




Improving Face Attendance Checking System with Ensemble Learning

Duc Tai Phan 
AiTA Lab
FPT University
Ho Chi Minh City, Vietnam
phantaiduc2005@gmail.com

Phuong-Nam Tran 
AiTA Lab
FPT University
Ho Chi Minh City, Vietnam
namtpse150004@fpt.edu.vn

Duc Ngoc Minh Dang* 
AiTA Lab
FPT University
Ho Chi Minh City, Vietnam
ducnm2@fe.edu.vn

Abstract—In many industrialized countries, implementing an efficient face attendance checking system is crucial for effective staff management. The face attendance system nowadays encounters such problems. The biggest challenge to be concerned with is the various light and angle conditions, which can damage the performance of these systems. In this paper, we present an innovative approach that combines the results of GoogLeNet, VGGFace, and FaceNet to achieve higher accuracy through Ensemble Learning. With this solution, newer systems can achieve better results. Our system encompasses image capture, face detection, database management, and feature extraction for face recognition. All related code is available at <https://anonymous.4open.science/r/Face-Attendance-Checking-2640/>

Index Terms—Face recognition, attendance checking, ensemble learning, face detection

I. INTRODUCTION

In today's high-speed industrial workplaces, accurate and effective attendance tracking is critical for good worker management. Traditional attendance methods, which usually comprise human entry or rudimentary automated systems, can be error-prone and resource-intensive. These manual tracking systems can result in inaccuracies, inefficiencies, and even fraud, eventually damaging an organization's production. Moreover, present automated systems that don't leverage advanced machine learning algorithms frequently lack the accuracy and stability needed in dynamic environments.

Recent developments in machine learning, notably in face recognition technology, offer a feasible answer to these challenges. Face recognition systems may automate and streamline attendance tracking, drastically boosting accuracy and efficiency. However, a single face recognition model may not always deliver ideal performance due to its constraints. Factors such as lighting fluctuations, angles, and facial expressions can alter the accuracy of these models.

To address these constraints, this research presents a unique face attendance strategy that leverages ensemble learning [1]. Ensemble learning incorporates different models to increase overall system performance. By merging the outputs of several models, we attempt to leverage their particular strengths and minimize their flaws, so obtaining increased accuracy in face recognition.

Our methodology is predicated on a two-phase process: encoding and comparing. In the encoding step, we develop a comprehensive database of students' photographs, capturing diverse perspectives and lighting situations to boost recognition accuracy. Detected faces in these photographs are processed by our defined models to provide embedding characteristics, which serve as ground truths.

In the comparison phase, the system processes new images taken throughout attendance sessions. It extracts facial traits from these photos and compares them with the stored features. This comparison uses a weighted voting approach, where each model's contribution is adjusted by a certain gamma coefficient γ . This advanced strategy ensures that the final prediction is a robust aggregate of the numerous model projections, boosting the system's accuracy.

Our experimental results, utilizing some models' faces in the Labeled Faces in the Wild (LFW) dataset [2], illustrate the efficiency of our ensemble learning technique, obtaining high accuracy. This emphasizes the promise of ensemble learning in creating reliable and efficient face attendance systems.

The next sections will investigate prior work in ensemble learning, present a full explanation of our proposed method, explain the experimental results and their implications, and finish with insights into the system's performance and potential areas for future research.

II. RELATED WORK

A. Face recognition

Face recognition is a biometric technique that uses digital images or video frames to analyze and compare face traits to identify or verify a person. The development of face recognition technology has significantly progressed through the expansion of machine learning and computer vision methodologies. Early face recognition algorithms, such as Principal Component Analysis (PCA) [3] and Linear Discriminant Analysis (LDA) [4], were innovative but had limitations in dealing with differences in lighting, position, and face emotions.

The use of deep learning has significantly enhanced the accuracy and resilience of face recognition systems. Convolutional Neural Networks (CNNs) [5] are widely used in modern face recognition systems, such as DeepFace [6], FaceNet [7],

* Corresponding author: Duc Ngoc Minh Dang (ducnm2@fe.edu.vn)

and VGGFace [8], which have shown impressive results on extensive facial recognition tests. Google's FaceNet, a prominent model in this field, uses a deep neural network to map images into a compact Euclidean space, enabling high-accuracy face recognition and clustering and display of faces.

