Content and Activity Based Friends Suggestion in an Annotation System

Jana Taha
Department of Computer Science
An-Najah National University
Nablus, Palestine, P.O.Box: 7
www.najah.edu
tahajana057@gmail.com

Ramah Hawash
Department of Computer Science
An-Najah National University
Nablus, Palestine, P.O.Box: 7
www.najah.edu
ramahahawash2003@gmail.com

Ahmed Awad
Department of Computer Science
An-Najah National University
Nablus, Palestine, P.O.Box: 7
www.najah.edu
ahmedawad@najah.edu

Amjad Hawash*
Department of Computer Science
An-Najah National University
Nablus, Palestine, P.O.Box: 7
www.najah.edu
amjad@najah.edu

***Abstract*—Annotation systems enhance user collaboration by allowing structured sharing, commenting, and content discussion. They improve clarity, ensure contextual relevance, facilitate real-time or asynchronous interaction, and foster a collaborative environment. The success of collaborative projects relies on finding annotators who share similar interests, as this ensures that feedback and contributions are relevant and meaningful. This project aims to create a text-based annotation system that will allow users to apply similarity metrics between their annotations and those of others to identify the most suitable annotators. Additionally, the program will enable users to take online tests automatically generated based on a selected term from the annotated text. The system recommends appropriate YouTube materials and identifies annotators who score highly in comparable quizzes based on the results of the quizzes. Users can search for and chat with others, promoting an enriched and collaborative learning experience. Experimental tests in this work reflect a promising enhancement in user collaboration and knowledge dissemination.**

***Index Terms*—Natural Language Processing, Annotation Systems, Similarity Measures, Online Quizzes.**

I. Introduction

The process of annotating digital documents and media with notes, highlights, comments, and other interactive input is known as digital annotation [1]. This feature is essential for user collaboration since it makes it possible to interact with shared content in a way that is more dynamic and interesting. Digital annotation improves teamwork by facilitating real-time sharing of ideas, feedback, and improvement suggestions by allowing several users to annotate a document at the same time [2]. In addition to expediting the decision-making process, this collaborative method guarantees that a variety of viewpoints are taken into account, which eventually produces more effective and refined results. Moreover, digital annotation solutions frequently provide functionality for tracking and organizing changes, which facilitates the management of complicated projects and keeps a clear record of the collaborative process [3].

Corresponding Author: amjad@najah.edu

There are many different types of annotations, and each one has a specific function in improving understanding and communication. One popular technique is to highlight important text passages so that they stand out and are simple to find for revision [4]. Textual comments enable readers to add observations or notes straight onto the page, allowing for in-depth conversation or feedback. Similar in purpose, sticky notes are also incredibly versatile, just like their physical counterpart. They can be placed wherever on the page. Users can easily circle, underline, or sketch concepts on the page using drawing and freehand comments, which is very helpful for visual learners and situations when visual emphasis is needed. With the use of hyperlink annotations, users can access similar documents or other resources by clicking links embedded within the text. A multimedia element is added by audio and video annotations, which let users record and add spoken remarks or visual demonstrations. These can be very helpful for breaking down complex information or giving feedback a more individualized touch. Every kind of annotation improves the process of collaboration by providing a variety of ways for users to interact with and contribute to the information [5], [6].

Sophisticated algorithms are used in recommendation systems to offer material, services, or products to consumers based on their interests and actions [7]. These systems are essential to contemporary digital experiences since they personalize the distribution of material, greatly increasing user pleasure and engagement. Recommendation systems make interactions more relevant and effective by predicting what a user could find interesting or useful by analyzing data such as past interactions, purchasing history, and demographic data [8]. While they keep users engaged in media streaming services by recommending movies, music, or articles based on their preferences, they also drive sales in e-commerce by suggesting products that match user interests. Recommendation systems give companies a competitive edge by enhancing client retention and revenue generation in addition to raising user satisfaction. They are crucial because they make it easier for people to find what they need or find new interests by bridging the gap between users and the enormous diversity of content that is available. This enhances the user experience as a whole [9].

An annotation system becomes much more useful and appealing when social aspects, such as friend connections, are included in it. Annotation systems that provide friends feature to enable users to interact and exchange comments, opinions, and insights in a more customized way [10], [11]. This social component encourages a cooperative learning and working environment where users can gain from a variety of viewpoints and levels of experience, improving the breadth and caliber of their output. Additionally, it fosters a feeling of solidarity and community, which enhances the interaction and fun of the annotation process. Friends can help make an annotation experience more lively and productive by offering support, helpful resource sharing, and constructive critique. Moreover, users are encouraged to be more detailed and attentive in their contributions when they know that their friends can see and interact with their annotations. Essentially, the inclusion of friends in an annotation system turns an isolated activity into a cooperative and socially stimulating one that promotes better results and increased engagement [12].

