will be a basis for discussion. To demonstrate the diverging opinions in the class, the students will be asked to answer a multiple-choice question and are shown the results before the discussion begins. After the discussion, students have the possibility to view both versions, with the differences visualised. This learning method would not only give students an explanation of why critical thinking is important but also make them part of an example of how easy it is to influence people’s opinions. Teachers should be able to set up discussions either by choosing presentments from a library or by creating new ones. There should consequently be support for viewing, creating and editing presentments, where the differences between the texts are visualised.

Goals for the project:

- Create a database for presentments;
- Create a diff engine for visualising differences between texts;
- Design support for creating and editing presentments;
- Design support for creating a discussion in a class;
- Design a solution for automatically dividing students within a discussion;
- And make it easy for students to participate in a discussion.

1.3 Delimitation

The presentments are an essential part of the platform; without them there is no basis for the discussions. Writing presentments however has very little to do with computer science and will therefore not be a part of this project.
The database will be an essential part of the prototype, handling information about at least presentations and user accounts. Since the database will be such an essential part, different kinds of administrators will be needed. Which kinds of administrators is out of scope for this thesis and will therefore not be discussed.

When dealing with user information, security becomes a big part. A full scale version would for instance need to use encryption for user passwords. As this is a thesis in human computer interaction, security is out of scope. Security of the different frameworks and techniques will thus not be discussed.

2 Technical background

There are several frameworks and techniques which could be useful for the prototype. Some of them will be discussed and compared in this section, for instance different types of databases and techniques for visualise differences in text. It is possible that some code from this prototype will be reused in a potential fully scaled version. This will be kept in mind during these comparisons. Which framework or technique is best suited is not only determined by specs. For instance the more widely used database might be a better candidate then the fastest database, oweing to the fact that it is more likely that a programmer can understand and further develop code from one s/he has seen before.

It is decided that the platform should be used through modern web browsers. There are many browsers that are considered modern for instance Google Chrome, Safari, Internet Explorer, Opera and Firefox, to name a few. These browsers have different Application programming interfaces (API) meaning the code might deviate between each other. It is not reasonable to make sure the prototype works as intended on all modern web browsers. A goal is therefore to use as little browser specific code as possible, and make sure that it works on two of the most popular browsers today, Google Chrome and Firefox [11].

2.1 Database

As code might be reused, flexibility and the ability to further develop the prototype must therefore be a goal. This means that not simply every database will do. The choice of database should be one that can be easily used and changed by other developers. Because of time limitations only two different types of databases will be evaluated: One relation database, MySQL and one noSQL database, CouchDB.
In 1994 the Swedish company TcX wanted a relational database with a Standard Query Interface (SQL). The problem was that there were none which met their needs. All databases that they tested were too slow for large tables or lacked features [12], and the work with MySQL began. In 1996, MySQL was released in binary form for Linux and Solaris. Today, it is also available in source form, and works on most platforms. The combination of speed and open source has made MySQL one of the most popular databases. It is used in both small and big projects.

Speed is one of MySQL’s show-pieces. Speed was one of the main reasons why MySQL was made and is constantly being improved [13]. Figure 1 shows speed difference between different versions of MySQL.

Since MySQL is a well known database, the assumption that most developers have already had some contact with it is therefore not very far fetched. Development tools that are well known often have several support mechanisms, such as mailing lists, forums and manuals. These support mechanisms are present for MySQL [12]. That it is well known is a big plus when aiming at a flexible prototype, since it will hopefully lead to other developers not having trouble modifying the underlying database.

MySQL is a relational database [12], meaning that data is stored in an entity. The database is composed of several entities with relations [15]. The entities are often a representation of something from our world, for instance a person or a class. The data saved to a entity is called attribute and should be something associated with that entity. These entities can have several connections with each others, a relation, thus the name relational database. For instance a person can own a car, and a car can be owned by a person [15].
Relational databases often use the standard query language (SQL) and as the name implies MySQL is no different on that point [12]. This is positive since developers with experience from other SQL based databases will have an easier time to understand the database of the prototype.

2.1.2 CouchDB

MySQL is a rather old database compared to CouchDB, which released its first unstable version in 2009 [16]. It is being marketed as a "database that completely embraces the web" [17]. With solutions such as storing data in JavaScript Object Notation (JSON) documents, query indexes via Hyper Text Transfer Protocol (HTTP) and the possibility of managing data by combining and transforming it with Javascript, it is easy to understand why.