In actual applications, face recognition systems must deal with obstacles such as occlusion, varying lighting conditions, and distinct facial emotions. To address these challenges, approaches such as data augmentation [9], transfer learning [10], and domain adaptation [10] are routinely applied. Ensemble learning [1] can significantly boost the overall accuracy and resilience of face recognition systems by merging numerous models with their strengths and shortcomings. This project aims to examine such an ensemble method, combining the characteristics of several face recognition models to produce a more reliable and efficient face attendance system.

B. Ensemble Learning

Ensemble learning is a sophisticated machine learning technique where numerous models are merged to enhance the accuracy and resilience of the system's performance [1]. This strategy is built on the premise that aggregating the predictions from numerous models generally leads to improved results compared to depending on a single model. The logic behind this is that individual models may thrive in certain areas while underperforming in others, and by combining different models, their strengths can be utilized while compensating for their deficiencies. This collaborative method often provides a more balanced and successful prediction.

Ensemble learning comprises a variety of strategies, each with its particular processes for merging model outputs [1]. One typical approach is Voting, which will be the focus of our discussion. Voting can be applied in numerous circumstances, such as classification and regression problems. A classification scenario frequently involves a majority vote where the class obtaining the most votes from the individual models is selected as the final prediction. For regression tasks, it can require averaging the predictions of all models to arrive at an outcome. However, we utilize a more nuanced way of combining forecasts in this particular procedure. Instead of a simple majority vote or average, we apply a specific weight, indicated as γ , to each model's prediction. These weighted forecasts are then averaged to obtain the outcome. This weighted approach allows for additional flexibility and can potentially boost performance by emphasizing the contributions of models that are more trustworthy or accurate in specific parts of the prediction task.

III. PROPOSED METHOD

A. Models

1) *Face Detection*: In our ensemble learning approach for robust face recognition, accurate face detection is the first crucial step. We employ 2 methods for this task, the Multi-task Cascaded Convolutional Networks (MTCNN) [11], and dlib toolkit [12].

MTCNN is a framework known for its robustness and efficiency in face detection and alignment. MTCNN integrates three stages of convolutional networks to detect faces and localize key facial landmarks, enabling the extraction of high-quality face images for subsequent recognition tasks.

MTCNN consists of three stages: Proposal Network (P-Net), Refine Network (R-Net), and Output Network (O-Net). Each stage progressively refines candidate face regions and improves detection accuracy. The Proposal Network (P-Net) processes an image pyramid to detect faces at different scales, generating candidate regions with binary classifications, bounding box regressions, and facial landmark localization. The Refine Network (R-Net) takes these candidate regions, refines them through convolution and fully connected layers, and outputs improved classifications, bounding box adjustments, and more precise landmark localization. The Output Network (O-Net) further refines the candidates, providing final classifications, accurate bounding box coordinates, and detailed facial landmarks. [11]

Dlib toolkit employs Histogram of Oriented Gradients (HOG) features combined with a linear Support Vector Machine (SVM) for face detection. This method begins by computing HOG features from the input image, which describes the distribution of intensity gradients and edge directions. These features are then used within a sliding window approach across the image, where each window is evaluated by a linear SVM classifier trained to distinguish between face and non-face patterns based on the extracted HOG descriptors. The sliding window technique involves scanning the image at multiple scales and positions to identify potential face regions. After classification, non-maximum suppression is applied to merge overlapping detections and refine the bounding boxes. This process ensures accurate localization of faces under varying conditions of scale, orientation, and occlusion. [12]

Both MTCNN and the Dlib toolkit are integral to our ensemble approach. We run experiments in Section. IV to identify the method that performs better in this scenario.

2) *Face Recognition*: In our ensemble learning approach for robust face recognition, precise face detection builds the foundation for accurate recognition. For the best results, we utilize these face recognition models: GoogLeNet [13], Facenet [7], and VGGFace [8], each adding distinct strengths to our system.