The work conducted here is related to designing a textual-based annotation tool that enables users to collaborate by submitting digital annotations of their interests. To maximize their collaboration, the tool enables the friends' suggestions based on two things: related thoughts and the amount of knowledge. Cosine similarity is used to suggest friends to users based on comparing their annotations and producing a ranked list of users. Moreover, the amount of knowledge and experience a user could have makes him/her attractive to other users; hence, the tool suggests them to others. With the increased number of users and the possible tremendous amounts of submitted annotations, there will be a possibility that these annotations are biased to some topic. However, this is currently out of our concern. A possible future work could be conducted regarding the amount of bias towards some topics.

The contribution of this work could be summarized in the following points:

- The task at hand involves creating a text-based annotation system that allows users to annotate material on websites and retrieve and see these remarks at any time.
- Connecting users based on ideas and interests by using the cosine, dot product, and Euclidean similarities.
- Creating online tests automatically using a few terms chosen from annotations and fixing tests that have already been administered.
- Making appropriate YouTube video recommendations based on quiz scores.
- Compiling all users' quiz results and utilizing them to recommend other users who score highly.

Although our contribution to this work is related to developing the tool and testing it, the issue related to the recommendation system might develop biases, affecting fairness and equal opportunity for user interactions and collaborations is out of our concern in this work despite it being one of our plans. However, this problem can be investigated by emphasizing a set of solutions like Bias The text emphasizes the importance of bias detection, diverse data collection, fairness constraints, transparency, regular monitoring, and personalization with fairness metrics in a recommendation system. It also highlights the need for fairness metrics to balance accuracy with fairness, ensuring equal treatment across user demographics.

One of the issues related to annotation systems is that the system's effectiveness heavily relies on active user participation, which could be a limitation if user engagement is low. However, the system can mitigate challenges by using historical and passive data to make suggestions, even with low active participation. To increase user engagement plans like gamification, rewards, or integration with other platforms are suggested. In low activity cases, collaborative filtering based on similar users or content similarity can keep recommendations relevant. A hybrid recommendation model is proposed for effective use.

Although Cosine Similarity still has limitations in capturing more nuanced similarities between annotations, the text acknowledges the limitations of Cosine Similarity, suggesting the need for supplementary methods like semantic similarity and word embeddings to provide more nuanced annotation assessments. It also suggests domain-specific adjustments to better capture subtle annotation differences. The text also suggests exploring advanced similarity measures like neural network-based approaches or hybrid models for future research.

The rest of this paper is organized as follows: Related Works is proposed in SectionII. Section III discusses the system architecture of the implemented work while Section IV illustrates the conducted recommendation system in the work. Section V discusses the conducted experimental tests and achieved results. Finally, Section VI concludes this paper.

II. Related Works

Textual annotation tools have revolutionized collaboration by allowing users to highlight text sections, add comments, and engage in threaded discussions. They offer real-time collaboration, version control, and integration with productivity software, ensuring feedback is organized and actionable. These tools support various formats and platforms, enhancing understanding and communication.

Labellerr[1] is a precision-focused text annotation tool for machine learning models, offering features like sentiment analysis, translation, and text classification. Its automated workflows enable labeling large data volumes while maintaining high-quality annotations. Despite its user-friendly interface, it doesn't currently support point cloud or 3D data formats.

Prodigy[2] is a machine learning model training tool developed by Explosion AI, offering active learning, text annotation, self-hosted workflows, and export formats for various data types. Its sleek interface and active learning approach reduce annotation efforts, making it a valuable asset for both novice and experienced users.

Label Studio[3] is an open-source data annotation tool that supports text classification, named entity recognition, and sentiment analysis, with a user-friendly interface and integration with machine learning models for improved annotation accuracy.

Doccano[4] is an open-source text annotation tool with a user-friendly interface, customizable configuration, and annotation options. It supports CSV or JSON data export, supports multiple users, and provides annotation guidelines. However, it may face issues in self-hosted environments and lacks an official API.

