Implementation of OCR and Face Recognition on Mobile Based Voting System Application in Indonesia

Inggrid Fortuna[1], Yaman Khaeruzzaman[2]
[1,2]Department of Informatics, Universitas Multimedia Nusantara, Tangerang, Indonesia
[1]inggrid.fortuna@student.umn.ac.id
[2]yaman.khaeruzzaman@umn.ac.id

Accepted 10 June 2022
Approved 5 July 2022

***Abstract*—Elections are a form of democratic practice in Indonesia. Every 5 years an election will be held to elect a president. People who have been able to take part in the election will come to the polling station (TPS) to channel their voting rights. However, this conventional method proved ineffective because some people who were unable to attend due to certain situations, for example; traveling out of town, did not want to queue, and experienced illness or physical disability. Therefore, this study aims to design and implement an online voting system based on Android as an alternative to conventional elections and digital transformation in the voting method in Indonesia. The system will use Optical Character Recognition technology by firebase ml-kit to read Identification Number on the Identity Card and face recognition technology to compare the faces of voters during registration and during online elections. The Face Recognition system is implemented using Multi-task Convolutional Neural Network to detect faces and using Tensorflowlite to translate the facial model provided by the FaceNet model. The Result of Research shows the success of the OCR system is 96.67% and the accuracy of face recognition is 100%. The accuracy of OCR ml-kit and face detection using Multi-task Convolutional Neural Network and Face Recognition using tensorflowlite and FaceNet models proved to be 100% successful.**

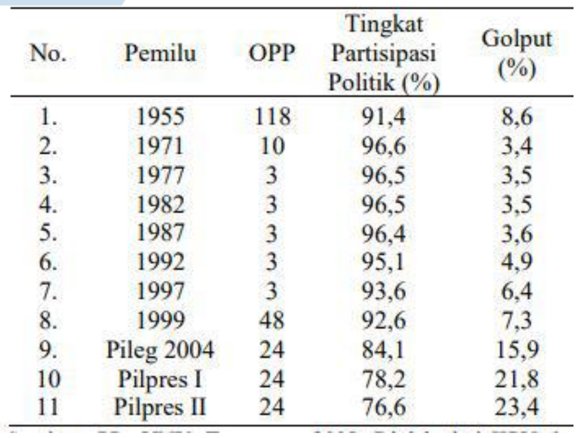
Index Terms— *Android**; face recognition; faceNet; mobile app; optical character recognition;*** **tensorflowlite**

I. Introduction

General elections in Indonesia still use conventional elections where people have to come directly to the polling station (TPS) to vote by bringing their ID card. Based on the results of the interviews, several interviewees also stated that they could not vote because they were unable to attend the TPS.

The problem faced according to the interview was that there were several conditions that did not allow the interviewees to go to the TPS, so they could not vote. The reasons based on the interviews are traveling out of town, not wanting to queue, and experiencing illness or physical disability. This causes some people not to participate in the people's party which is held every 5 years.

This causes an increase in the number of abstentions in Indonesia. The abstention rate reached 23.30% in the 2004 presidential election, 27.45% in 2009, 30.42% in 2014, and 19.24% in 2019 [1] as shown on Fig. 1.



No.	Pemilu	OPP	Tingkat Partisipasi Politik (%)	Golput (%)
1.	1955	118	91,4	8,6
2.	1971	10	96,6	3,4
3.	1977	3	96,5	3,5
4.	1982	3	96,5	3,5
5.	1987	3	96,4	3,6
6.	1992	3	95,1	4,9
7.	1997	3	93,6	6,4
8.	1999	48	92,6	7,3
9.	Pileg 2004	24	84,1	15,9
10	Pilpres I	24	78,2	21,8
11	Pilpres II	24	76,6	23,4

Fig. 1. Source : PPs UNIS Tangerang 2001, Kompas diolah

Thus, this problem needs to be solved with a solution to create a system that could facilitate people to choose from home without having to go to a polling station. This system can allow people to choose as long as they are connected to the internet. People who could not go to TPS can be visited by the General Elections Commission (KPU) who will bring this mobile application so that people can still take part in elections even though they are unable to attend on TPS. This is certainly more practical than bringing the ballot box and ballot papers to the voters' residence.

