Jutisi: Jurnal Ilmiah Teknik Informatika dan Sistem Informasi
Jl. Ahmad Yani, K.M. 33,5 - Kampus STMIK Banjarbaru
Loktabat – Banjarbaru (Tlp. 0511 4782881), e-mail: puslit.stmikbjb@gmail.com
e-ISSN: 2685-0893
p-ISSN: 2089-3787

Penggunaan *Convolutional Neural Network* Sebagai Pengenalan Huruf Bahasa Ibrani

Julian Rusli Tee Baldi[1*], Yohannes[2], Siska Devella[3]
Program Studi Informatika, Universitas Multi Data Palembang, Palembang, Indonesia
*e-mail *Corresponding Author*: julianrtb@mhs.mdp.ac.id

Abstract

Hebrew is important language due to it has a great attachment with Edenics. Edenics is ancestor language of Semitic that broke down within 70 languages for about 3.784 years ago that influenced many languages in the world also have connection to Hebrew. Hebrew has an important role because it's used to study Bible and Mishnah. Research was made as a translate system for Hebrew Letter and the author used 27 of Hebrew letters, using Convolutional Neural Network method with AlexNet architecture. The Hebrew letter recognition made by using the Python.The dataset that used seperated into 27 letters for each of every training and testing data. The amount of training data is 3.638 pictures and testing data is 810 pictures. The highest accuration from 3 optimizers were obtained from Adam optimizer with 81,5% accurate.
Key Words: *AlexNet Architecture; Hebrew, Convolutional Neural Network*

Abstrak

Bahasa Ibrani penting dikarenakan erat hubungannya dengan *Edenics. Edenics* ialah bahasa Ibu Semit yang tersebar ke 70 bahasa kurang lebih 3.784 tahun yang lalu dan berpengaruh besar pada banyak bahasa di bumi serta memiliki keterkaitan dengan bahasa Ibrani. Bahasa Ibrani berperan penting dikarenakan digunakan untuk mempelajari Alkitab dan Mishnah. Penelitian dibuat sebagai sistem penerjemah huruf bahasa Ibrani, dan penulis menggunakan 27 huruf alfabet Ibrani serta menggunakan metode *Convolution Neural Network* dengan arsitektur *AlexNet.* Pengenalan huruf Ibrani dibuat menggunakan *Python*. Dataset yang digunakan terbagi menjadi 27 huruf pada setiap data latih dan uji. Total data latih ialah 3.638 gambar. Total data uji ialah 810 gambar. Penggunaan *optimizer* seperti *Adam*, *SGD* dan *RMSprop* menghasilkan nilai *precision, recall, dan accuracy* yang berbeda*.* Hasil akurasi tertinggi diperoleh dari *optimizer Adam* dengan tingkat akurasi sebesar 81,5%.
Kata Kunci: Arsitektur *AlexNet; Bahasa Ibrani; Convolutional Neural Network*

1. Pendahuluan

Bahasa Ibrani penting dikarenakan bahasa Ibrani erat hubungannya dengan *Edenics.* Menurut Wulandari, *Edenics* merupakan bahasa ibu Semit yang tersebar ke 70 bahasa kurang lebih 3.784 tahun yang lalu [1]. Bahasa Ibrani adalah bahasa resmi yang digunakan negara Israel dan dipakai hampir sebagian bangsa Yahudi yang tersebar di seluruh dunia. Bahasa Ibrani hampir punah selaku bahasa yang dituturkan pada Abad Kuno, namun bahasa Ibrani sangat berperan penting dikarenakan digunakan untuk mempelajari Alkitab dan juga Mishnah. Dengan kemajuan teknologi pada zaman sekarang, memudahkan masyarakat yang sebelumnya tidak paham bahasa Ibrani dapat mempelajari mengenai bahasa Ibrani dengan suatu sistem yang dapat menerjemah huruf Ibrani dalam bentuk penyebutan nama daripada huruf tersebut. Mempelajari dan mengenali bahasa Ibrani dengan baik dikarenakan bahasa Ibrani merupakan bahasa yang berperan penting dalam sejarah bahasa di dunia saat ini. Oleh sebab itu, penelitian ini dibuat untuk membuat sistem penerjemah huruf Ibrani dengan total 27 huruf bahasa Ibrani [2].

Lalu penelitian ini memakai metode *Convolution Neural Network (CNN)*. CNN ialah salah satu jenis jaringan saraf yang paling banyak dipakai untuk data gambar. CNN bisa dipakai untuk mengenali dan mendeteksi juga objek pada suatu citra. Disebabkan CNN tidak berbeda jauh dengan jaringan saraf biasanya tersusun dari sel-sel yang mempunyai bobot, bias dan fungsi aktivasi, hanya CNN yang dilakukan dengan menggunakan filter konvolusi. Lalu juga

penelitian ini menggunakan arsitektur AlexNet. AlexNet adalah satu dari beberapa arsitektur *CNN* yang meraih predikat pemenang di kejuaraan *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)* yang mana ialah kompetisi *image classification* dengan tingkatan luas tahun 2012. *AlexNet* bagaikan terobosan hangat pada *deep learning* dengan memakai *ConvNet* yang dihimpun dengan cara *Dropout Regularization*, penerapan *ReLu* untuk fungsi aktivasi serta *data augmentation*. *AlexNet* dibuat agar bisa dilakukan pengklasifikasian dengan 1000 tingkatan. Arsitektur *Alexnet* terdiri dari 5 *convolution layer*, 2 *dropout layer*, 3 *pooling layer*, dan 3 *fully connected layer* [3].

Berlandaskan latar belakang tersebut, dapat disimpulkan bahwa rumusan masalah dari penelitian ialah bagaimana cara untuk mengidentifikasi huruf bahasa Ibrani menjadi penyebutan nama daripada masing-masing huruf tersebut menggunakan metode *Convolutional Neural Network*.

Tujuan penelitian ialah mengidentifikasi atau melakukan perancangan sistem untuk melakukan pengenalan huruf bahasa Ibrani dengan menggunakan metode *Convolutional Neural Network*.