Perusall[5] is an online annotation tool that enables instructors to create quizzes based on student annotations, while Hypothesis[6] is a social annotation tool that can be integrated with Learning Management Systems. EdPuzzle[7], a video content tool, allows instructors to add interactive questions and quizzes. Kiami[8] is a digital classroom tool that supports text annotation.

Advanced similarity measures are used in annotation tools and platforms to suggest collaborators. These measures are often part of sophisticated systems rather than commercial tools. Examples include custom academic platforms like Mendeley[9], Zotero[10], ResearchGate[11], and collaborative filtering systems in Learning Management Systems (LMS). These platforms analyze user profiles, publication histories, and annotations to recommend potential collaborators. These systems can also be integrated with similarity-based recommendation systems to find users with similar research interests or publication topics.

VideoAnt[12], EdPuzzle[13], Annotate.tv[14], and Timelinely[15] are tools that allow users to add annotations to online videos, including YouTube videos. These annotations are linked to specific points in the video, allowing for easy reference to exact moments. They also allow for interactive learning experiences with text, links, and images.

To the best of our knowledge, no single annotation method incorporates every historical feature. This motivates us to continue working on the project, which aims to create social networks based on common interests through a customized portal that lets users annotate and comment on text online. It also allows users to engage with other users who have similar interests by using similarity metrics to compare annotated texts and by recommending the most accomplished users to one another based on the outcomes of online quizzes. Users can establish connections with people who share their interests by sharing these annotations. Our system, in contrast to other solutions, combines all of the earlier services into a single solution, facilitating greater user collaboration. This method makes use of annotation's social component to transform a solitary task into a fun and interactive group activity. Our software improves the overall user experience and fosters meaningful interactions and friendships by matching individuals with similar interests.

III. System Architecture

Adding a transparent layer above an already-annotated website—this invisible layer serving as a representation of the tier containing annotations—is the notion behind annotating web content. The client/server architecture of the tool used in this project is web-based. Through a browser plugin, users can annotate texts on a web page. The relevant information—highlighted text, user comment, URL, and creation time—is kept in a dedicated database when a user highlights text, adds comments, and saves the annotation.

Our proposed system is composed of several layers: **Presentation**, **Processing**, **Database**, and **NLP (Natural Language Processing)**. The Presentation Layer handles user interactions through the browser extension and the website being annotated by injecting a special Javascript code, where annotations can be created, viewed, liked, saved, and managed. The Processing Layer, implemented using *Spring*

[1]https://www.labellerr.com/text-annotation-platform
[2]https://prodi.gy/
[3]https://labelstud.io/
[4]https://doccano.herokuapp.com/
[5]https://www.perusall.com/
[6]https://web.hypothes.is/
[7]https://edpuzzle.com/
[8]https://www.kamiapp.com/
[9]https://www.mendeley.com/
[10]https://www.zotero.org/
[11]https://www.researchgate.net/
[12]https://ant.umn.edu/
[13]https://edpuzzle.com/
[14]https://annotate.tv/
[15]https://edtechbooks.org/onlinetools/timelinely

Boot, manages interactions between the Presentation and Database layers, ensuring data is securely stored, retrieved, and managed efficiently.

The Database Layer includes a MySQL database for storing annotations and user information and Redis for caching JWT tokens to secure the system. The NLP layer, implemented in Python, performs cosine similarity to recommend relevant notes based on user annotations, extracts keywords, and provides additional functionalities like recommending YouTube videos and connecting users with others who have excelled in specific topics.

Moreover, users can engage with the system by selecting notes, performing keyword analysis, taking tests, and receiving recommendations based on their performance. They can also interact with other users by following, chatting, and viewing each other's notes. The system's architecture ensures smooth data flow and secure user authentication, maintaining the integrity and confidentiality of user data.

Figure 1 The diagram below depicts the structure of the tool, where the request/response protocol is used to save and retrieve annotations.

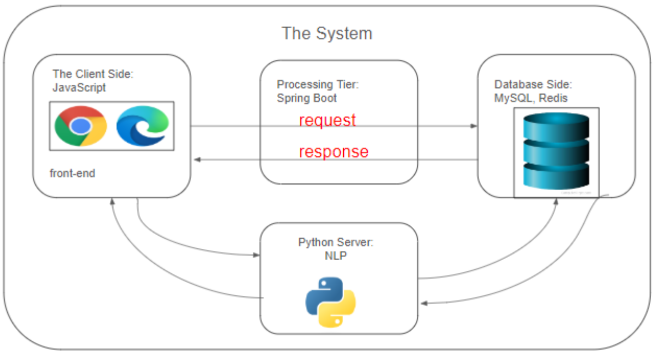

Fig. 1: The architectural design of the tool.