Although the conventional selection is still relevant to use, there are several disadvantages in this method, such as higher - cost, time and security inefficiencies. Conventional elections require the cost of printing ballots and other costs. In addition, there is a long process from voting to counting ballots and it takes quite a lot of time. For security, there is still a percentage of human error or fraud occurring during vote counting. One alternative way to solve this problem is through online voting.

II. Related Work

The difference between previous research and current research, can be seen in table 1.

TABLE I. LATEST RESEARCH VS RECENT RESEARCH

Latest Research		Recent Research
Oka N.H, et all, 2018	Using Student ID Card to login with Optical Character Recognition (OCR) without face recognition	The app could extract data from ID Card using OCR
Siti S. A, et all, 2020	ID Card was used as registration using OCR with Template Matching Correlation without face recognition	Using OCR and regular expression to extract ID Number and using face recognition
Noha E. E, et all, 2013	Face detection using Gabor Filter with accuracy of 93,5 % for the Voting app	E-Voting app using Multi-task Cascaded Neural Network to detect faces with accuracy of 100%
Ahmadi I. Lubis, 2018	The system could detect faces with the accuracy 88% using Kairos Face Recognition Libraries	The system could recognize faces with the accuracy of 100% with tensorflowlite and FaceNet

In analyzing and designing this mobile-based application, it is necessary to study the literature that supports the research. The theoretical basis and sources of this literature study are in accordance with the topic of discussion.

A. *E-Voting*

E-voting is a voting and counting votes in general elections using electronic devices [2]. There are some conditions that has to be fulfilled by e-voting [3]:

- People who can participate in elections are people who are legally entitled to vote.
- Selection can only be made once for each person.
- Other people's choices must not be known by other parties.
- Voters may not be duplicated.
- Other people's choices may not be changed without that person's knowledge.
- Each person can ensure that his/her voice has been entered into the vote tabulation.
- Each person can find out who has voted and who did not vote.

B. *ML kit Firebase*

According to Google Firebase, ML Kit is a Mobile SDK that brings Google's machine learning expertise to Android and iOS apps in a powerful and easy-to-use package. ML Kit provides a handy API that can help users use custom TensorFlow Lite models in mobile apps.

Fig. 2 shows several features of the ML Kit that can be used to recognize text, detect faces, identify famous buildings, scan barcodes, label images and identify text language. On device or in the cloud, the various APIs for ML Kit can run and process data quickly and work even when there is no network connection.

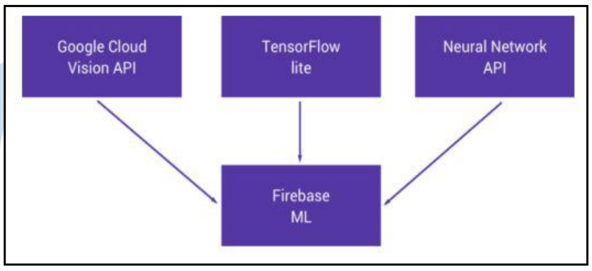


Fig. 2. Diagram MLKit [4]

C. *Face Detection and Recognition*

Face Recognition is a human face recognition system that can be used in the field of security system such as room entry access, location monitoring surveillance, and individual identity searches in police databases. In addition, face recognition is also used in identifying criminals, developing security systems, processing images and films, and human computer interaction. Face Recognition is also a Computer Vision activity to identify and verify a person based on their photo. Face recognition consists of 4 stages; detection, alignment, feature extraction and matching as shown in Fig. 3.

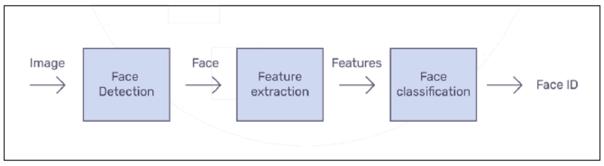


Fig. 3. Face Recognition Pipeline [5]

D. *Tensorflow*

Based on the tensorflow.org [6], Tensorflow is a computational framework for modeling machine learning that provides various toolkits that allow modeling at the level of abstraction the user wants. In addition, Tensorflow also helped create a neural network which is an artificial network similar to the human brain on a large scale [7]. Tensorflow has two components:

1. Protocol Buffer(.pb) which contains graphics and models to run the tested model.
2. The runtime that runs the graph.

E. *Multi-task Cascaded*

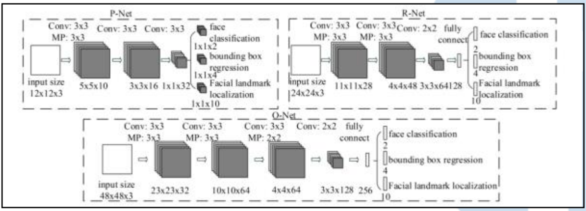


Fig. 4. MTCNN Architecture [8]

Multi-task Cascaded Convolutional Neural Network(MTCNN) is one of the Deep Convolutional Neural Network methods used to recognize or detect faces. There are 3 separate networks on MTCNN as shown in Fig. 4:

1. Proposal Network (P-Net)

   In this network, each image will be searched by the 12x12 kernel to search for faces horizontally to the bottom corner of the image.

2. Refine Network (R-Net)

   Similar to P-Net, on the R-Net network, the image will be searched for faces horizontally. However, on this network, faces will be detected using a 24x24 kernel

3. Output (O-Net)

   On this network, the image will be searched by the 48x48 kernel to detect faces.

F. *FaceNet*

FaceNet is a face recognition system developed by Google researchers with high accuracy results. In its implementation, Deep Convolutional Neural Network will extract facial features into vectors. This vector is also called Vector Embedding which is generated. The vector can map the similarity of faces. FaceNet uses the Deep CNN model in the form of ZF-et or Inception [9].

FaceNet will be implemented in the mobile device-based E-Voting application at the feature learning stage. The vectors are generated by FaceNet consists of 128 elements or Face embedding classified using the Linear Support Vector Machine (SVM) which will classify the facial identity of vector embedding.

Face Embedding is a vector representing the extracted features of the face. Next, these features will be compared with the vectors generated for other faces. Vectors that are close to zero may be the same person, while other vectors that are far away may be different people. The classification model developed will take the input face embedding and predict the identity of the face. The FaceNet model will generate an embedding for a specific face image and be used as part of the classification itself. In addition, FaceNet models pre-process faces to create embedding faces that can be stored and used as input for the classification model.

G. *Eucledean Distance*

The Euclidean Distance method is one of the techniques used to determine the degree of similarity or dissimilarity using the distance method between two vectors. The following is the Euclidean formula, which is the root of the square of the difference between 2 vectors.

$$d(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \qquad (1)$$

d = Euclidean distance
p, q = two dots in n-space Euclidean
qi, pi= Euclidean vector, starting point of vector
n = n-space

The Euclidean distance is always greater than or equal to zero. If the measurement result is zero, then the two vectors are identical. However, if the measurement result is high, the two vectors are not identical [10].

H. *Optical Character Recognition*

Optical Character Recognition (OCR) is the recognition of alphanumeric characters from handwritten characters or files or images into editable text. OCR, according to Yadav (2013) is a process to convert scanned printed or handwritten documents into ASCII characters, namely machine-readable characters.

I. *Precicion, Recall and Accuracy*

Precision is the level of accuracy between the information requested by the user and the answer given by the system. Meanwhile, recall is the success rate of the system in retrieving information. In addition, in pattern recognition, the term accuracy is also known, namely the level of closeness between the predicted value and the actual value.

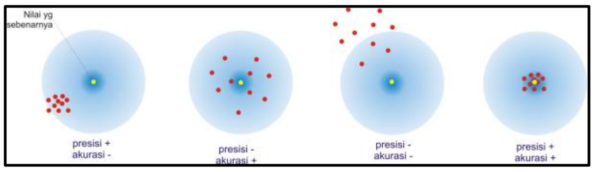


Fig. 5. Accuracy vs Precicion

III. IMPLEMENTATION

The OCR system and facial recognition will be implemented in the online voting application as seen in the design that has been made previously.

1. Optical Character Recognition, the data on the ID card will be read through by Optical Character Recognition to extract identity numbers. This data can be read because it uses regular expression or regex that fulfills the data characteristics. Regex is a formula for finding the pattern of a sentence/string. For example, the Population Identification Number (NIK) written on the National Identity Card (KTP) can be validated using regex. According to government regulations in PP-No-37-year-2007 article 37 states that NIK has a length of 16 digits with details of the first 6 digits being the regional code of the province, district/city, and sub-district. The second 6 digits are the date, month, year of birth. But for women plus 40 for the date of birth. Then the last 4 digits are sequence values only. Therefore, the regex on the ID card can be set to 16 digits. This regex will be implemented in OCR in the E-Voting application.