Manfaat penelitian ialah dapat melakukan penerapan metode *Convolutional Neural Network* dalam mengidentifikasi atau melakukan pengenalan huruf bahasa Ibrani serta mengetahui tingkat akurasi pengenalan huruf bahasa Ibrani dengan menggunakan metode *Convolutional Neural Network*

2. Tinjauan Pustaka

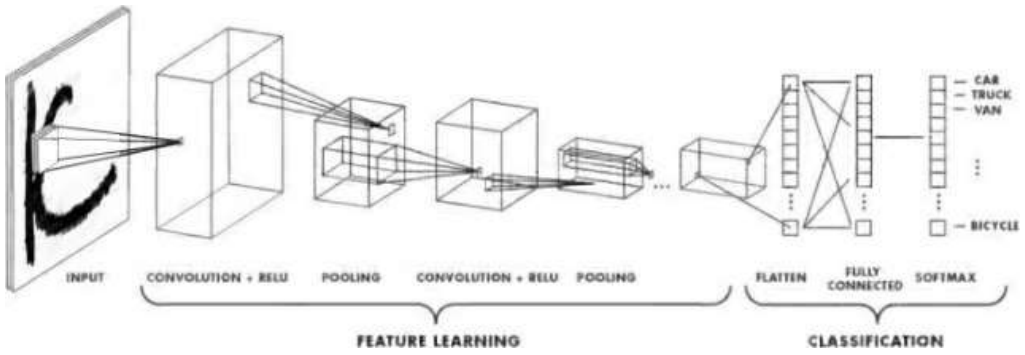
Ditemukan beberapa penelitian dahulu yang menggambarkan dasar dan referensi dari penelitian yang akan dilakukan. CNN digunakan dalam melakukan identifikasi karakter huruf, seperti penelitian yang dilakukan oleh Chaerul Umam, dkk dengan judul *Convolutional Neural Network (CNN)* Untuk Identifikasi Karakter Hiragana. Dengan memakai *optimizer* Adam, hasil yang didapatkan adalah akurasi sebesar 82% dengan menggunakan 872 citra *training* dan 128 citra *testing* menggunakan metode *CNN* [4]. Selain dapat digunakan untuk mengidentifikasi data, CNN juga dapat digunakan untuk melakukan melakukan pengenalan, seperti penelitian yang dilakukan oleh Saufi dengan judul *Deep Learning for Roman Handwritten Character Recognition.* Penelitian ini membandingkan arsitektur AlexNet dan GoogleNet. Hasil dari penelitian ini ialah AlexNet mempunyai akurasi yang berbeda sangat tipis dengan GoogleNet (AlexNet : 94,23%, GoogleNet : 94,47%) namun dengan waktu belajar yang lebih cepat daripada GoogleNet dimana AlexNet memakan waktu 150 *seconds per epoch* sedangkan GoogleNet memakan waktu 200 *seconds per epoch* [5]**.** Selain pengenalan huruf, CNN juga pernah digunakan untuk melakukan pengenalan pada bahasa Arab, seperti pada penelitian yang dikerjakan oleh Nanang Kasim dengan judul penelitian Pengenalan Pola Tulisan Tangan Aksara Arab Menggunakan Metode Convolutional Neural Network. Model yang didapatkan bisa melakukan klasifikasi dataset 8400 citra dengan bagus serta memiliki akurasi 78.10%, presisi 79.68%, dan recall 77.82%. Model bisa menjalankan pengklasifikasian dengan performa yang sangat bagus pada dataset 16800 dengan akurasi 92.64%, presisi 93.29%, dan performa terendah terletak di dataset 25200 dengan akurasi 69.76%, presisi 71.44%, dan *recall* 69.18% [6].

Dikarenakan belum adanya penelitian untuk melakukan pengenalan huruf bahasa Ibrani serta membandingkan 3 *optimizer,* maka dilakukan penelitian untuk pengenalan huruf bahasa Ibrani dengan digunakannya metode *Convolution Neural Network*. Arsitektur yang digunakan adalah AlexNet dengan 3 *optimizer*, yaitu Adam, SGD, dan RMSprop.

3. Metodologi

3.1 Convolutional Neural Network

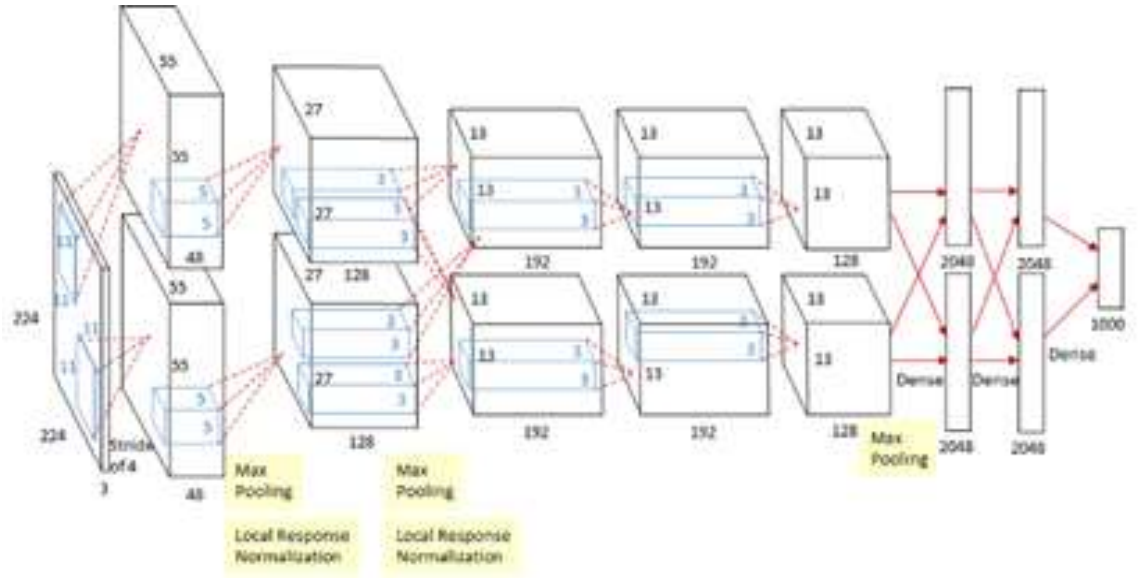
Menurut Saha,*Convolutional Neural Network (CNN)* ialah satu dari beberapa kelas *Deep Learning* yang bisa melaksanakan pengenalan dan klasifikasi gambar [7]. Metode CNN ialah suatu *class* pada neural jaringan yang berkhusus pada menangani data yang punya topologi seperti grid, misalnya gambar. Metode CNN bisa dipakai dalam pengenalan wajah, analisis dokumen, klasifikasi gambar, klasifikasi video, dan sebagainya[6]. Gambar proses *Convolutional Neural Network* bisa ditinjau pada Gambar 3.1.