In contrast to MySQL, which requires a model of the data, CouchDB is a schema free database [18]. CouchDB achieves this by using JSON documents as data structure. This makes it easy to add or remove data. The idea is that a document should be like a sheet of paper in the real world [18]. This makes CouchDB flexible, which suits this project. However, it might be a problem as well. If an application relies on all documents to contain a data entry called "type", and one document is missing such an entry, then the application might miss that document. In a database with schemas this would not be a problem since the schema guarantees that there will be a data entry with that name. However, with some good practice while creating documents this should not be a big problem. The JSON objects work in a similar way to several other objects in object-oriented language, which is a big plus since it will ease the learning curve for developers without any experience working with CouchDB.

As mentioned before, CouchDB uses JavaScript, to combine and transform data. This is done with a view [18]. A view is a function telling CouchDB which data to return. Once the view is queried, CouchDB will run the function on every document in the database, and return the result. This might sound like a costly operation if there are many documents, and it is. Luckily, CouchDB caches the result and only runs the function once per changed document. This feature is really useful, and makes CouchDB even more flexible, something that suits this project very well.

2.2 Server

For choosing what kind of frameworks to use in the server the required functionalities have to be settled. In this project the server needs to be able to
fetch data from the database with all presentments, possible accounts and classes. Teachers should have some support for creating and editing presentments, as stated in section 1.2. The server will therefore also be able to add and modify data. During a discussion the class should be divided into two groups, therefore the server needs to be able to group clients in someway. A way to update pages to clients in real time would be useful for showing how students vote during discussions.

These are functionalities that will affect the choice of frameworks used for creating the server. There are several other functionalities that the server will need, for instance dividing students in groups of two within a discussion.

- Be able to request, change, add and possible delete data from the database;
- Grouping mechanism of clients;
- And updating pages to clients in realtime.

2.2.1 Node.js

Node.js is a framework for developing scalable server side web applications built on Chrome’s V8 engine [19]. Node.js is an open source project with a large ecosystem called Node Package Manage (NPM). NPM has a large number of open source libraries to be used with Node.js. NPM is perhaps the biggest advantage of using Node.js, since there are many frameworks that can add functionality to already existing server code. With Node.js developers can also take advantage of the full stack javascript paradigm, giving them insight in all parts of the application. Javascript is the language that at the time of writing impregnates the web, which means most developers has been in contact with Javascript at some point. This is a big plus for this project, since some code might be reused in a full scale version.

2.2.2 Apache Server

"The most popular web server on the Internet since April 1996" is the Apache HTTP Server [20]. It is, like Node.js an open source project, which can be considered as a plus due to the fact that developers can read the code and learn how to use it. Apache is a webserver with a long history that is being actively maintained, which is a result of a big number of users. Apache is commonly known for its reliability and performance. Since it is the most commonly used web server there is a lot of information about how to set up a server and why to use one. Apache is often used with a scripting language
and a database, the most commonly setup being used is the Linux Apache MySQL Hypertext Preprocessor stack, or LAMP-stack for short. Using this specific stack require knowledge in several technologies, Apache, MySQL, Hypertext Preprocessor (PHP) and a bit of Linux aswell. Since Apache is the most popular web server and the LAMP-stack one of the most popular stacks with apache, the need for knowledge in several technologies might not be a problem. However the more code that is bound to a specific stack is more code that has to be rewritten and cannot be reused.

2.2.3 Socket.io

Socket.io is a library designed for real time web applications, and it claims to feature "the fastest and most reliable real-time engine" [21]. The documentation provided by the developers is very minimal, however they provide several examples on how to use the framework. In almost all of these examples Node.js is being used. So the developers of Socket.io indirectly encourage to use Socket.io with Node.js. This framework is also used in many projects, such as Microsoft Office, Yammer and Trello [21]. That it is being used in big projects is a plus that indicates that it at least has potential to be really good. The realtime framework also has the ability to create rooms for clients which is one desired functionality for the server, as discussed in section 2.2. A quick look at the documentation reveals that the framework focus on delivering one thing and keep it simple.