GoogLeNet, or Inception ResNet V1 blends the Inception architecture with residual connections, leveraging the benefits of both. Trained on the VGGFace2 dataset, it processes facial pictures using several convolutional layers and inception blocks, reinforced by residual connections that boost gradient flow and facilitate the training of deeper networks. This model creates a 512-dimensional embedding vector that accurately incorporates facial features, making it highly successful for face recognition applications.

Google's FaceNet architecture, which frequently uses deep convolutional networks like Inception-ResNet, serves as the foundation for the Facenet model. The system produces a 512-dimensional embedding vector that is optimized via triplet

loss. This loss function minimizes the discrepancy between embeddings of the same individual while maximizing the discrepancy between embeddings of different individuals. The architecture is widely recognized for its exceptional precision in face recognition, offering a full representation of facial features through its embeddings.

VGG-Face is a deep convolutional network generally based on the VGG16 architecture, including 16 weight layers. It processes facial images through a sequence of convolutional and fully connected layers, yielding a feature vector that effectively describes the face. This model is well-known for its durability and excellent accuracy in facial recognition, thanks to its thorough training on a large-scale dataset specifically developed for face recognition.

Each of these models processes identified and aligned face photos to produce high-dimensional embedding vectors that capture unique facial traits. These embeddings are subsequently employed for matching and verification activities, ensuring great accuracy and reliability in our face recognition system. The combination of multiple designs and their unique strengths boosts the robustness and overall performance of our ensemble learning approach.

B. Workflow

In this section, we will examine the workflow of this project. The workflow, as shown in Fig. 1, is divided into two parts: the Database phase, which we will discuss further in Section III-B1, where the goal is to prepare all the features we consider as ground truth, extracted from the database we collected; and the Inference phase, which we will delve into in Section III-B2, where the goal is to compare new features taken from the real world with the ground truth collected during the Database phase. The database phase in Fig. 1 takes 3 images per student and extracts the features, then saves them in the database. When doing inference, we will detect the faces in the target picture and then extract new features from these faces. Then apply the similarity function to calculate the score between them, then decide which person this face belongs to. This function can be cosine similarity [14], or the L2 distance [15], the more appropriate function will be chosen when we dive deeper into each part in the next part.

1) *Database Phase*: As we indicated before, this step is the initial one, which we—the developers who host the whole system—must perform at the beginning of the process. We have three images per pupil, which are frontal, left, and right side views of the face. This idea, if embraced, can improve the results by providing useful information on the target face. Although, while creating this project, we still obtained outstanding accuracy despite the lack of this guideline. However, since our face recognition technology will be employed in vast communities such as universities, we highly suggest adopting this notion to attain ideal outcomes.

After collecting images, we need to collect the appropriate information as well. Each student, of course, might enroll in different classes, and that information is vital for our system. Imagine an entire university comprising thousands of

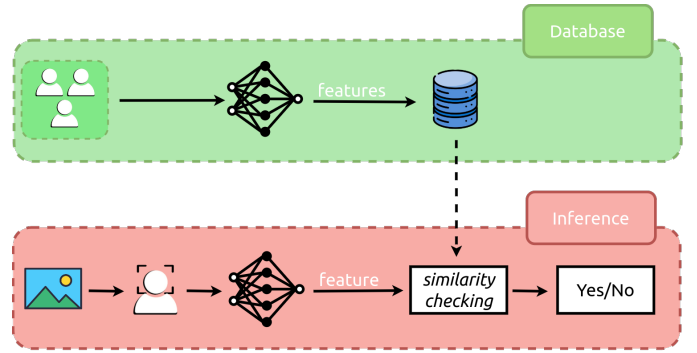


Fig. 1: Workflow of the Face Attendance Checking System using Ensemble Learning.

students, and our project employs ensemble learning, meaning numerous models will join the process, which may cause the total storage required to climb dramatically. This is not a good scenario while completing the inference task III-B2. We will need to retrieve all students' features, load them, and compare them with new features one by one, which substantially degrades system performance. The way to cope with this circumstance is the use of class codes. One student may have many class codes, however one class has a maximum of 40 pupils (in our case). This indicates that during the inference work, with the condition on the class code, we retrieved just 40 students from that class to compare with the new features. This not only saves a large amount of time and hardware calculation but also boosts accuracy since there are just 40 pairs of comparisons instead of over 1000 pairs.