Gambar 3.1 Proses *Convolutional Neural Network*

3.2 Arsitektur AlexNet

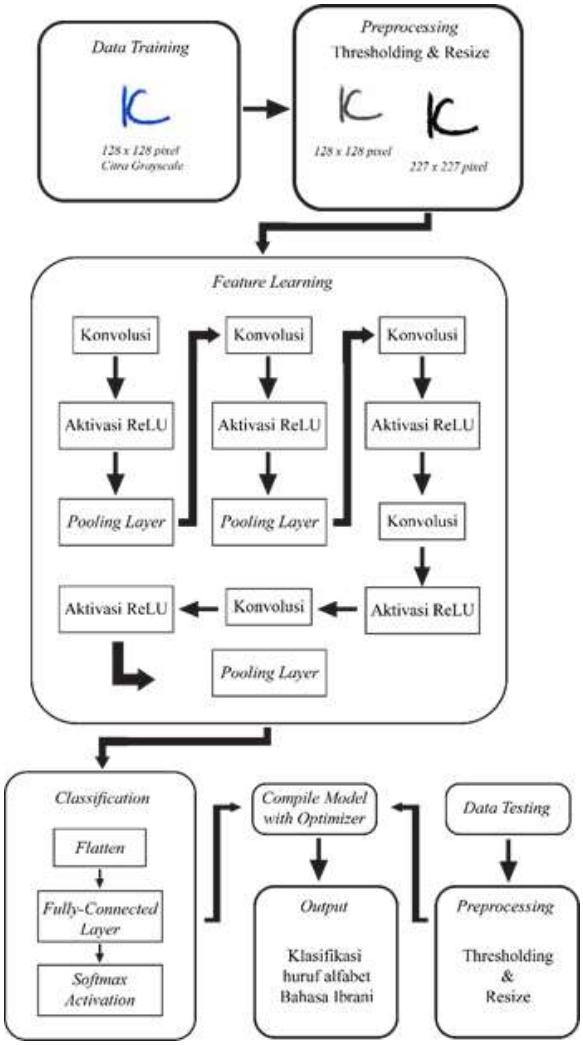
AlexNet adalah salah satu arsitektur *CNN* yang meraih predikat juara pada kejuaraan *ImageNet Large Scale Visual Recognition Challenge (ILSVRC)* yang mana ialah kompetisi *image classification* dengan skala luas tahun 2012. *AlexNet* bagaikan terobosan hangat pada *deep learning* dengan memakai *ConvNet* yang dihimpun dengan teknik *Dropout Regularization*, penerapan *ReLu* untuk fungsi aktivasi serta *data augmentation*. *AlexNet* dibuat agar bisa dilakukan pengklasifikasian dengan 1000 tingkatan. Arsitektur *Alexnet* terdiri dari 5 *convolution layer*, 3 *pooling layer*, 2 *dropout layer*, dan 3 *fully connected layer* [8]. Arsitektur Alexnet bisa ditinjau pada Gambar 3.2.



Gambar 3.2 Arsitektur *AlexNet*

3.3 Perancangan Sistem

Pada tahap perancangan ini, diperlukan perancangan dan juga sistem penelitian untuk melakukan penelitian, yaitu penggunaan metode *Convolutional Neural Network (CNN)* untuk melakukan pengenalan huruf bahasa Ibrani berdasarkan citra gambar. Dataset yang dipakai dipecah menjadi 2, antara lain data *training* dan data *testing.* Jumlah data *training* adalah 27 huruf x 135 gambar = 3.638 gambar. Jumlah data *testing* adalah 27 huruf x 30 gambar = 810 gambar*,* sehingga total data gambar adalah 4.448. Diagram Perancangan sistem dapat dilihat pada Gambar 3.3. dan model perancangan sistem dapat dilihat pada Tabel 3.1.



Gambar 3.3 Diagram Perancangan Sistem

Tabel 3.1 Model Perancangan Sistem

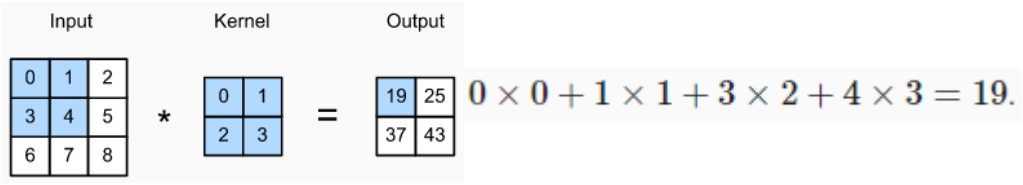
Layer (Type)	*Output Shape*	*Params*
Conv2d_10 (Conv2D)	*(None, 55, 55, 96)*	*34944*
Batch_normalization_10 (BatchNormalization)	*(None, 55, 55, 96)*	*384*
Max_pooling2d 6 (MaxPooling2D)	*(None, 27, 27, 96)*	*0*
Conv2d_11 (Conv2D)	*(None, 27, 27, 256)*	*614656*
Batch_normalization_11 (BatchNormalization)	*(None, 27, 27, 256)*	*1024*
Max_pooling2d 7 (MaxPooling2D)	*(None, 13, 13, 256)*	*0*
Conv2d_12 (Conv2D)	*(None, 13, 13, 384)*	*885120*
Batch_normalization_12 (BatchNormalization)	*(None, 13, 13, 384)*	*1536*
Conv2d_13 (Conv2D)	*(None, 13, 13, 384)*	*147840*
Batch_normalization_13 (BatchNormalization)	*(None, 13, 13, 384)*	*1536*
Conv2d_14 (Conv2D)	*(None, 13, 13, 256)*	*98560*
Batch_normalization_14 (BatchNormalization)	*(None, 13, 13, 256)*	*1024*
Max_pooling2d 8 (MaxPooling2D)	*(None, 6, 6, 256)*	*0*
Flatten 2 (Flatten)	*(None, 9216)*	*0*
Dense_7(Dense)	*(None, 4096)*	*37752832*

Layer (Type)	*Output Shape*	*Params*
Dropout 5(Dropout)	*(None, 4096)*	*0*
Dense_8(Dense)	*(None, 4096)*	*16781312*
Dropout 6(Dropout)	*(None, 4096)*	*0*
Dense_9(Dense)	*(None, 27)*	*110619*
Total param		*56,431,387*
Trainable params		*56,428,635*
Non_trainable params		*2,752*

4. Hasil dan Pembahasan

Pada penelitian, pengujian penelitian dilaksanakan memakai metode Convolutional Neural Network dengan arsitektur Alexnet serta optimizer Adam, SGD, dan RMSprop. Dataset yang dipakai [9] dipecah menjadi 2, antara lain data *training* dan data *testing.* Jumlah data *training* adalah 27 huruf x 135 gambar = 3.638 gambar. Jumlah data *testing* adalah 27 huruf x 30 gambar = 810 gambar, sehingga total data gambar adalah 4.448. Pengujian dilakukan dengan jumlah epoch sebesar 20, batch size sebesar 40, dan learning rate 0,0001. Lalu juga dilakukan perubahan untuk beberapa parameter yang ada sehingga menghasilkan sebuah model yang akan digunakan untuk untuk menghasilkan tingkat akurasi yang tinggi dalam mengidentifikasi / pengenalan huruf bahasa Ibrani. Beberapa parameter yang dipakai antara lain data yang dipakai yaitu data *testing* yang melalui proses *preprocessing* yang meliputi perubahan ke dalam *grayscale, thresholding,* dan *resize,* lalu menggunakan arsitektur *AlexNet* serta *optimizer* berupa Adam, SGD, dan RMSprop.