2. Face Recognition, face verification provides security that residents are valid in making elections. Faces will be detected using MTCNN which consists of 3 layers, namely P-Net, R-Net, and O-Net. The face detection results will then be processed by tensorflowlite which is a bitmap translator using the FaceNet model. The result will then be converted to a byte array to produce vectors or face embedded. The vectors of these two faces will be measured using the Euclidean formula to determine whether the two faces are identical or not.

A. *Flowchart*

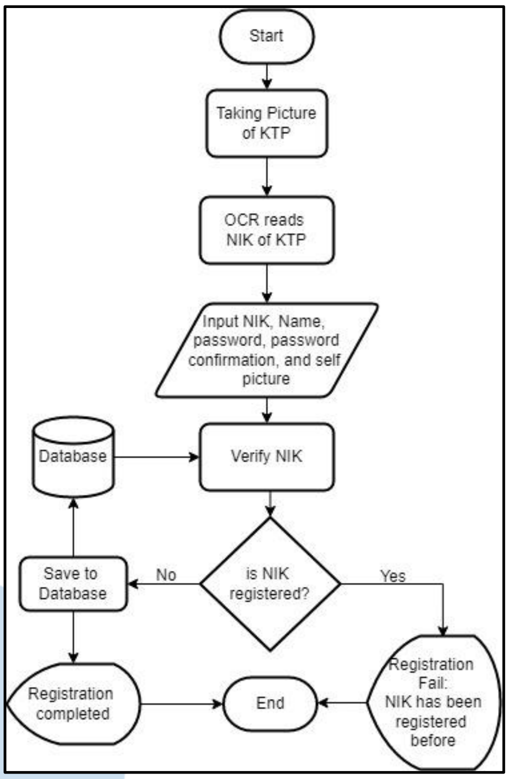


Fig. 6. Register Flowchart

In Fig. 6, you can see the registration flow that will be carried out when the user first uses the application. Users need to enter their NIK, name, password, password confirmation and a photo of themselves. The data will then be stored in Firebase. In Fig. 7, the process of voting flow as shown consists of face verification. If the face does not match the photo at the time of registration, the user cannot make a selection. Conversely, if the face has been verified according to the photo at the time of registration, then the user is directed to the selection.

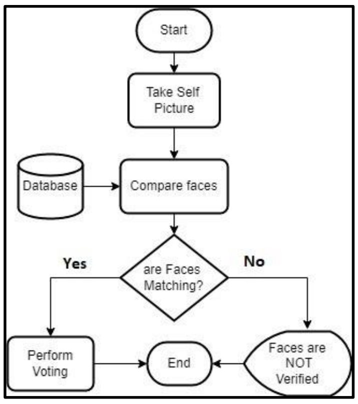


Fig. 7. Voting Flowchart

B. *Application Design and Interface*

The system is based on Android mobile application as seen on Fig. 8, Fig. 9, Fig 10, and Fig. 11. The user must register their Identity Card (KTP) to login. Optical Character Recognition will extract the Identity Number (NIK) from the ID Card picture and then pass the value to the register page, where user has to fill in several inputs. User also needs to take a selfie picture in the process of registration. After obtaining the picture, user will be taken to the face matching section before voting activity.