Figure 2 below represents the logical schema of the database represented by an Entity-Relationship diagram. The diagram contains 3 different entities: **User**, **Note**, and **Mark** alongside the entity relationships between them. The user entity reflects all attributes related to users with their credentials, Text is used to save data about the annotated text and the Voice entity is to save data about the vocal annotation itself.

A database server, Python code, and Spring Boot make up the server side of the system. Together, the parts create our intended browser extension, which makes it easier for users to participate in our platform's dynamic text selection and community. With the help of this extension, users can choose text excerpts from a variety of websites and have them effortlessly integrated into our unified online platform. Users can engage, exchange ideas, and create connections based on shared interests here. The cooperative endeavors of these constituents not only facilitate the execution of our inventive annotation mechanism but also actualize our ambition of

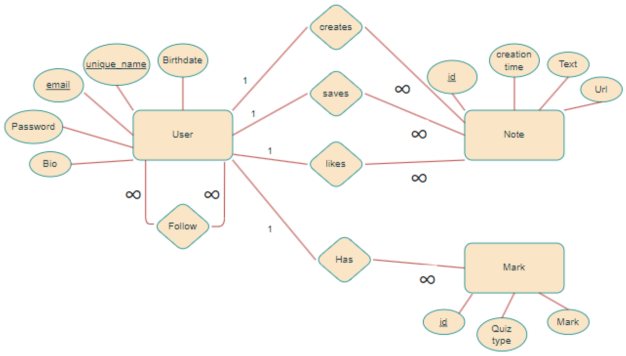

Fig. 2: The ER diagram of the system.

augmenting user engagement and cultivating novel relationships via mutual content encounters. The ensuing subsections provide a thorough discussion of each system component.

Concerning scalability, especially when the amount of users, data, and interactions rises, the annotation system has to be scalable to fulfill users' needs. However, the current system design does not support scalability, although this can be considered one of our plans. A distributed architecture could be used in the system's design to enable the distribution of annotation jobs among several servers or cloud instances, hence ensuring scalability. Performance can be enhanced in situations of heavy demand by utilizing database sharding, caching, and load balancing. Elastic scaling, which automatically modifies resources in response to surges in consumption, is a feature of cloud-based systems. Furthermore, managing big datasets requires making sure the annotation system facilitates effective data retrieval via indexing and optimized queries. Managing concurrent user interactions in real-time without sacrificing speed becomes crucial when collaborative annotations are an option.

Considering privacy and security issues related to storing and managing user data and annotations, users have their credentials. Each of which has its username and password, so that the process of annotation creation and submission is done after the login process takes place. According to this, users are not allowed to update other's annotations. They only can view others' annotations when they visit the websites that are previously annotated by others.

A. Google Extension

Our solution for a user interface between the system and users is a browser extension. Users can highlight any text on any website, add comments, save the annotations, and log in thanks to the JavaScript code that is incorporated in the plugin. These annotations are kept in a MySQL database together with the creation time and URL of the text fragment. The recorded annotations can then be retrieved and managed by users using a web application written in Vue JS.

Users can choose the text they want to annotate after visiting an intriguing website. The selected text and a text box for a remark are shown in a pop-up window that appears when they click on a system extension icon in the extensions section of the Google Chrome browser. Clicking "Save" completes the commenting process for users. The generated annotation and its related data are then transmitted by the Google Extension via communication with Spring Boot. The pop-up window used to produce the annotation, where content is copied from the source website to the pop-up window, is shown in Figure 3 below. This is where users can annotate content with comments and save it by clicking the save button.

Fig. 3: The pop-up window to create an annotation that appears upon the selection of some text.

From Figure 4, the activity commences with a user highlighting textual content on a website, depicted by the function `highlight(text)` within the *Google Extension* object. This action triggers the execution of the `display(text)` function within the *Pop-Up Window* object, where the selected text is showcased. Upon invoking the `display(text)` function, users are presented with a dedicated space to input their comments. Upon completion, clicking the "Save" button initiates a sequence where the *Google Extension* interfaces with a backend system powered by *Spring Boot*. This backend, as represented by the function `saveData(text, comment, text_fragment_url, creation_time)`, ensures the data's persistence by interacting with a MySQL database, securely storing the annotated information. This systematic approach not only facilitates efficient annotation of scientific content but also ensures robust data management and accessibility for research collaboration and reference.