Pertama, dilakukan proses *convolution*. Berikut perumpamaan proses *convolution* dengan kernel 2x2. Perhitungan proses *convolution* dapat dilihat pada Gambar 4.1.



Gambar 4.1 Perhitungan Proses *Convolution*

Sumber : https://d2l.ai/chapter_convolutional-neural-networks/conv-layer.html

Untuk menghitung ukuran output dari *Convolutional Layer* dapat menggunakan rumus pada Gambar 4.2 dimana ukuran output sama dengan ukuran input ditambah 2 kali *padding* dikurangi ukuran kernel di atas *stride* plus satu.

$$n_{out} = \frac{n_{in}+2p-k}{s}+1$$

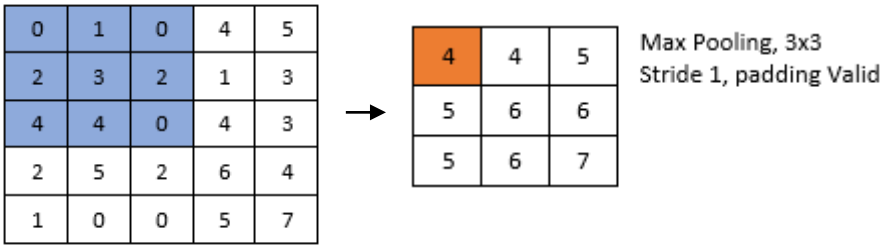
Gambar 4.2 Rumus Output *Convolutional Layer*

Kemudian dilanjutkan proses *Activation ReLU* dimana berfungsi untuk memberikan output 0 jika input negatif atau 0 dan nilai itu sendiri jika input bernilai positif. Rumus ReLU dapat dilihat pada Gambar 4.3

$$f(x) = \max(0, x)$$

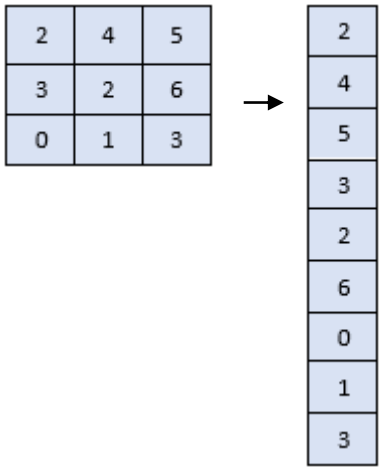
Gambar 4.3 Rumus *Activation ReLU*

Pooling Layer dilakukan setelah *Activation ReLU.* Perhitungan max pooling diambil dari ukuran size pooling dimana dalam gambar tersebut 3x3. Kemudian diambil bilangan yang paling besar (*max*) dan di *convert* ke dalam matriks berukuran 3x3. Proses *max pooling* dapat dilihat pada Gambar 4.4.



Gambar 4.4 Proses *Max Pooling*

Kemudian setelah melalui proses tersebut, dilanjutkan bagian *classification* dimana terdapat proses *flattening.* Proses ini berfungsi mengubah matriks pada *pooling layer* menjadi sebuah vektor tunggal. Ilustrasi proses *flattening* dapat dilihat pada Gambar 4.5



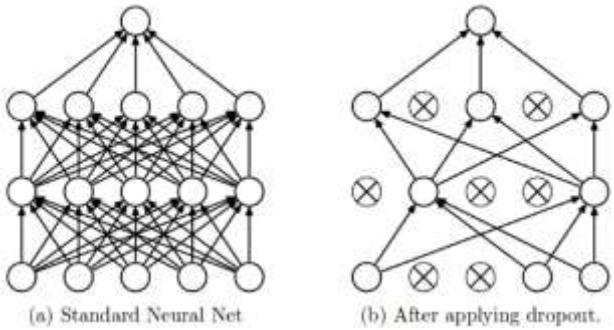
Gambar 4.5 Proses *Flattening*

Setelah itu masuk ke proses fully connected yang akan dilanjutkan ke proses softmax dan dropout. Persamaan untuk transformasi non-linear untuk fully connected dapat dilihat pada Gambar 4.6

$$y_{jk}(x) = f\left(\sum_{i=1}^{n_H} w_{jk}x_i + w_{j0}\right)$$

Gambar 4.6 Persamaan Transformasi Non-Linear Fully Connected

Lalu proses *dropout* berfungsi untuk menghindari *overfitting.* Dan proses serta rumusnya dapat dilihat pada Gambar 4.7 dan 4.8.



https://laid.delanover.com/dropout-explained-and-implementation-in-tensorflow/

Gambar 4.7 Proses *Dropout*

The feed-forward operation of a standard neural network

$$z_i^{(l+1)} = \mathbf{w}_i^{(l+1)}\mathbf{y}^l + b_i^{(l+1)},$$
$$y_i^{(l+1)} = f(z_i^{(l+1)}),$$

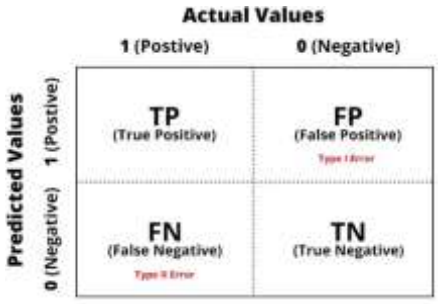
With dropout, the feed-forward operation becomes

$$r_j^{(l)} \sim \text{Bernoulli}(p),$$
$$\tilde{\mathbf{y}}^{(l)} = \mathbf{r}^{(l)} * \mathbf{y}^{(l)},$$
$$z_i^{(l+1)} = \mathbf{w}_i^{(l+1)}\tilde{\mathbf{y}}^l + b_i^{(l+1)},$$
$$y_i^{(l+1)} = f(z_i^{(l+1)}).$$

https://laid.delanover.com/dropout-explained-and-implementation-in-tensorflow/

Gambar 4.8 Rumus Dropout

Confusion Matrix berbentuk table matriks yang berfungsi untuk memberikan informasi perbandingan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi sebenarnya. Berikut contoh confusion matrix dengan 4 kombinasi nilai prediksi dan nilai aktual berbeda pada Gambar 4.9



Sumber : https://ksnugroho.medium.com/confusion-matrix-untuk-evaluasi-model-pada-unsupervised-machine-learning-bc4b1ae9ae3f

Gambar 4.9 Confusion Matrix 4 Kombinasi

Precision berfungsi menghitung tingkat perbandingan jumlah data yangmana diprediksi benar dengan total keseluruhan data yang diprediksi. Kemudian *Recall* berfungsi untuk mengevaluasi perbandingan jumlah data yang diprediksi benar dengan total keseluruhan data, dan secara keseluruhan performa yang dihasilkan oleh sistem ditunjukan dengan menggunakan *Accuracy* yang berfungsi untuk mengukur tingkat keberhasilan identifikasi dengan tepat.