Collecting images and class codes is significantly easier compared to the feature extraction operation. We need to feed all photos through our models to obtain the features. These features serve as the ground truth, providing a basis for comparison with new features captured during inference.

Each model will create output features of varied lengths. We can't quickly reduce their dimensions since it can damage critical information. Thus, during inference, if we extract a new feature from model A, we cannot use the ground truth extracted from model B for comparison because of the mismatch in dimensions. Therefore, all features gathered by each model will be put into independent data files. This technique tackles the dispute in feature sizes.

2) *Inference phase*: The inference phase is the main phase of this system, where all the algorithms are applied as effectively as feasible. When we deploy this system in the real world, the instructor will hold the camera and take a shot of all the pupils in the class every day to check attendance. These photographs contain the faces of all the students. Each face will then be detected and delivered to the models progressively. These faces come with a class code indicating where the picture was taken.

Each face is processed through the models one by one until we get all these new features, which serve as the forecasts. Remember that we have the class code for these predictions.

We will use this code to filter out all students belonging to this class, for example, SE1900 in our case. After the filtering stage, we have all the features that serve as the ground truths that the predictions will be compared to.

The predictions will be denoted as P_j , where j is the order of the students belonging to that class, referred to as students for short, and P simply stands for Person. These predictions will have the comprehensive form indicated in Eq. 1.

$$P_j = \left\{ \left(\vec{p}_{ij}^{(h)} \right)_{h=1}^{H(j)} \mid i = 1, 2, \dots, n \right\} \quad (1)$$

$H(j)$ is the total number of photos of student j . As noted in Section III-B1, each student should have 3 images, although, for various reasons, developers may choose to have a different amount of images per student. So, $H(j)$ is simply the number of photographs belonging to student j , and h is just the order of these images. i is the order of the models, where n is the total number of models.

Next are the faces. These faces will have an overall form for uniformity. In Eq. 2, k is the order of each face, where K is the total number of faces present in the primary picture.

$$F_k = \left\{ \vec{f}_{ki} \mid i = 1, 2, \dots, n \right\} \quad (2)$$

\vec{f}_{ki} is just the feature of person k extracted by model i .

Now, we will go deeper into the primary algorithm. Here we calculate something called a "score," which is obtained from the feature face compared with the ground truths for each person. In Eq. 3, we calculate the cosine similarity of the average similarity of \vec{f}_{ki} and $\vec{p}_{ij}^{(h)}$, then multiply it by the γ of model i to get the final score. This γ functions as a weight that contributes to the overall calculation of that model; a lower γ suggests this model will contribute less than other models.

$$s_{jk} = \sum_{i=1}^n \gamma_i \times \frac{1}{H(j)} \sum_{h=1}^{H(j)} \cos(\vec{p}_{ij}^{(h)}, \vec{f}_{ki}) \quad (3)$$

For better understanding, s_{jk} is just the score of the pair face k and the person j in the list. If this formula is hard to follow, a shorter version is presented in Eq. 4, where we assume that each student has only one image in the database.

$$s_{jk} = \sum_{i=1}^n \gamma_i \times \cos(\vec{p}_{ij}, \vec{f}_{ki}) \quad (4)$$

After repeating this process with all faces, we have a list of scores matching each face. We then use the max method to identify which individual best matches each face. This step is produced in Eq. 5.

$$j^* = \arg \max_j (s_{jk}) \quad (5)$$

P_{j^*} is the individual who best matches face k . After these four phases, we can discover the individual that best suits each face.