2.3 Text difference

One essential part of the platform is the visualisation of the differences between two texts. It will help users to easily locate and compare different choice of word, which will make the contrast between the text more visible. Comparing texts has been done for a long time and the algorithms used are similar to one and other. How to visualise the differences are more open.

2.3.1 Line by line

One area where visualising difference is heavily used is in integrated development environments (IDE) used when programming. Programmers use diff engines to visualise the difference between different versions of code. These diff engines often show the differences line by line, meaning that if there is a single difference on a line the whole line will be marked as a diff. To visualise this most IDE's show both versions of the line with background colours, one colour for the original line and another colour for the line that differs.
Figure 2: Visualise differences line by line can result in visualizing trivial differences.

Figure 3: Visualise differences line by line done right.

Indenting is a big part of writing code, it enhances readability and is even required by some programming languages. Comparing and visualising differences line by line might therefore seem like an good and easy way to visualise the differences for the target users. However for programmers the indenting is often not essential when comparing differences in code, since different sets of code can be interpreted the same while have different indentation.

Figure 2 shows how the git repository service Github visualise differences line by line. The only difference between the two versions of the file ”5.5.fps” is in the indentation, more specifically a single space. The code will be interpreted the same by the compiler, therefore the behavior will be the same. That difference does not tell a programmer any essential information about the code. Most programmers would most likely want to use the diff engine to understand why the behaviour of a program has changed, not how the indentation in the code has changed, figure 3 shows how visualising differences line by line can be helpful.

Line by line can be useful for comparing code, but would it be useful when comparing plain text? Displaying multiple versions of a line would hinder the readability of a text, therefore not suiting plain text. However if there was an ability to hide and displaying the differential line, it would not hinder
the readability. Another fix would be to fully display both text and mark the differential lines in both texts. Displaying a line as a diff might be too much. For instance if there is only one word that differs on a line with two sentences a user might interpret that both sentences differ, which is not the case.

2.3.2 Character by character

In contrast to visualise the differences line by line, there is character by character, which is marking only the characters that differ. This is perhaps the easiest way of visualise the differences, due to the fact that there are several algorithms that compare strings character by character. One is the Levenshtein distance, which is sometimes referred to as the edit distance [22]. Only visualise the character that differ would probably be easy to implement and easy for developers to understand. For users it might be a little too simple, since marking up small parts of a word might hinder readability.

**Example 1**

```plaintext
string A = "The bar is pink."
string B = "The car is pink."
```

Look at example 1, here it looks quite okay when it is only the first character in the words "bar" and "car" that have a background colour. The reader might stumble a little but most will be able to read the text just fine, especially when the diffs are right above each other. Now take a look at example 2.

**Example 2**

```plaintext
string A = "The bat in a house"
string B = "The cat with a hat"
```

Most people will stumble when having these many characters marked up so close to each other. It is perhaps most obvious at string B’s third word "with", which is marked up both sides of its letter i. The markup colour is very important, using a darker colour would give too little contrast between the background and the black text. Using a light colour as in example 1 and 2 makes the text visible. Using different colours for the corresponding differences would perhaps make it more clear that "b" in bat and "c" in cat is one diff. And that "ouse" in "house" and "at" in "hat" is another diff. However since there is the possibility to get a word marked up several times it might not be a good idea, since multicoloured words could get even more distracted for the reader, as shown in example 3.
3 Implementation

3.1 Database

The choice of database fell on CouchDB for its flexibility and use of programming language, JavaScript which most developers should have seen. The implementation centres around three different types of documents: user, class and presentment. These documents have a relationship between each other, a relation similar to the ones seen in a relational database. This design choice was done to keep the number of entries in each document down, for easier maintenance. As discussed in section 2.1.2, there are some data entries that are required for the platform to work. These relations add some entries to that list. Luckily, all database management can and should be handled through the server which makes it easy to check if a request that will manipulate the database satisfies the requirements of a document.

3.1.1 User documents

The only current kind of possible user is a teacher. A teacher can create classes, manage discussions in these classes, and create and edit presentments. Since presentments are stored in different kinds of documents and don’t need any connection to a teacher, all information stored in the user document regards the teacher and a connection to his/her classes corresponding class document. This design choice results in a minimal user document as shown in figure 4.