Rumus *Precision, Accuracy,* dan *Recall* ditunjukkan pada Persamaan 4.1, 4.2, dan 4.3.

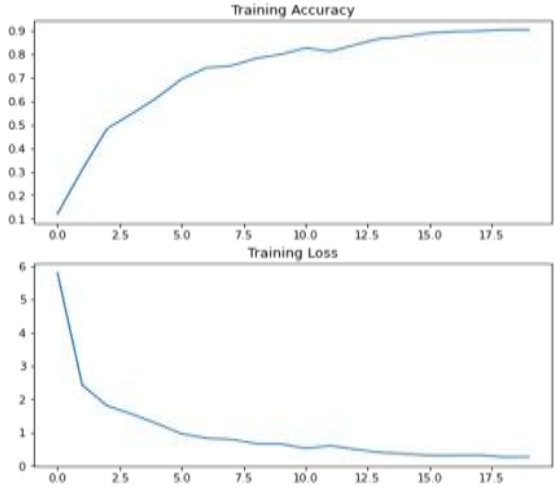
$$precision = \frac{TP}{TP + FP} \quad \text{.....(4.1)}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad \text{.....(4.2)}$$

$$recall = \frac{TP}{TP + FN} \quad \text{.....(4.3)}$$

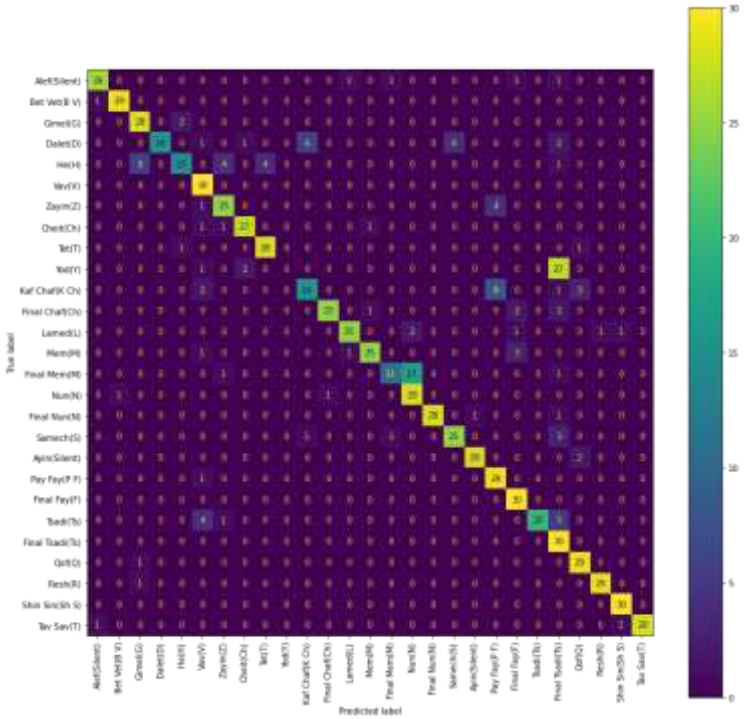
4.1 Hasil Pengujian dengan Optimizer Adam

Hasil dari pengujian sistem untuk pengenalan huruf bahasa Ibrani dengan memakai optimizer Adam dapat ditinjau pada Gambar 4.10, dimana menunjukan grafik accuracy dan loss. Jumlah epoch yang digunakan adalah 20, dimana pada epoch pertama diperoleh akurasi sebesar 12% , lalu akurasi meningkat pada epoch kedua menjadi 30,98%, lalu pada epoch ke 11 naik menjadi 82,68%. Pada epoch ke 12 terdapat penurunan ke 81,17%, namun pada epoch berikutnya kembali naik hingga pada epoch-epoch selanjutnya diperoleh hasil akurasi sebesar 90,4%.



Gambar 4.10 Grafik *Training* dengan Optimizer Adam

Pengujian sistem dilakukan dengan memasukkan data *testing* dan menghasilkan *confusion matrix* yang bisa dilihat pada Gambar 4.11



Gambar 4.11. *Confusion Matrix* Optimizer Adam

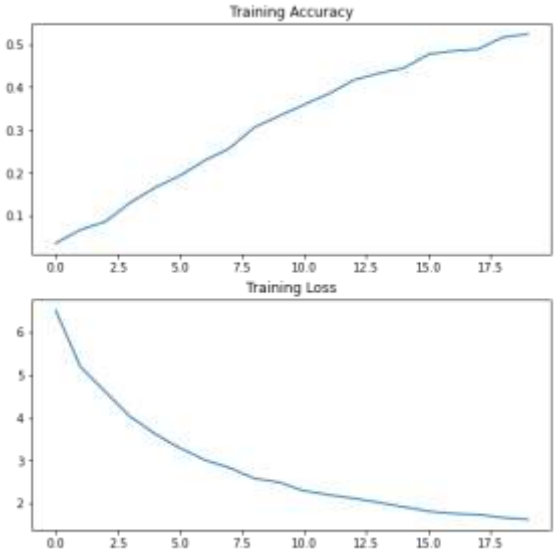
Hasil *precision, recall, f1-score* dan *accuracy* pada setiap *class* dengan optimizer Adam bisa ditinjau pada Tabel 4.1.

Tabel 4.1. Hasil Pengujian dengan Optimizer Adam

No.	Huruf	*precision*	*recall*	*f1-score*	*support*
1	Alef(Silent)	0,929	0,867	0,897	30
2	Bet Vet(B V)	0,967	0,967	0,967	30
3	Gimel(G)	0,778	0,933	0,848	30
4	Dalet(D)	1,000	0,533	0,696	30
5	He(H)	0,833	0,500	0,625	30
6	Vav(V)	0,714	1,000	0,833	30
7	Zayin(Z)	0,781	0,833	0,806	30
8	Cheit(Ch)	0,900	0,900	0,900	30
9	Tet(T)	0,875	0,933	0,903	30
10	Yod(Y)	0,000	0,000	0,000	30
11	Kaf Chaf(K Ch)	0,696	0,533	0,604	30
12	Final Chaf(Ch)	0,962	0,833	0,893	30
13	Lamed(L)	0,926	0,833	0,877	30
14	Mem(M)	0,926	0,833	0,877	30
15	Final Mem(M)	0,846	0,367	0,512	30
16	Nun(N)	0,596	0,933	0,727	30
17	Final Nun(N)	1,000	0,933	0,966	30
18	Samech(S)	0,862	0,833	0,847	30
19	Ayin(Silent)	0,966	0,933	0,949	30
20	Pay Fay(P F)	0,707	0,967	0,817	30
21	Final Fay(F)	0,811	1,000	0,896	30
22	Tsadi(Ts)	1,000	0,667	0,800	30
23	Final Tsadi(Ts)	0,405	1,000	0,577	30
24	Qof(Q)	0,829	0,967	0,892	30
25	Resh(R)	0,967	0,967	0,967	30
26	Shin Sin(Sh S)	0,938	1,000	0,968	30
27	Tav Sav(T)	1,000	0,933	0,966	30
	accuracy			0,815	810
	macro avg	0,823	0,815	0,800	810
	weighted avg	0,823	0,815	0,800	810