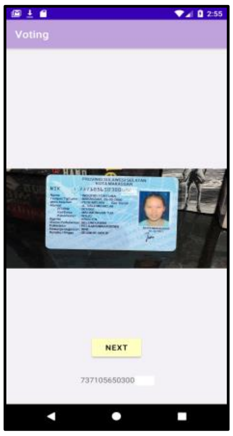


Fig. 8. KTP registration

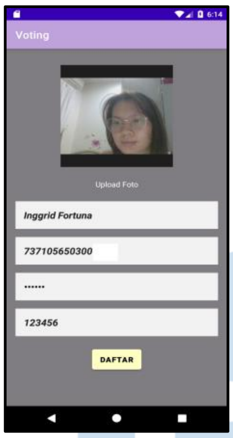


Fig. 9. Registration

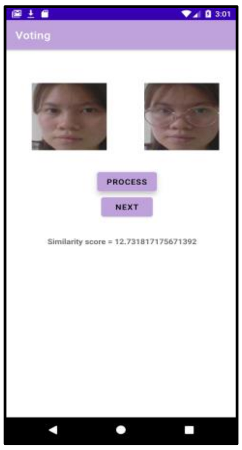


Fig. 10. Face Comparison

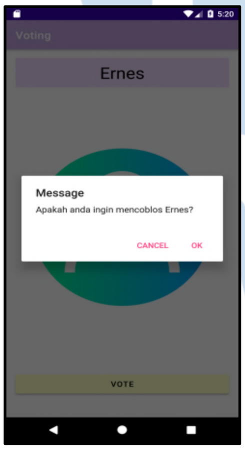
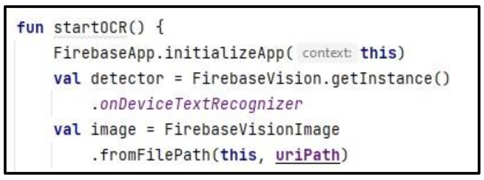


Fig. 11. Perform Voting

C. *Optical Character Recognition*

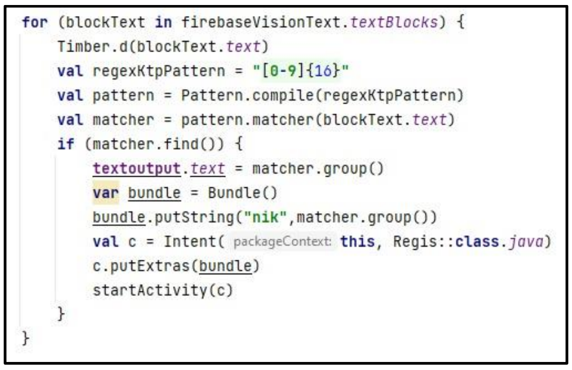
Fig. 12 shows the initialized photo path processed by the firebase, when the user has uploaded or taken a photo of his ID card.



```
fun startOCR() {
    FirebaseApp.initializeApp( context: this)
    val detector = FirebaseVision.getInstance()
        .onDeviceTextRecognizer
    val image = FirebaseVisionImage
        .fromFilePath(this, uriPath)
```

Fig. 12. Photo Path to Firebase Vision Image

Fig. 13 shows the Regular Expression of NIK on KTP in the system which functions to detect the National Identity Number (NIK) after the text has been successfully detected. The Regular Expression is represented by "[0-9]{16}" which is a pattern of 16 digits.

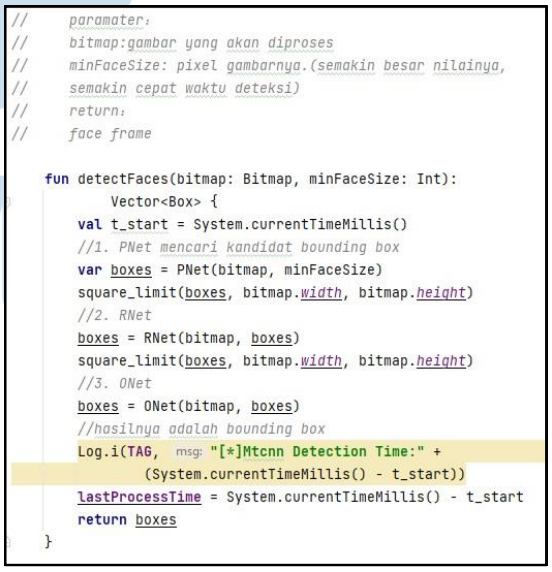


```
for (blockText in firebaseVisionText.textBlocks) {
    Timber.d(blockText.text)
    val regexKtpPattern = "[0-9]{16}"
    val pattern = Pattern.compile(regexKtpPattern)
    val matcher = pattern.matcher(blockText.text)
    if (matcher.find()) {
        textoutput.text = matcher.group()
        var bundle = Bundle()
        bundle.putString("nik",matcher.group())
        val c = Intent( packageContext: this, Regis::class.java)
        c.putExtras(bundle)
        startActivity(c)
    }
}
```

Fig. 13. Regular Expression of NIK

D. *Face Recognition*

Fig. 14 shows the face detection function with bitmap parameters consists of the image to be detected and minFace-Size which is the minimum image size of a face in pixels. Processing speed will be faster if the parameter minFaceSize is larger. The result that will be given is the position of all detected faces along with facial landmarks.