It was discussed to use logins for students as well and have the user document use a data entry called ‘user role’ to determine if the user was a teacher or a student. This idea would require each student to create an account and be logged in to participate in a discussion. This approach would give the possibility to save each student’s answers. This idea was dropped for two reasons, the first one was simplicity, the argument that students might be more willing to participate if they could do so through a url, rather than have to register an account, log in and apply for the right class. The other reason were integrity. Without any distinct argument that saving student answers would ease the teaching or learning experience for any user, the decision to protect the students’ integrity was of higher priority.

Example 3
string A = "The cat in a hat"
string B = "The cat with a hat"
When handling user information in databases the need for different types of administrators is inevitable. However, this is out of scope for this project.

3.1.2 Class documents

The classes are stored in documents of type "class". A class document contains the data entries: "ID", "Type", "Name", "Teacher" and "Discussions". The data entry "Type" is as mentioned before an entry to determine what type the inspected document is. "Name" is just the name of the class. The "Teacher" entry is the ID to the user document that corresponds to the class teacher. "Discussions" is a set of all discussions ever held in the class. The discussion set has several keys with a value. The key is the date and time when the discussion was created, and the value has three data entries, "Presentment", "Active" and "Answers". "Presentment" is the ID of the document of the presentment used in the discussion. "Active" is a variable that can either be true or false, and it is used to determine if the discussion is still active. It can at most be one active discussion in each class. "Answers" is another set that store each submitted answer. Figure 5 shows the data entries in a document of type class.

3.1.3 Presentment documents

The presentment documents store discussion materials. The documents have two different versions of a text, a question about that text and a set of answering alternatives, shown in figure 6.

Having the presentments in separate documents, finding and maintaining specific presentments should be not be a problem for an administrator. Using a flexible NoSQL database adding support for saving statistics should not be
any problem either. For instance, each presentment document could contain all submitted answers from all discussions with that presentment, which could be compared to answers from specific classes.

3.2 Server

For the server the first choice stood between Apache and Node.js. The choice fell on Node.js for several reasons. Firstly its flexibility, with the NPM ecosystem there are several frameworks that can be included to ease further development. Secondly the language, it uses Javascript which most likely will be recognized by the next developer. The third is that it appears more fun, this might seem unessential and for some people it might be. But having fun developing can be a strong motivation to keep on writing code and develop a better platform. With all possible frameworks provided through NPM Node.js appear more fun then Apache.

Socket.io was chosen to be used for real time updates between clients. It was chosen because of its simplicity and ease of use. The ability to use Socket.io’s rooms was also a big factor when deciding to use this real time abstraction layer. The rooms helps when deciding which version of a presentment the students should get, by counting the number of clients in a room, every other client (user) should get version B instead of version A.

Many actions done by the clients are being done through a HTTP API provided by the server. For instance, a client can request a presentment,
create a discussion or submit an answer to a question. As expected these often require the server to access the database, which is done through a framework available at NPM called nano which is a minimalistic CouchDB driver for Node.js. In the case of an incoming vote the server will broadcast the new score to each client in the same room.

3.3 Graphical user interface

The graphical user interface was designed to be simple and intuitive.

As mentioned before the goal of the platform is not to deliver a solution for managing administration, documentation and evaluation, but rather give support while learning critical thinking. This leads to a much simpler platform which can scale down the graphical user interface and make it simple and intuitive. The graphical user interface will be discussed in two parts, one for each kind of user, Teacher and Student. Where the interface for the teachers have focus on keeping all the core functionality easy to hand. While the interface for students is more straightforward with clear steps to go with the corresponding phase in the learning methodology.

3.3.1 Teacher

There are two main objects that teachers can control with the platform, classes and presentments. These are always accessible for the teacher at a menu on the left hand side of the platform, see figure 10.
By choosing a class from the left hand list, the teacher will get an overview of the discussions that the class have had. These are represented as small cards with the most essential information visible: date, presentment, question and a pie chart with the distribution of the submitted answers, see figure 9. By clicking one of these cards an overlay with more detailed information about selected discussion will appear. In excess of the information from the card, this overlay also contains a Uniform Resource Locator (URL) for students so they can participate, the option to archive the discussion (if and only if it is active) and both texts of the used presentment, see 7. The difference of the texts are visualised by the implemented diff engine, see section 3.5.