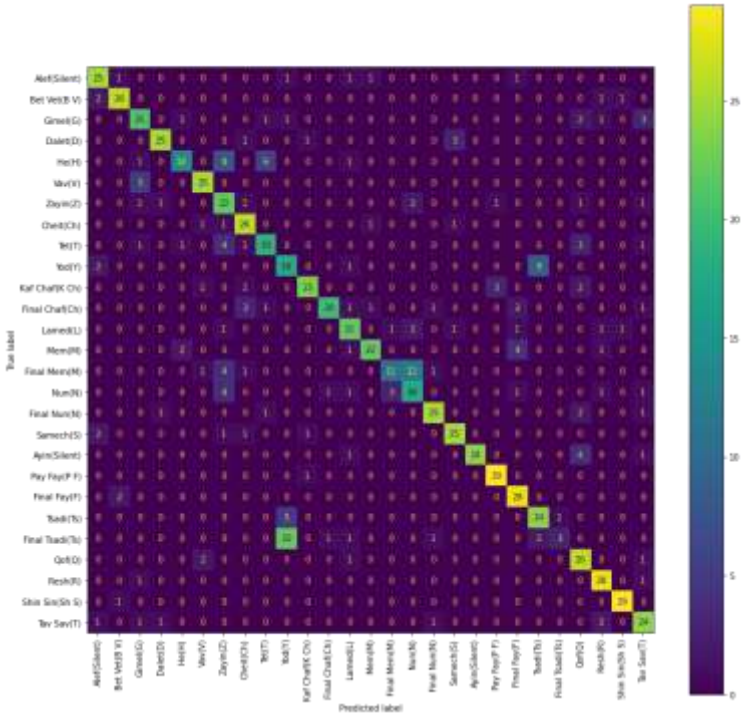
4.2 Hasil Pengujian dengan Optimizer SGD

Hasil dari pengujian sistem untuk pengenalan huruf bahasa Ibrani dengan menggunakan optimizer SGD dapat dilihat pada Gambar 4.12, dimana menunjukan grafik *accuracy* dan *loss*. Jumlah epoch yang digunakan adalah 20, dimana pada epoch pertama diperoleh akurasi sebesar 3,66% , lalu akurasi meningkat pada epoch kedua menjadi 6,79%, lalu . Pada epoch ke-16 mencapai akurasi 47,6%. Hingga pada epoch berikutnya diperoleh hasil akurasi sebesar 52,34%.



Gambar 4.12. Grafik *Training* dengan Optimizer SGD

Pengujian sistem dilakukan dengan memasukkan data *testing* dan menghasilkan *confusion matrix* yang bisa ditinjau pada Gambar 4.13.



Gambar 4.13. Confusion Matrix Optimizer SGD

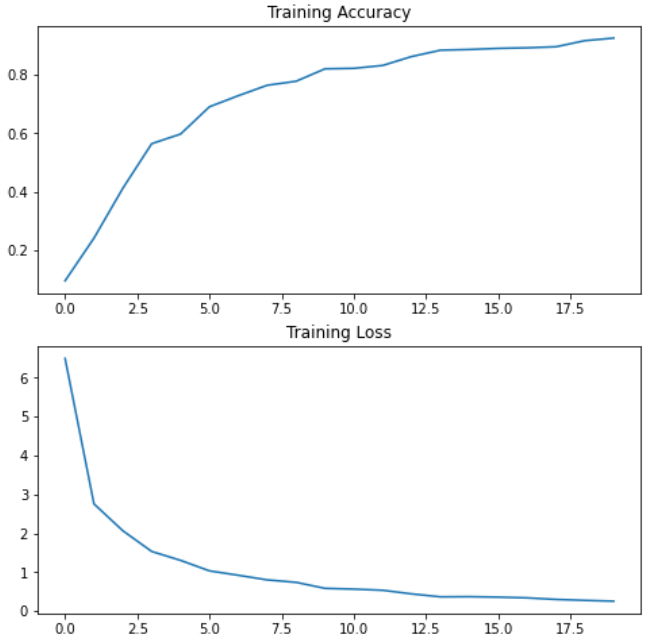
Hasil *precision, recall, f1-score* dan *accuracy* pada setiap *class* dengan optimizer SGD bisa ditinjau pada Tabel 4.2.

Tabel 4.2. Hasil Pengujian dengan Optimizer SGD

No.	Huruf	*precision*	*recall*	*f1-score*	*support*
1	Alef(Silent)	0,781	0,833	0,806	30
2	Bet Vet(B V)	0,867	0,867	0,867	30
3	Gimel(G)	0,677	0,700	0,689	30
4	Dalet(D)	0,893	0,833	0,862	30
5	He(H)	0,778	0,467	0,583	30
6	Vav(V)	0,833	0,833	0,833	30
7	Zayin(Z)	0,489	0,733	0,587	30
8	Cheit(Ch)	0,743	0,867	0,800	30
9	Tet(T)	0,679	0,633	0,655	30
10	Yod(Y)	0,383	0,600	0,468	30
11	Kaf Chaf(K Ch)	0,885	0,767	0,821	30
12	Final Chaf(Ch)	0,909	0,667	0,769	30
13	Lamed(L)	0,710	0,733	0,721	30
14	Mem(M)	0,880	0,733	0,800	30
15	Final Mem(M)	0,733	0,367	0,489	30
16	Nun(N)	0,529	0,600	0,562	30
17	Final Nun(N)	0,862	0,833	0,847	30
18	Samech(S)	0,833	0,833	0,833	30
19	Ayin(Silent)	1,000	0,800	0,889	30
20	Pay Fay(P F)	0,879	0,967	0,921	30
21	Final Fay(F)	0,757	0,933	0,836	30
22	Tsadi(Ts)	0,686	0,800	0,738	30
23	Final Tsadi(Ts)	0,750	0,100	0,176	30
24	Qof(Q)	0,650	0,867	0,743	30
25	Resh(R)	0,800	0,933	0,862	30
26	Shin Sin(Sh S)	0,935	0,967	0,951	30
27	Tav Sav(T)	0,686	0,800	0,738	30
	accuracy			0,743	810
	macro avg	0,763	0,743	0,735	810
	weighted avg	0,763	0,743	0,735	810