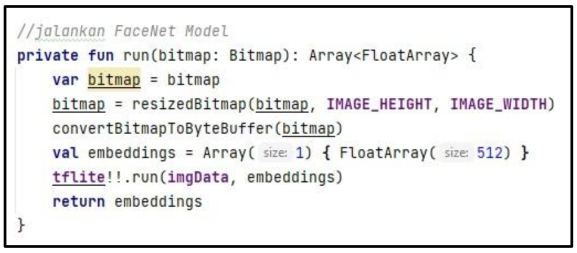


```
//    paramater:
//    bitmap:gambar yang akan diproses
//    minFaceSize: pixel gambarnya.(semakin besar nilainya,
//    semakin cepat waktu deteksi)
//    return:
//    face frame

    fun detectFaces(bitmap: Bitmap, minFaceSize: Int):
            Vector<Box> {
        val t_start = System.currentTimeMillis()
        //1. PNet mencari kandidat bounding box
        var boxes = PNet(bitmap, minFaceSize)
        square_limit(boxes, bitmap.width, bitmap.height)
        //2. RNet
        boxes = RNet(bitmap, boxes)
        square_limit(boxes, bitmap.width, bitmap.height)
        //3. ONet
        boxes = ONet(bitmap, boxes)
        //hasilnya adalah bounding box
        Log.i(TAG,  msg: "[*]Mtcnn Detection Time:" +
                (System.currentTimeMillis() - t_start))
        lastProcessTime = System.currentTimeMillis() - t_start
        return boxes
    }
```

Fig. 14. Detect Faces

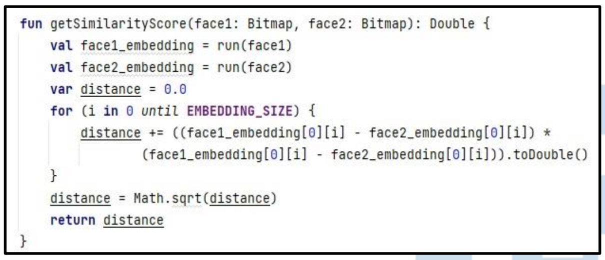
First of all, the interpreter tflite of assets folder will be initialized and loaded into android in FaceNet class. Next, to get the face vector, the bitmap needs to be resized via the resizedBitmap function with filtration. After that the bitmap will be converted to Byte buffer. Then, the Tflite Interpreter will run the FaceNet Model to generate a face vector as shown in fig. 15. Calculate the Euclidean distance between 2 faces to find the degree of similarity.



```
//jalankan FaceNet Model
private fun run(bitmap: Bitmap): Array<FloatArray> {
    var bitmap = bitmap
    bitmap = resizedBitmap(bitmap, IMAGE_HEIGHT, IMAGE_WIDTH)
    convertBitmapToByteBuffer(bitmap)
    val embeddings = Array( size: 1) { FloatArray( size: 512) }
    tflite!!.run(imgData, embeddings)
    return embeddings
}
```

Fig. 15. Run FaceNet Model

The result of the run function is that the vectors of the two faces will be compared with the distance using the Euclidean formula as mentioned in the literature. If the distance between these two vectors is close to 0 then the two faces are identical. If the distance between these two vectors is away from 0, then the two faces are different. The code is shown in Fig. 16.



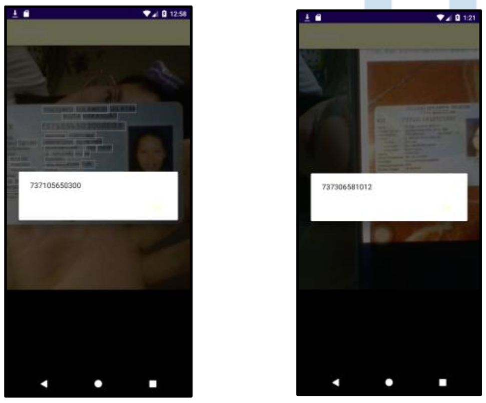
```
fun getSimilarityScore(face1: Bitmap, face2: Bitmap): Double {
    val face1_embedding = run(face1)
    val face2_embedding = run(face2)
    var distance = 0.0
    for (i in 0 until EMBEDDING_SIZE) {
        distance += ((face1_embedding[0][i] - face2_embedding[0][i]) *
                (face1_embedding[0][i] - face2_embedding[0][i])).toDouble()
    }
    distance = Math.sqrt(distance)
    return distance
}
```

Fig. 16. Get Similarity

IV. Analysis

A. *Testing Optical Character Recognition on Identity Card Data*

KTP data that has been successfully read by OCR would be given a box mark right in the NIK section. If data is not detected, an error notification will appear on the application screen as shown on Fig. 17.