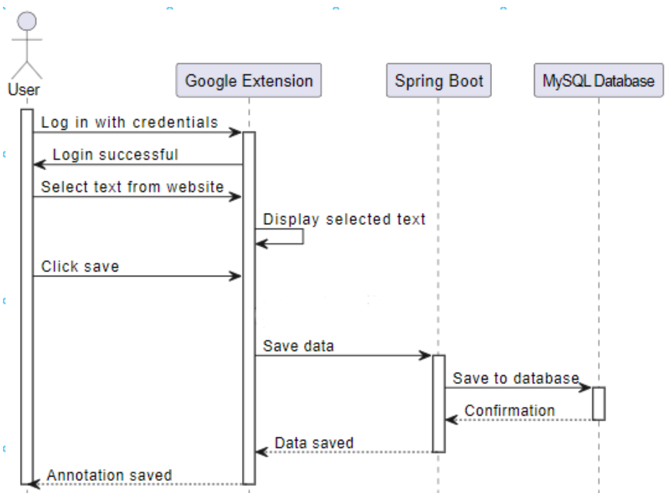

Fig. 4: The Sequence Diagram for creating new annotations.

Annotation retrieval is demonstrated in Figures 5 and 6. To start the procedure, a user must join in to the website and go to their profile page, where all of their notes are shown (Figure 5). A user can look up a desired person's name and visit their profile to see annotations posted by other users. Annotations are listed on each profile, and users can click the "Go to URL" button. The original annotation page (Figure 6) with the specified annotation text highlighted is the page that the user is taken to when they click this button.

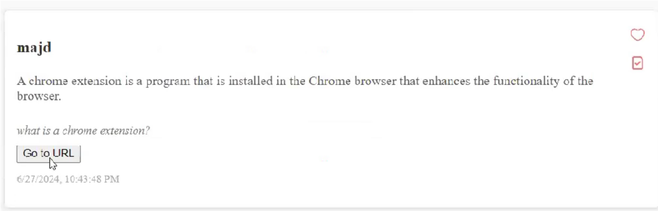

Fig. 5: A sample annotation made by some user in the user profile page.

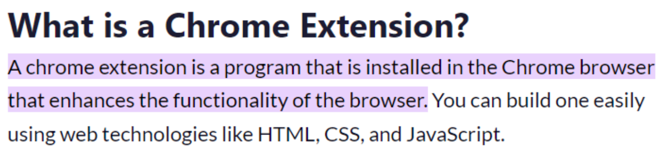

Fig. 6: Original website page showing the highlighted text of the annotation and user's comment.

B. Spring Boot

The backend system is a Spring Boot application that offers a RESTful API for CRUD (Create, Read, Update, Delete) actions on user data, including marks and annotations. Communication between the database and the front-end systems is facilitated by this API. The system uses JSON Web Tokens (JWT) for authorization and authentication in order to guarantee safe connection. By utilizing a stateless design, this method improves performance and scalability.

C. Database

MySQL is utilized as the relational database management system (RDBMS) to facilitate the archiving and retrieving user data and notes on the website. It is an RDBMS-class relational database that is available for free.Structured query language (SQL) is supported by MySQL for relational data management. It offers many indexing strategies to maximize query efficiency, facilitates transactions for data integrity, and enables effective data storage and retrieval via SQL queries. Because of its well-known dependability, scalability, and simplicity of interaction with online applications, MySQL is a popular choice for managing structured data on websites, including user profiles and notes. Furthermore, Redis is used by us to cache user tokens. Redis is an in-memory data structure store that may be used as a message broker, cache, and distributed in-memory key-value database. Because of its high-speed operations and support for a variety of data structure types, it is perfect for caching user tokens, which improve system efficiency and provide prompt access to authentication data.

D. Python Code

Several crucial capabilities in our system were enabled by the Python code. First, preparatory measures like eliminating stop words and lemmatizing words (like changing "better" to "good") were part of the recommendation system. Sentences were then converted to vectors using a trained model named `w2v_model`, and textual similarity was measured using cosine similarity. When it came to making recommendations, the system gave the user's annotations priority if they were accessible; if not, it used the user's stored or liked annotations or presented random annotations. The user's notes and every note from unfollowed users were analyzed to identify the most pertinent annotations. Furthermore, the Python code used a pre-trained model named `key_bert` to tokenize annotation keywords. To run a test, users could choose these keywords. The process of creating tests was handled by an external API known as Arlinear, which produced tests according to the topic, quantity and kind of questions, and degree of difficulty specified in a POST request. By integrating with Gemini AI, which returned the category based on the test subject and pre-existing categories in the database, the system was able to identify the test category and save the user's mark. After then, the mark was added to the database. In order to encourage peer support and learning, the system also suggested that users connect with the top users in the relevant category if they had a low grade.