IV. EXPERIMENTAL RESULT AND DISCUSSION

In this section, we present the experimental results obtained from our proposed face attendance checking system using ensemble learning. The experiments were conducted using a group of 40 people in the Labeled Faces in the Wild (LFW) [11] dataset to validate the effectiveness of our approach. We just take only 40 celebrities and each of them has more than 4 pictures. 3 of them will be used to calculate the ground truth vector, and we only take 1 left to test the system. The reason why we chose the number 40 is because, in our countries, a classroom usually has 40 students or lower. With this approach we believe it will best illustrate the conditions face attendance system has to face in Vietnam.

We employed three different face detection methods: dlib [12] and MTCNN [11] and YOLOFacev5 [?]. Dlib is a toolkit that comprises algorithms, that can be used for various tasks. Here we use Dlib and MTCNN for face-detecting tasks, to see which one is better in scenarios.

The experiments were also designed to compare the accuracy of our system under different conditions: calculate the average result before and after calculating the score. Since each person has, for example, 3 images, then there are 3 feature vectors. But since there's only one vector comes from the inference phase, we will have 2 ways to compare these vectors. First, calculate the average ground truth vector, then directly calculate the score with the target vector. The other way is to calculate the score of the target with each of the ground truth vectors sequentially, like the Eq. 3. Additionally, we tested the system with different methods of calculating the score, it's the choice between the cosine similarity and L2 distance. The results of our experiments are summarized in Tables I and II.

TABLE I: Experiments with Averaging Vector approach

Model	Score Method	Face Detector	Accuracy	Timing
Resnet	Distance	Dlib	100%	0.1225
		MTCNN	97.5%	0.1318
		YoloFacev5	100%	0.1021
	Cosine	Dlib	100%	0.1324
		MTCNN	97.5%	0.1223
		YoloFacev5	100%	0.1015
VGGFace	Distance	Dlib	85.0%	0.2147
		MTCNN	90.0%	0.1544
		YoloFacev5	90.0%	0.0935
	Cosine	Dlib	85%	0.2236
		MTCNN	90%	0.1276
		YoloFacev5	90%	0.0899
Facenet512	Cosine	Dlib	90%	0.4465
		MTCNN	95%	0.3178
		YoloFacev5	90%	0.2762
	Distance	Dlib	90%	0.4164
		MTCNN	95%	0.3176
		YoloFacev5	90%	0.2772
Our 3 models	Distance	Dlib	100%	0.7645
		MTCNN	97.5%	0.6195
		YoloFacev5	100%	0.4725
	Cosine	Dlib	100%	0.7473
		MTCNN	97.5%	0.6223
		YoloFacev5	100%	0.4926

As we can see, comprehensively, the processing time when using the averaging vector method tends to be shorter than

TABLE II: Experiments with Averaging Score approach

Model	Score Method	Face Detector	Accuracy	Timing
Resnet	Distance	Dlib	97.5%	0.1139
		MTCNN	97.5%	0.1423
		YoloFacev5	97.5%	0.1215
	Cosine	Dlib	97.5%	0.1059
		MTCNN	97.5%	0.1356
		YoloFacev5	97.5%	0.1145
VGGFace	Distance	Dlib	85.0%	0.1154
		MTCNN	90.0%	0.1345
		YoloFacev5	90.0%	0.1151
	Cosine	Dlib	85%	0.1135
		MTCNN	90.0%	0.1348
		YoloFacev5	90.0%	0.1157
Facenet512	Cosine	Dlib	92.5%	0.3087
		MTCNN	92.5%	0.3198
		YoloFacev5	95.0%	0.2874
	Distance	Dlib	92.5%	0.3233
		MTCNN	95.0%	0.3031
		YoloFacev5	92.5%	0.2817
Our 3 models	Distance	Dlib	100%	0.5676
		MTCNN	97.5%	0.5575
		YoloFacev5	100%	0.5304
	Cosine	Dlib	100%	0.5727
		MTCNN	97.5%	0.5392
		YoloFacev5	97.5%	0.5092

when using averaging score. This can be explained by the common logic, averaging vector and then calculating the score once, on the other side, the averaging score needs to loop through all possible features of a pre-defined person in the database, which causes more time to finish. Also, calculating with the l2 distance formula is faster than calculating with cosine similarity since cosine similarity includes all calculations l2 distance has, except for the subtraction. Although the difference is not that noticeable.