Creating or editing a presentment has the same interface, one section for editing the two versions of the text and one section for managing the question with its alternatives. Editing text is done in two resizeable text areas to give the user the ability to optimise their workspace. While editing the texts the diff engine will run in the background and mark up the differences in real time. This will help the user to evaluate his/hers texts. Below the text editing section is the section for manage the question. The user will have to add a question and at least two alternatives, with the ability to add and remove extra alternatives, see fig 8.
Figure 8: Overview of the presentment edit.

Figure 9: Overview of discussions in a specific class.
3.3.2 Student

The student side of the platform aims to be simple and easy to use. This approach aims to get students to participate in higher numbers. By using a URL students can take part of the discussion basis, and easily participate in an upcoming discussion. Once entered a discussion one version of a text will be presented with an multiple-choice question about that text, see fig 11. By answering this question a pie chart of all submitted votes will be shown. The purpose is to show that students can interpret the described problem differently, which may also increase their motivation to discuss the text. Lastly students have the possibility to see both versions of the text with their differences colour coded, and see the pie chart once again, see fig 12.

It is important to point out that the idea is to let students discuss the text. Between which step the discussion should take place for the best learning experience is up for discussion. Little or no research has been done on the subject. A suggestion is to have two discussions one directly after the multiple-choice question and one after the students have seen both versions of the text. By having two discussions the students might realise why they had different thoughts on the text. This will give them a hands-on experience of why the lack of critical thinking might steer peoples point of view.
Imagine that the United States is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume that the exact scientific estimates of the consequences of the programs are as follows:

If Program A is adopted, 400 people will die. If Program B is adopted, there is a one-third probability that nobody will die and a two-thirds probability that 600 people will die.

Which of the two programs would you favor?

- Program B
- Program A

Submit vote

Figure 11: First step when joining a discussion as a student.

Imagine that the United States is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume that the exact scientific estimates of the consequences of the programs are as follows:

If Program A is adopted, 300 people will be saved. If Program B is adopted, there is a one-third probability that 600 people will be saved and a two-thirds probability that 0 people will be saved.

Figure 12: Students can see the difference in text with color coding.
3.4 Architechtual design choices

When the teacher loads the presentments from the database, it also loads the content of each presentment. Once the content is loaded, it is stored in a hidden div under the corresponding presentment. This will lower the number of requests from the server to the database, and increase speed while inspecting a presentment. It also opens up for future uses. For instance visual recognition could be used to help the teacher find their previously created presentments, or presentments they have used before. This design choice works fine in a small prototype, but for a finished product one would have to take scalability in regard. Loading a couple of hundred names of presentments is no problem, but loading them with their content is another matter, in particular if they have long texts. A fix would be to limit the number of presentments to be loaded, and implement a function to load in more. Alternatively, a finished product could load the content just in time (when the user needs it).

3.5 Calculating difference between texts

One of the requirements of the platform is that the user should be able to see the difference between two texts, provided from the database. To show the user these differences, there needs to be rules for what should count as a difference. For instance a difference might be declared as when words in a position do not match. Imagine two strings \( a \) and \( b \), where \( i \) is a number and \( a[i] \) represents a word at position \( i \). Then a difference could be declared as \( a[i] \neq b[i] \). This would perhaps be one of the most correct definition, however it might not be the best.

**Example 4**

TextA = "Hi! My name is Manfred"
TextB = "My name is Manfred"

Consider example 4, if the difference is declared as \( a[i] \neq b[i] \) then every word would be a diff. While some people would argue that it is only the word "Hi!" that is a diff. Another approach would be to declare a difference from, textA and textB as the minimum number of edits to go from from textA to textB. Where an edit can be one of the following things:

1. deletion of a word
2. insertion of a word
3. substitution of a word
With the minimum edits approach example 4 will result in a minimum number of edits of one insertion of the word "Hi!". Minimum indicates that there might be other solutions that are not optimal. For instance a non-optimal solution is to substitute each word in TextA and then do an insertion of the word "Manfred" at the end. These types of problems are called optimisation problem. Dynamic programming is a method suitable for solving optimisation problems [22]. Like the divide-and-conquer method, dynamic programming solves a problem by using solutions to subproblems. The main difference from divide-and-conquer is that dynamic programming does not have to solve a subproblem several times. With dynamic programming the solution to a subproblem is stored in a table, so that if the same subproblem appears again the solution can be looked up.