E. Chatting

Chat Engine IO is a third-party service that facilitates the chat feature. In order to add a user to the chat system, the system sends a POST request to Chat Engine IO upon user registration. The smooth communication within the application is made possible by this connection, which guarantees that all registered users are automatically enrolled in the chat service.

F. Vue JS

The front end of the application was constructed using the progressive JavaScript UI framework Vue.js used to create many pages with dynamic and interactive features. The main pages and their features are as follows:

1) The "Liked Notes" tab lets users quickly manage and revisit their preferred content by showcasing the notes they have loved.
2) The Saved Notes Page offers a handy means of accessing crucial information by showcasing the notes that the user has stored for future use.
3) Dashboard: The user's following list and the notes from the persons they follow are shown on the dashboard page. A consolidated view of the user's network and shared content is offered via this page.
4) The Page for You: The user is presented with a tailored list of notes that may be of interest on this page. Three filter buttons are included to enable relevance-based note sorting: (1) Low $(0.0 - 0.2)$, (2) Medium $(0.2 - 0.7)$, (3) High (the rest).
5) In addition, it has buttons to follow, save, like, and open a quiz-filled modal. The user views their quiz result after finishing it. YouTube API is used to provide them with a list of related YouTube videos and user suggestions for those who performed well in that particular topic if they obtain a score of less than three. Present Page for User Profile: The current user's profile details, such as their list of followers and followers and notes they have made, are contained on this page.
6) Other Users' Profile Page: This page shows other users' profiles and lets the present user see and communicate with their notes.
7) Search Functionality: Within the application, users can locate and connect with other people by using the search button.

By using Vue.js, the fluid and dynamic user experience is guaranteed, which makes it simple for users to interact with the information, take notes, and efficiently manage their profiles and notes. Furthermore, viewers who earn a score below three after finishing the quiz on the For You page will be sent recommendations from the YouTube API that include pertinent films related to the topic.

IV. Recommendation System

Recommendation systems are programs that employ various methods to examine user behavior and preferences to offer users tailored recommendations. One popular strategy is using similarity measures, which evaluate the similarity between objects or users based on their characteristics or interactions. Recommendation systems use these techniques to improve user experience by displaying pertinent content, which raises user happiness and engagement levels. In this work, 3 different similarities (Cosine, Dot-Product, and Euclidean) are implemented to compare them in recommending users as friends.

A. Cosine Similarity

The most widely used measure of similarity between two vectors in the inner product family [13] is cosine similarity. It can be used to represent the degree of similarity between two texts inside the text categorization domain. It accepts numbers between 0 and 1, where 0 denotes no resemblance and 1 denotes exact similarity between the texts. As a result, the degree of similarity between two texts, let's say *text1* and *text2*, may be stated as follows:

$$Sim(text_1, text_2) = \frac{text_1 \cdot text_2}{\| text_1 \|\| text_2 \|} = \frac{\sum_{i=1}^{n} A_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (1)$$

where A_i and B_i represent the components of vectors *text1* and *text2*, respectively. [14]

B. Dot Product

One way to measure how similar two vectors are to each other in a vector space is to utilize the dot product similarity. It determines the degree of direction alignment between two vectors by computing the cosine of their angle. Summing the products of the corresponding elements of two vectors **A** and **B** of the same dimensionality n yields the dot product $\mathbf{A} \cdot \mathbf{B}$ mathematically:

$$\mathbf{A} \cdot \mathbf{B} = \sum_{i=1}^{n} A_i \cdot B_i \quad (2)$$

This yields a single scalar number that represents the similarity: if the result is zero, it implies orthogonality (perpendicularity), but a lower dot product indicates less similarity. When comparing feature vectors or document representations, the dot product similarity is frequently employed in computer vision, natural language processing, and machine learning, among other domains.