(a) Detection 1 (b) Detection 2

Fig. 17. NIK Detection by OCR

Through the emulator camera, the photo of the ID card will be taken and processed by OCR. The photo was taken during the day with quite dim lighting. However, the NIK data can be read perfectly. NIK detection has several references that are used in the data collection process to determine the level of accuracy of the Optical Character Recognition(OCR) ID card. Photo distance is one of the references used.

TABLE II. The Result of NIK Detection by OCR

Respondent	Distance (cm)		
	5	10	15
1	✓	✓	✓
2	✓	✓	✓
3	✓	✓	✓
4	×	×	×
5	✓	✓	✓
6	✓	✓	✓
7	✓	✓	✓
8	✓	✓	✓
9	✓	✓	✓
10	✓	✓	✓
11	✓	✓	✓
12	✓	✓	✓
13	✓	✓	✓
14	✓	✓	✓
15	✓	✓	✓
16	✓	✓	✓
17	✓	✓	✓
18	✓	✓	✓
19	✓	✓	✓
20	✓	✓	✓
21	✓	✓	✓
22	✓	✓	✓
23	✓	✓	✓
24	✓	✓	✓
25	✓	✓	✓
26	✓	✓	✓
27	✓	✓	✓
28	✓	✓	✓
29	✓	✓	✓
30	✓	✓	✓

Note: ✓ Detected, Correct result
× Detected, Incorrect result

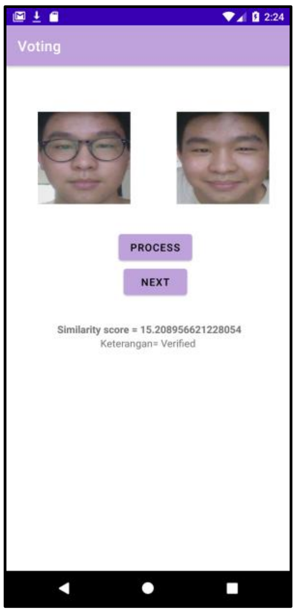
The result of reading NIK data on different distances is recorded on the table 2. Based on the test, the level of precision and accuracy of the OCR in reading NIK data reached 96.67%.

$$Precision = \frac{TP}{FP + TP} = \frac{29}{30} \times 100\% = 96.67\%$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{29}{30} \times 100\% = 96.67\%$$

B. *Testing Face Recognition*

In Fig. 18, both faces where the first picture used glasses, while in the second picture the respondent did not wear glasses. The similarity number used the Euclidean formula, which is 15.21 (rounded up by 3 digits after the comma). In the Euclidean theory mentioned in the literature, if the result is close to zero, then the two faces are identical. So it can be concluded that the results of the comparison of the two faces are identical.



Similarity score = 15.208956621228054
Keterangan= Verified

Fig. 18. An Example of the Test

In this research, the same face and different faces are tested to define the threshold number to determine either the faces are identic or not. The lowest Euclidean distance generated is 6.93 which is the same face without any attribute in the bright light. The highest Euclidean distance is 28.052 which is different persons' faces with eyeglasses in the bright light. Therefore, the application used the threshold value of Euclidean distance is 17 (rounded of 17.491)

$$Threshold = \frac{6.93 + 28.052}{2} = \frac{34.982}{2} = 17.491$$

Table 3 is the test result of face recognition by 30 respondents with different attributes used and the light condition.