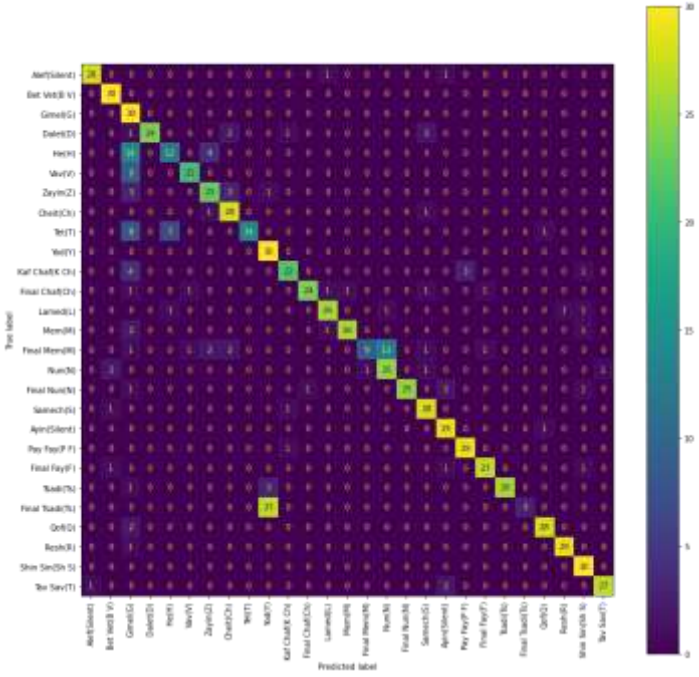
4.3 Hasil Pengujian dengan Optimizer RMSprop

Hasil dari pengujian sistem untuk pengenalan huruf bahasa Ibrani dengan memakai optimizer RMSprop bisa ditinjau pada Gambar 4.14, dimana menunjukan grafik accuracy dan loss. Jumlah epoch yang digunakan adalah 20, dimana pada epoch pertama diperoleh akurasi sebesar 9,4% , lalu akurasi meningkat pada epoch kedua menjadi 24%, lalu . Pada epoch ke-10, akurasi meningkat menjadi 82%, hingga pada epoch berikutnya diperoleh hasil akurasi sebesar 92,58%.



Gambar 4.14. Grafik *Training* dengan Optimizer RMSprop

Pengujian sistem dilakukan dengan memasukkan data testing dan menghasilkan *confusion matrix* yang bisa ditinjau pada Gambar 4.15.



Gambar 4.15. *Confusion Matrix* Optimizer RMSprop

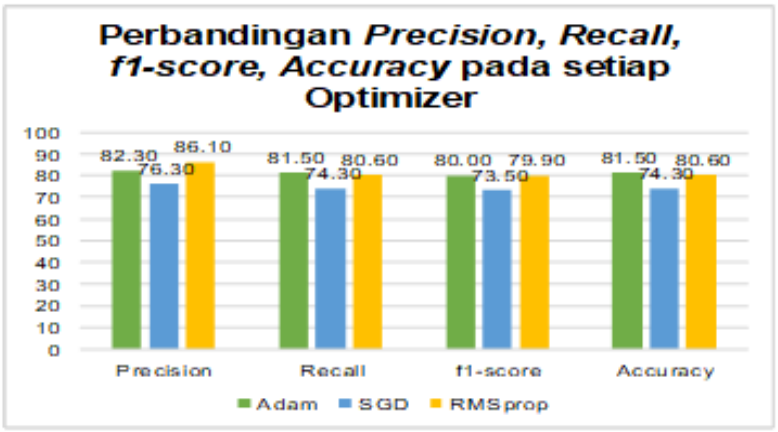
Hasil precision, recall, f1-score dan accuracy pada setiap class dengan optimizer RMSprop dapat dilihat pada Tabel 4.3.

Tabel 4.3. Hasil Pengujian dengan Optimizer RMSprop

No.	Huruf	*precision*	*recall*	*f1-score*	*support*
1	Alef(Silent)	0,966	0,933	0,949	30
2	Bet Vet(B V)	0,882	1,000	0,938	30
3	Gimel(G)	0,385	1,000	0,556	30
4	Dalet(D)	1,000	0,800	0,889	30
5	He(H)	0,600	0,400	0,480	30
6	Vav(V)	0,913	0,700	0,792	30
7	Zayin(Z)	0,767	0,767	0,767	30
8	Cheit(Ch)	0,800	0,933	0,862	30
9	Tet(T)	1,000	0,467	0,636	30
10	Yod(Y)	0,492	1,000	0,659	30
11	Kaf Chaf(K Ch)	0,880	0,733	0,800	30
12	Final Chaf(Ch)	0,960	0,800	0,873	30
13	Lamed(L)	0,897	0,867	0,881	30
14	Mem(M)	0,963	0,867	0,912	30
15	Final Mem(M)	0,900	0,300	0,450	30
16	Nun(N)	0,641	0,833	0,725	30
17	Final Nun(N)	1,000	0,833	0,909	30
18	Samech(S)	0,824	0,933	0,875	30
19	Ayin(Silent)	0,829	0,967	0,892	30
20	Pay Fay(P F)	0,906	0,967	0,935	30
21	Final Fay(F)	0,931	0,900	0,915	30
22	Tsadi(Ts)	1,000	0,867	0,929	30
23	Final Tsadi(Ts)	1,000	0,100	0,182	30
24	Qof(Q)	0,933	0,933	0,933	30
25	Resh(R)	0,967	0,967	0,967	30
26	Shin Sin(Sh S)	0,857	1,000	0,923	30
27	Tav Sav(T)	0,964	0,900	0,931	30
	accuracy			0,806	810
	macro avg	0,861	0,806	0,799	810
	weighted avg	0,861	0,806	0,799	810

4.4 Analisis Hasil Pengujian

Hasil dari pengujian sistem untuk pengenalan huruf bahasa Ibrani dengan menggunakan 3 optimizer yang berbeda, yaitu Adam, SGD, dan RMSprop didapatkan hasil yang berbeda-beda. Perbandingan dari hasil *precision, recall, f1-score*, dan *accuracy* bisa ditinjau pada Gambar 4.16.



Gambar 4.16. Grafik Perbandingan Hasil Optimizer Adam, SGD, dan RMSprop

Pada Gambar 4.16 menunjukkan penggunaan optimizer Adam mendapatkan akurasi tertinggi dibandingkan dengan optimizer SGD dan RMSprop, dengan nilai *precision* sebesar 82,3%, *recall* sebesar 81,5%, *f1-score* 80%, dan *accuracy* sebesar 81,5%. Sehingga dapat disimpulkan pada penggunaan optimizer Adam pada arsitektur AlexNet mendapatkan hasil yang cukup baik dalam melakukan pengenalan huruf bahasa Ibrani. Lalu, pada Gambar 4.7 menunjukkan bahwa penggunaan optimizer SGD mendapatkan akurasi terendah dibandingkan dengan optimizer Adam dan RMSprop, dimana terlihat bahwa nilai *precision* yang didapat adalah sebesar 76,3%, *recall* sebesar 74,3%, *f1-score* 73,5%, dan *accuracy* sebesar 74,3%.