3.5.1 Algorithm

With the help of dynamic programming the two texts get a score called "edit-distance" [22]. The score represents the minimum edit going from TextA to TextB. This is done by first removing potential special characters and tags, like new-line \( \text{"n"} \), tab \( \text{"\t"} \) and HTML tags. This was done since the difference in format is not interesting, but the difference in words. After removing special characters and tags, a matrix \( m \) is created with the following characteristics:

\[
\begin{align*}
\text{size}(m) &= \text{numberOfWords}(\text{TextA}) + 1 \\
\text{size}(m[i]) &= \text{numberOfWords}(\text{TextB}) + 1
\end{align*}
\]

Where \( 0 \leq i \leq \text{size}(m), \) \( \text{numberOfWords}(a) \) returns the number of words in the string \( a \) and \( \text{size}(m) \) returns the size of \( m \). The plus one is for being able to represent the empty string. Each cell has a value representing the edit-distance between two substrings.

Example 5

- TextA = "The cat in the hat"
- TextB = "The black cat with a hat"

Example 5 compares string TextA and TextB, for instance cell \( m[1][1] \) should represent the edit-distance value of the substrings "The" and "The" which is equal to 0, since they are exactly equal. Cell \( m[1][2] \) should represent the edit-distance of the substrings "The" and "The black" which is equal
The cat in the hat

Table 1: Matrix used for string comparison, filled with initial values

<table>
<thead>
<tr>
<th></th>
<th>ε</th>
<th>The</th>
<th>cat</th>
<th>in</th>
<th>the</th>
<th>hat</th>
</tr>
</thead>
<tbody>
<tr>
<td>ε</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>The</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>black</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>cat</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>with</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>hat</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

to 1, since an insertion of the word ”black” is enough. See table 2 for full matrix of edit-distances to example 5.

Some values in the matrix are given namely the first row and the first column. They represent the edit-distance compared with an empty string. In example 5 the first row represents the edit-distance between the empty string and all substrings of TextA. And the first column represents the edit-distance between the empty string and all substrings of textB. Therefore will the edit-distance be increasing, since there will require an insertion for each word in every substring. Every other cell will get an initial value of 0. See table 1 for initial values of example 5.

The rest of the matrix is then filled column by column, with the value defined as follows:

\[
f_{a,b}(i, j, m) = \begin{cases} 
m[i][j] = m[i - 1][j - 1] & \text{if } a[i] = b[j] \\
m[i - 1][j] + 1, & \\
min = \begin{cases} 
m[i][j - 1] + 1, & \text{if } a[i] \neq b[j] \\
m[i - 1][j - 1] + 1 & 
\end{cases}
\end{cases}
\]

Where a and b are strings, \(0 < i \leq |a|\), \(0 < j \leq |b|\) and \(m\) is a matrix with \(|a|\) number of column and \(|b|\) number of rows.

Once the matrix is filled, the actual differences can be extracted with help of the original strings and the matrix. This is done by traversing the matrix from the last cell to an cell with value 0. The rules for traversing the matrix \(m\) holding edit-distance from string \(a\) to string \(b\) is as follows:
Table 2: Fully filled matrix used for String comparison

<table>
<thead>
<tr>
<th></th>
<th>ᵇ</th>
<th>The</th>
<th>cat</th>
<th>in</th>
<th>the</th>
<th>ᵆ</th>
</tr>
</thead>
<tbody>
<tr>
<td>ᵇ</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>The</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>black</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>cat</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>with</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>a</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>hat</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 3: Traverse path in the matrix

\[
f_{a,b,i,j,m} = \begin{cases} 
m[i - 1][j - 1] & \text{if } a[i] = b[j] 
m[i][j - 1] & \text{if } m[i][j] = m[i][j - 1] + 1 
m[i - 1][j - 1] & \text{if } m[i][j] = m[i - 1][j - 1] + 1 
m[i - 1][j] & \text{if } m[i][j] = m[i - 1][j] + 1 
\end{cases}
\]

Where \(0 < i \leq |a|\) and \(0 < j \leq |b|\). The first case represents when the words are equal, no edit needed. The other three cases represents the possible edits, insertion, substitution and deletion (in that order). Once inspecting a cell with the value 0 all edits have been found. See table 3 for the path when traversing example 5.