TABLE III. THE RESULT OF FACE RECOGNITION

Respondent	Condition		Euclidean Distance
	Attribute	Lighting	
1	None	Dark	15.210
2	None	Bright	8.000
3	Mask	Bright	8.800
4	Eyeglasses	Dark	11.490
5	Eyeglasses	Dark	12.720
6	Eyeglasses	Dark	9.350
7	None	Bright	6.931
8	Mask and Eyeglasses	Bright	13.028
9	Mask	Bright	8.170
10	Mask	Bright	8.661
11	None	Dark	15.656
12	None	Bright	13.955
13	None	Bright	13.867
14	None	Dark	6.950
15	None	Dark	14.854
16	None	Dark	14.140
17	None	Bright	8.930
18	None	Bright	11.548
19	None	Bright	10.128
20	None	Bright	17.806
21	None	Dark	15.700
22	None	Bright	15.836
23	None	Bright	15.540
24	None	Dark	14.958
25	None	Dark	17.000
26	None	Dark	14.840
27	None	Bright	16.623
28	None	Bright	13.415
29	None	Bright	12.227
30	None	Bright	10.257

Since all of the tests done by all respondents generate Euclidean Distance are less than or equal the threshold (17), therefore the level of precision and the overall accuracy in the data in table III reached 100%.

V. CONCLUSION

Mobile-based E-voting application development could implement Optical Character Recognition and Face Recognition. The data on the KTP read by OCR during registration. This data can be read because it used regular expression or a regex that met the characteristics of the data, namely the 16-digit ID card in the form of an integer. Then, before voting, the user would verify the face as an evidence that the residents are valid in making the election. Faces would be detected using Multi-Task Convolutional Neural Network (MTCNN) which consists of 3 layers; P-Net, R-Net, and ONet. The face detection results would then be processed by tensorflowlite which is a bitmap translator using the FaceNet model. The result would then be converted to a bytearray to produce vectors or face embedded. The vectors of these two faces would be measured using the Euclidean formula to determine whether the two faces are identical or not. The result of OCR data detection accuracy is 96.67% in 30 respondents. Meanwhile, the accuracy of face Recognition using MTCNN reached an accuracy of 100% in sufficient lighting. Testing on the application also shows the success of testing the components needed for the E-voting application. So it can be concluded that all the E-Voting requirements designed in this study have been met.

The suggestions for further research are:

1. Improved accuracy of reading NIK data by optical character recognition by using a more appropriate regular expression by following the pattern of city/district code, date of birth and 4-digit serial number found on the KTP.

2. Improved facial recognition accuracy even when photos are taken in low light conditions.
3. Design items from the E-Voting application can also be further developed with news features or portfolios from prospective candidates.
4. Improved features in the application to conduct more than one election such as in simultaneous elections (President / Vice President), DPD, DPR, DPRD I, DPRD II) and tiered elections such as party elections or candidates for people's representatives.

REFERENCES

[1] B. News, "Lembaga survei: Jumlah golput di Pilpres 2019 paling rendah sejak 2004," 2019. [Online]. Available: https://www.bbc.com/indonesia/indonesia-48130161

[2] F. N. D. Edi Priyono, "E-VOTING: URGENSI TRANSPARANSI DAN AKUNTABILITAS," Seminar Nasional Informatika (SEMNASIF), vol. 1, no. 5, pp. 3–4, 5 2010.

[3] Schneier B, Applied Cryptography, Second Edition: Protocols, Algorithms, and Source Code in C. John Willey Sons,Inc, 1996.

[4] Hitherejoe, "EXPLORING FIREBASE MLKIT ON ANDROID: INTRODUCING MLKIT." [Online]. Available: https://joebirch.co/uncategorized/exploring-firebase-mlkit-onandroid-introducing-mlkit-part-one/

[5] L. Dulˇci´c, "Face Recognition with FaceNet and MTCNN." [Online]. Available: https://arsfutura.com/magazine/face-recognition-with-facenetand-mtcnn/

[6] Tensorflow, "Tensorflow." [Online]. Available: https://www.tensorflow.org/

[7] E. Uri, "Real time face recognition with Android + TensorFlow Lite." [Online]. Available: https://medium.com/@estebanuri/real-timeface-recognition-with-android-tensorflow-lite-14e9c6cc53a5

[8] Z. L. Kaipeng Zhang, Zhanpeng Zhang, "Joint Face Detection and Alignment using Multi-task Cascaded Convolutional Networks." [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1604/1604.02878.pdf

[9] J. P. Florian Schroff, Dmitry Kalenichenko, "FaceNet: A unified embedding for face recognition and clustering," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2015, vol. 1, no. 1, pp. 1–10, 2015.

[10] Tibco.com, "Eucledian Distance." [Online]. Available: https://docs.tibco.com/pub/spotfire/7.0.0/doc/html/hc/hceuclideandistance.htm