C. Euclidean Distance

The natural length of the vector, or the distance from the point to the origin, is known as the Euclidean distance (Anton, 1993). It is the actual distance between two points in m-dimensional space. This is how its formula can be explained:

$$d^E(X, Y) = \sqrt{\frac{\sum_{i=1}^{a} (x_i - y_i)^2}{n}}, n = m \quad (3)$$

where $X = (x_1, x_2, \cdots, x_n)$ and $Y = (y_1, y_2, \cdots, y_m)$

Time series have been used Euclidean distance extensively as it offers the benefits of minimal complexity and quick calculation. Euclidean distance was used by Qiang and Vasileios (2007) to enhance a dimensionality reduction method for time series analysis, which greatly increased the effectiveness and precision of similarity searches. Chen et al. (2015) developed a method for time series similarity matching using Euclidean distance, and this approach significantly outperformed the others in terms of accuracy and efficiency. However, the biggest drawback of Euclidean distance is that it is very sensitive to singularities [15].

V. Experimental Tests

The performance of three similarity measurement models—cosine similarity, dot product, and Euclidean distance—was assessed experimentally in this section. The model that best recognizes similar annotations amongst users based on their shared interests was what wanted to find. This is done by measuring each model's recall and precision. The ratio of true positives, or comparable annotations that were successfully detected, to the total of true positives and false negatives, or similar annotations that were not identified, is known as recall. The ratio of true positives to the total of true positives and false positives (inaccurately discovered similar annotations) is known as precision.

30 users' annotations from a dataset were used to test the models. Every user posted their unique notes on a range of interesting topics. The three similarity models were used to vectorize and compare the annotations. As can be seen from the results in Table I and figure 7, cosine similarity performs better than the other models in terms of recall and precision values, proving to be useful in locating pertinent, comparable annotations in text-based contexts.

TABLE I: Comparison of Recall and Precision for Different Similarity Models.

Model	TP	FP	FN	Precision	Recall
Cosine Similarity	35	5	10	0.875	0.778
Dot Product	30	10	15	0.750	0.667
Euclidean Distance	25	15	20	0.625	0.556

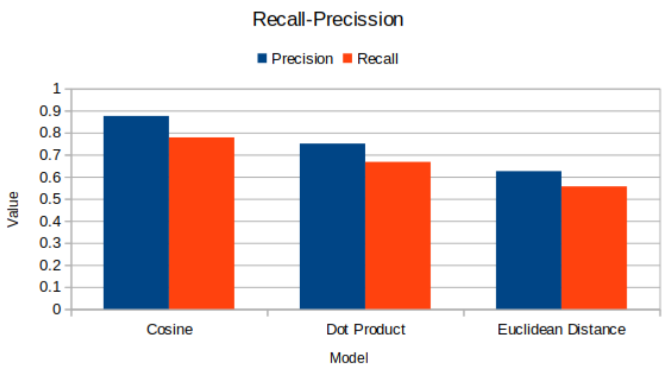

Fig. 7: Recall and Precision values reflected in Table I

.

Out of the three models investigated, cosine similarity had the best precision and recall values, according to the findings of our experimental tests. About accuracy, cosine similarity achieved a precision of 0.875 and a recall of 0.778 [16]. This means that it was able to identify a greater percentage of similar annotations with fewer false positives and false negatives. Because cosine similarity can efficiently normalize

[16] An ontological-based friends suggestion is conducted in published research authored by one of the authors.

the annotation vectors and concentrate on their direction rather than their magnitude, it performs better than other methods. The dot product and Euclidean distance models, on the other hand, showed reduced precision and recall, which was indicative of their inability to handle fluctuations in the length and magnitude of annotations. As a result, cosine similarity is the recommended model for our service, which links individuals based on common interests through text annotations, because of its strong performance in finding pertinent similarities.

To quantify the amount of collaboration between users before and after using the friends' recommendation system, we asked the participants to fill out a questionnaire that contains the following questions:

1) How often do you interact with other annotators during the annotation process? (Before:30%, After 70%).
2) How effective do you find the communication between yourself and other annotators? (Before 34%, After 87%).
3) Do you use any collaborative tools or platforms to discuss and refine annotations? (Before-Yes 80%, After-Yes 25%).
4) Do you feel that collaboration has improved the overall quality of the annotations? (Yes 93%, No 15%).
5) How important do you feel collaboration is to the success of the annotation process? (Yes 89%, No 12%).

The work published in [16] is related to suggesting friends for annotators depending on some ontological measures that relate annotators to each other by calculating the amount of relevancy between their annotations. A group created by some annotator is linked with a related ontology and the friends-suggesting process is executed when the concepts of a given ontology are compared with all annotations' tags for all users of the system. When comparing the results of the work with the current results, we found there is a match in that both works had an improvement in friends' suggestions which means comparing text with ontologies and comparing the amount of relevance between annotations using the three measures in this work both led to the same improvement in the friend's suggestion process.