As mentioned before, the difference was declared as the edit-distance between words, which raise the question of how words are declared. In the working implementation words are declared as substrings separated by spaces. This declaration makes the implementation very easy and it works, but it is not necessarily the best declaration of words.
Example 6

TextA = "Hi, Manfred!"
TestB = "Hi, Manfred?"

In example 6 the diff would be on the substring "Manfred!" and "Manfred?". But what if the desired behaviour would be to only mark the substrings "!" and "?" as a diff. Or what if one would mark the whole sentence as an diff, since textB is a question and TextA is not. Depending on the desired behaviour the declaration of word might have to change. However since the main point of this project is not to build an text comparison tool, no further investigation was made. It is worth mentioning though that in a fully developed platform details like this should be investigated more.

3.5.2 Visualising text difference

One source of information provided to humans comes from the light that hit the human eye. Once the light reaches the eye a perception starts to shape. This is called visual perception. It help humans to recognise things as alert symbols and buttons in a computer program to name a few.

In the early twentieth century a group of psychologists notified five 'laws' of visual perception, that humans are born with. These laws are known as the 'Gestalt Laws'. One which is known as 'similarity', says that humans tend to group things together if they are similar [23]. This can take form in shape as well as in colour. Figure 13 shows how the similarity law works with figures of the same shape. Most people tend to group together the figures in triangles and diamonds. However when looking at figure 14 most people tend to group the circles by color. This similarity law is used when visualising the difference in two texts for the user.

In section 2.3 two different ways of visualising differences was brought up, the line by line and the char by char method. Neither of these methods was evaluated as optimal for this use case. Since the presentments will use different word of value, and mostly differ in words (not total lines) the choice to make a third option was made. One that will color code words.

As discussed in section 3.5.1 the difference in the two texts uses the edit distance, which represent the minimum edit to go from one text to another. These edits can be one of three different types, insertion, substitution or deletion. To display this difference the similarity law is used. By showing both texts and colour coding the difference in both texts, the user should intuitively group together the the words in with the same colour in the seperated texts.

Figure 15 shows how the similarity is used in this project. The colour coding works with the three previously mentioned possible edits, however
Figure 13: Humans tend to group together figures of similar shape

Figure 14: Humans tend to group together figures of similar color
Imagine that the United States is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume that the exact scientific estimates of the consequences of the programs are as follows:

If Program A is adopted, 200 people will be saved.
If Program B is adopted, there is a one-third probability that 600 people will be saved and a two-thirds probability that no people will be saved.

Imagine that the United States is preparing for the outbreak of an unusual Asian disease, which is expected to kill 600 people. Two alternative programs to combat the disease have been proposed. Assume that the exact scientific estimates of the consequences of the programs are as follows:

If Program A is adopted, 400 people will die.
If Program B is adopted, there is a one-third probability that nobody will die and a two-thirds probability that 600 people will die.

Figure 15: The similarity law being used to show difference in two texts

If a text has a deleted word. And some other changes, it is hard to show the changes in the best way.

If a text has a word. And some other changes, then it is hard to know what is the best way to show the changes.

Figure 16: The word ‘deleted’ is colour coded in the left text, however it has no corresponding colour coding in the right text

It colour code edits in a streak and not separated edits. Looking at the difference ‘be saved’ and ‘die’ displays this since the edit is one substitution of the word ‘be’ to the word ‘die’ and one deletion of the word ‘saved’. This design choice was made since colour coded individual edits can easily be experienced as cluttered, due to the additional colours. This strategy to colour code depending on edit streak has a potential weak point that occurs when there is only one deletion or only one insertion. Figure 16 shows that the word ‘deleted’ in the left text has no corresponding colour coded word in the right text. This might be a problem for some users, who are intuitively trying to group the word with another word in the corresponding text.

Potential designs to address this issue were investigated. For instance the deleted word could be included in the other text but crossed out or faded. Alternatively the blank space where the missing word was could be colour coded. Potential strategies dealing with missing words were abandoned, since then it wouldn’t show the text as the writer intended, and the opportunity to read the original text would be lost. The strategy to colour code the space where the word would have been indeed make it feel like there were something missing. However it did cause an annoyance rather than clarifying the difference between the texts.
4 Discussion

4.1 Future work

For this platform to be a useful tool when teaching critical thinking there has to be a number of quality presentments. Teachers have the ability to create their own which will be shared to all. The probability that teachers start using the platform will most likely be higher if there are some quality presentments there from the start.