Dari data tersebut, dapat disimpulkan pada penggunaan optimizer Adam pada arsitektur AlexNet dengan memakai metode Convolutional Neural Network mendapatkan hasil yang cukup bagus dalam melakukan pengenalan huruf bahasa Ibrani. Hal tersebut selaras dengan penelitian terdahulu yang dilakukan oleh Ocktavia Nurima [10] dan Irfansyah serta didukung karena Adam algoritma optimisasi yang menggabungkan unsur-unsur terbaik dari algoritma AdaGrad dan RMSprop [11]. Lalu juga terbukti bahwa metode *Convolutional Neural Network* dapat melakukan pengenalan dengan baik selaras dengan penelitian terdahulu yang dilakukan oleh Putra, dkk [12], M. Al Rivan [13] [14], Lorentius [15], dan [16] sehingga disarankan untuk digunakan pada penelitian berikutnya.

5. Simpulan

Kesimpulan dari penelitian adalah arsitektur Alexnet dapat melakukan pengenalan huruf bahasa Ibrani dengan cukup baik dan akurat. Hal tersebut didukung dengan hasil yang didapat seperti; *optimizer* dengan akurasi tertinggi didapatkan oleh *optimizer* Adam dibandingkan dengan *optimizer* SGD dan RMSprop. Hal tersebut didukung karena Adam algoritma optimisasi yang menggabungkan unsur-unsur terbaik dari algoritma AdaGrad dan RMSprop. Di lain sisi, penggunaan *optimizer* SGD kurang optimal dan mendapatkan akurasi terendah dibandingkan dengan optimizer Adam dan RMSprop. Sehingga pada penelitian ini, *optimizer* Adam pada metode *Convolutional Neural Network* arsitektur Alexnet pada pengenalan huruf bahasa Ibrani mendapatkan nilai *precision* sebesar 82,3%, *recall* sebesar 81,5%, f1-score sebesar 80% serta accuracy sebesar 81,5%.

Saran dari penulis mengenai pengembangan dan perbaikan penelitian ini adalah menggunakan algoritma *Deep Learning* yang lebih beragam dalam melakukan proses pengenalan. Lalu juga menggunakan data dengan jumlah yang lebih banyak dan bervariasi agar pengenalan dapat diproses dengan lebih baik.

Daftar Referensi

[1] W. R. Wulandari, “Edenic Language Sebagai Bahasa Ibrani : Ibu Bagi Seluruh Bahasa Di Dunia,” *J. Teol. dan Pendidik. Kristen*, vol. 1, pp. 155–174, 2021, doi: 10.46362/didache.vli2.37.

[2] I. Indha Oytesarp, *Pengenalan dasar-dasar Huruf Hebrew*. 2019.

[3] D. Irfansyah, M. Mustikasari, and A. Suroso, “Arsitektur Convolutional Neural Network (CNN) Alexnet Untuk Klasifikasi Hama Pada Citra Daun Tanaman Kopi,” *J. Inform. J. Pengemb. IT*, vol. 6, no. 2, pp. 87–92, 2021, [Online]. Available: http://ejournal.poltektegal.ac.id/index.php/informatika/article/view/2802

[4] C. Umam and L. Budi Handoko, “Convolutional Neural Network (CNN) Untuk Identifkasi Karakter Hiragana,” *Pros. Semin. Nas. Lppm Ump*, vol. 0, no. 0, pp. 527–533, 2020, [Online]. Available: https://semnaslppm.ump.ac.id/index.php/semnaslppm/article/view/199

[5] M. M. Saufi, M. A. Zamanhuri, N. Mohammad, and Z. Ibrahim, “Deep Learning for Roman Handwritten Character Recognition,” *Indones. J. Electr. Eng. Comput. Sci.*, vol. 12, no. 2, pp. 455–460, 2018, doi: 10.11591/ijeecs.v12.i2.pp455-460.

[6] N. Kasim, “Pengenalan Pola Tulisan Tangan Aksara Arab Menggunakan Metode Convolutional Neural Network,” *Jurnal Teknologi Informasi, Komputer, dan Aplikasinya (JTIKA)*, vol. 3, no. 1, pp. 85-95. 2021.

[7] S. Saha, “A Comprehensive Guide to Convolutional Neural Networks — the ELI5 waytle,” 2018. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet Classification with Deep Convolutional Neural Networks,” *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017,

[Online]. Available: https://dl.acm.org/doi/10.1145/3065386
[9] I. Rabaev, K. Barakat, Churkin, and El-Sana, “The Handwritten Hebrew Dataset (HHD_v0),” *The 17th International Conference on Frontiers in Handwriting Recognition*, 2020. https://tc11.cvc.uab.es/datasets/HHD_v0_1 (accessed Oct. 20, 2022).
[10] O. N. Putri, “Implemetasi Metode CNN Dalam Klasifikasi Gambar Jamur pada Analisis Image Processing,” Tugas AKhir, Universitas Islam Indonesia, 2020.
[11] D. P. Kingma and J. L. Ba, “Adam : A Method For Stochastic Optimization,” 2015.
[12] O. V. Putra, A. Musthafa, M. Nur, and M. Rido, “Classification of Calligraphy Writing Types Using Convolutional Neural Network Method (CNN),” *Procedia Eng. Life Sci.*, vol. 2, no. October, pp. 1–7, 2021, doi: 10.21070/pels.v2i0.1136.
[13] M. E. Al Rivan and A. Setiawan, “Pengenalan Gestur Angka Pada Tangan Menggunakan Arsitektur AlexNet Dan LeNet Pada Metode Convolutional Neural Network,” *Komputika J. Sist. Komput.*, vol. 11, no. 1, pp. 19–28, 2022, doi: 10.34010/komputika.v11i1.5176.
[14] M. E. Al Rivan and A. G. Riyadi, “Perbandingan Arsitektur LeNet dan AlexNet Pada Metode Convolutional Neural Network Untuk Pengenalan American Sign Language,” *J. Komput. Terap.*, vol. 7, pp. 53–61, 2021.
[15] C. A. Lorentius, R. Adipranata, and A. Tjondrowiguno, “Pengenalan Aksara Jawa dengan Menggunakan Metode Convolutional Neural Network,” Tugas Akhir, Universitas Kristen Petra, 2019.
[16] Y.F. Riti, & S.S. Tandjung, "Klasifikasi Covid-19 Pada Citra CT Scans Paru-Paru Menggunakan Metode Convolution Neural Network". *Progresif: Jurnal Ilmiah Komputer*, vol. 18, no. 1, pp. 91-100, 2022.