VI. Conclusion and Future Works

This paper presents a thorough system for using a browser extension to annotate web pages and encourage user engagement. The system allows users to highlight text, add comments, and save these annotations to a database, ensuring the information is retrievable and manageable. The similarity between the current web page content and the previously created annotations is computed using an NLP algorithm. This similarity is used to link the existing annotations in the database to the appropriate web page content.

Allowing users to easily post their annotations on social networking platforms is one of our future work plans to improve the system. This can increase user engagement attract more users to the platform, and Implement more advanced search and filtering options using NLP and machine learning algorithms to help users find specific annotations or users with similar interests more easily.

References

[1] R. Mao, K. He, X. Zhang, G. Chen, J. Ni, Z. Yang, and E. Cambria, "A survey on semantic processing techniques," *Information Fusion*, vol. 101, p. 101988, 2024.

[2] P. Tp, U. Sulthana, and A. Parkavi, "Enhancing educational content interaction: A case study on the integration of annotation and video recommendation systems," in *2024 IEEE 9th International Conference for Convergence in Technology (I2CT)*. IEEE, 2024, pp. 1–5.

[3] H. Zhang, Y. He, X. Wu, P. Huang, W. Qin, F. Wang, J. Ye, X. Huang, Y. Liao, H. Chen *et al.*, "Pathnarratives: Data annotation for pathological human-ai collaborative diagnosis," *Frontiers in Medicine*, vol. 9, p. 1070072, 2023.

[4] R. Rennie Panter, "Digital annotations: an exploration of experiences," 2022.

[5] E. Asmar, S. Salahat, F. Zubdeh, and A. Hawash, "Enhancing users collaboration by vocal annotations," in *2021 18th International Multi-Conference on Systems, Signals & Devices (SSD)*, 2021, pp. 845–851.

[6] Z. Taha, R. Ghanem, and A. Hawash, "Enhancing deaf and dumb collaboration by video annotations," *Avrupa Bilim ve Teknoloji Dergisi*, no. 49, p. 83–88, 2023.

[7] H. Ko, S. Lee, Y. Park, and A. Choi, "A survey of recommendation systems: Recommendation models, techniques, and application fields," *Electronics*, vol. 11, no. 1, 2022. [Online]. Available: https://www.mdpi.com/2079-9292/11/1/141

[8] Z. Fayyaz, M. Ebrahimian, D. Nawara, A. Ibrahim, and R. Kashef, "Recommendation systems: Algorithms, challenges, metrics, and business opportunities," *Applied Sciences*, vol. 10, no. 21, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/21/7748

[9] N. Gu, R. Yu, K. London, Z. Pablo, and M. Roberts, "Evaluating the use of digital technologies to support design collaboration," in *Multimodality in Architecture: Collaboration, Technology and Education*. Springer, 2024, pp. 39–64.

[10] P. G. George, "The effectiveness of cooperative learning strategies in multicultural university classrooms," 2024.

[11] V. Tran, L. Nguyen, N. De, S. Chau, and N. Doan, "Does cooperative learning may enhance the use of students' learning strategies?" *International Journal of Higher Education*, vol. 8, p. 79, 07 2019.

[12] D. Avola, P. Bottoni, and A. Hawash, "Supporting group collaboration in an annotation system," *Journal of Visual Languages & Computing*, vol. 41, pp. 22–40, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1045926X17300691

[13] J. S. H. Kwangil Park and W. Kim, "A methodology combining cosine similarity with classifier for text classification," *Applied Artificial Intelligence*, vol. 34, no. 5, pp. 396–411, 2020. [Online]. Available: https://doi.org/10.1080/08839514.2020.1723868

[14] K. Park, J. S. Hong, and W. Kim, "A methodology combining cosine similarity with classifier for text classification," *Applied Artificial Intelligence*, vol. 34, no. 5, pp. 396–411, 2020.

[15] F. Zhao, Y. Gao, X. Li, Z. An, S. Ge, and C. Zhang, "A similarity measurement for time series and its application to the stock market," *Expert Systems with Applications*, vol. 182, p. 115217, 2021.

[16] D. Avola, P. Bottoni, and A. Hawash, "Relevance measures for the creation of groups in an annotation system," *J. Vis. Lang. Comput.*, vol. 25, no. 6, p. 695–702, Dec. 2014. [Online]. Available: https://doi.org/10.1016/j.jvlc.2014.09.010