As discussed in section 1.3 there has not been any security aspect when creating this prototype. Before releasing a full scale version these security aspects would need to be studied and some form of security layer would need to be implemented.

The support for creating and editing presentments needs to be more consistent, at testing the caret would occasionally make a jump. This is most likely the effect of running the differential algorithm and color code text, at the same time as the user is inputting additional text. A solution for this is needed, and would probably be possible with an algorithm running in the background only updating the texts at safe times.

4.2 Conclusion

The core goal for this project was to develop a prototype for a platform with one single task, give support during learning of critical thinking. By using learning methods based on collaborative learning and code with modern frameworks and techniques the goal for this project has been achieved.

As described in section 1.2 the learning method is based on collaborative learning in shape of discussions based on text with different versions. Students are divided in two groups, one group reads one version and the other group reads the other version. Students are then asked a multiple choice question and after they have answered anonymously they can see how the class as a whole have answered, discuss and see both versions of the texts. This will have them reflect over why they answered as they did. This collaborative learning method has been achieved in several steps with different frameworks. The server has been written with the node.js framework discussed in section 2.2.1. In pair with Socket.io, Node.js (discussed in section 2.2.3) has been used for creating rooms for the classes, dividing them in groups, give real time updates of the answers. Node.js also helps with setting up a route per room so that students can easily join a discussion with a URL without needing to create an account.

The presentments and how the versions of the text differs to one and
other are a central part of the platform. One essential goal was therefore to display the difference between texts in a clear intuitive way. After looking at several diff engines it was concluded that not one of the existing diff engines was optimal for this case. A new diff engine had to be implemented, it was implemented in pure javascript which has one major advantage. It can be easily edited and reused since it does not rely on any other framework. As described in section 3.5.1 the implemented algorithm uses dynamic programming helping speed when comparing large amount of text.

All these presentments has been stored in a CouchDB database. The advantages and disadvantages of using CouchDB has been discussed in section 2.1.2, one of being that it is flexible, making the platform less sensitive to changes in data structure.

In retrospective the choice of each framework and technique has been made with care. Most if not all goals could have been made with other alternatives. However since one aspect of the project was to make it as easy as possible for other developers to continue to develop the platform or develop add-ons, the choices become more important. All frameworks used in this project are built with javascript, hopefully making it easier for developers writing code and less on trying to understand syntax.

With the exposure from media increasing [1], and the acknowledgement of the importance of critical thinking [7] this project is in the right period of time. As of October this year the new minister of swedish upper secondary school, Anna Ekström talked about the importance of critical thinking, and how IT should be used as a tool for pedagogy. Statements such as "Pedagogy should be master and IT its servant" and "I think there are gigantic advantages in knowing a great deal about source criticism..." [24] make it clear that there is a demand for tools like the prototype developed in this project.

The bigger question is if the approach used will give good support during learning/teaching of critical thinking. The experience for students using this platform is more or less straight forward, enter a url, read the text, answere the question, view both texts and discuss how and why classmates answered as they did. The use of collaborative learning has been shown to be beneficial while learning critical thinking [10]. They will also be a part of an example of how words of value can nudge people in different directions. Ease of use, collaborative learning and example of why critical thinking is important should make this an appropriate tool in the view of the students. However it is important to remember that the core of the learning experience is the presentment, if the presentment is not good the students will not have a great experience. The teaching experience of the platform circle more around preparation before the discussion then the discussion itself. The presentment is essential for a good learning experience and the
platform help the teacher choosing, creating or editing a presentment. During the discussion the platform is more or less redundant for the teacher, only showing answers of the question and difference of the two texts, meaning they can take a part of the discussion if they would like to or at least make sure students stays on topic. Hopefully teachers found the platform easy to use and it gives them the visual feedback needed for setting up a good basis for discussion.

With a demand for this tool, a good learning approach and focus on one thing give support while learning critical thinking, this could be a successful tool. But will require more testing and a full scale version of the platform.
References