In general, using MTCNN can make the process faster due to its optimized architecture, whereas the dlib toolkit is full of algorithms. One more reason for this is that MTCNN can utilize the GPUs while dlib - which can still run parallel but there is only a CPU comes to work.

The most important thing that the results tell us is the accuracy when using the combined version of all 3 models. As we can see, when other models do their jobs, it's hard to get 100%, only ResNet can do that. Implementing 3 models to perform the calculation has raised the accuracy to 100% four times with the dlib toolkit, with both distance or cosine methods.

We are also concerned about a *threshold* for scores. It is a level that every pair of features needs to be higher, or else it will be considered as a "student who does not belong to that class". "Higher" here means we are using cosine similarity since its range of value is between 0 and 1. But we also experiment on distance metrics, meaning the score here should be "lower" than the distance threshold. So here we will have 2 thresholds for 2 metrics. The results are presented in Fig. 2. We experiment with it on the LFW Deepfunneled version [2], with a class size is 40 students.

As we can see, with cosine similarity, the threshold of 0.8 can achieve 90% accuracy on the LFW dataset. On the other hand, with distance metric, the threshold of 4.25 can achieve

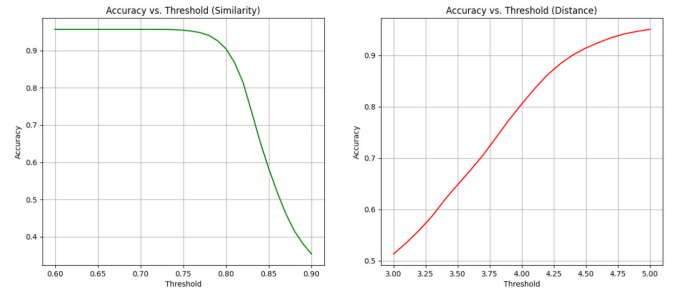


Fig. 2: Thresholds for Cosine Similarity and Distance

up to 87.5% accuracy. But we have noticed that this number 4.25 here can vary depending on different scenarios since the range of value of l2 distance is 0 to positive infinity ($0 \rightarrow +\infty$).

$$score \geq threshold$$

V. CONCLUSION

In conclusion, our ensemble learning-based face attendance checking system offers a robust solution for accurate and efficient attendance management. By leveraging multiple face recognition models and a weighted voting approach, we achieved up to 100% accuracy in Section IV. This system demonstrates significant potential for practical deployment in real-world environments, such as educational institutions and workplaces, where reliable attendance tracking is essential. Future efforts will focus on scalability and further enhancing system performance through advanced model integration.

REFERENCES

- [1] F. Huang, G. Xie, and R. Xiao, "Research on ensemble learning," in *2009 International Conference on Artificial Intelligence and Computational Intelligence*, vol. 3, 2009, pp. 249–252.
- [2] G. B. Huang, M. Mattar, H. Lee, and E. Learned-Miller, "Learning to align from scratch," in *NIPS*, 2012.
- [3] K. Pearson, "Liii. on lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, no. 11, pp. 559–572, 1901.
- [4] R. A. Fisher, "The use of multiple measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.
- [5] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015.
- [6] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
- [7] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
- [8] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," *British Machine Vision Conference (BMVC)*, 2015.
- [9] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," in *Journal of Big Data*, vol. 6, no. 1. Springer, 2019, pp. 1–48.
- [10] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, 2010.
- [11] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," *IEEE Signal Processing Letters*, vol. 23, no. 10, p. 1499–1503, Oct. 2016.
- [12] D. E. King, "Dlib-ml: A machine learning toolkit," in *Journal of Machine Learning Research*, vol. 10, no. Jul, 2009, pp. 1755–1758.

- [13] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.
- [14] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.
- [15] Euclid, *The Elements of Euclid: Viz. the First Six Books, Together with the Eleventh and Twelfth*. Encyclopaedia Britannica, 1956.