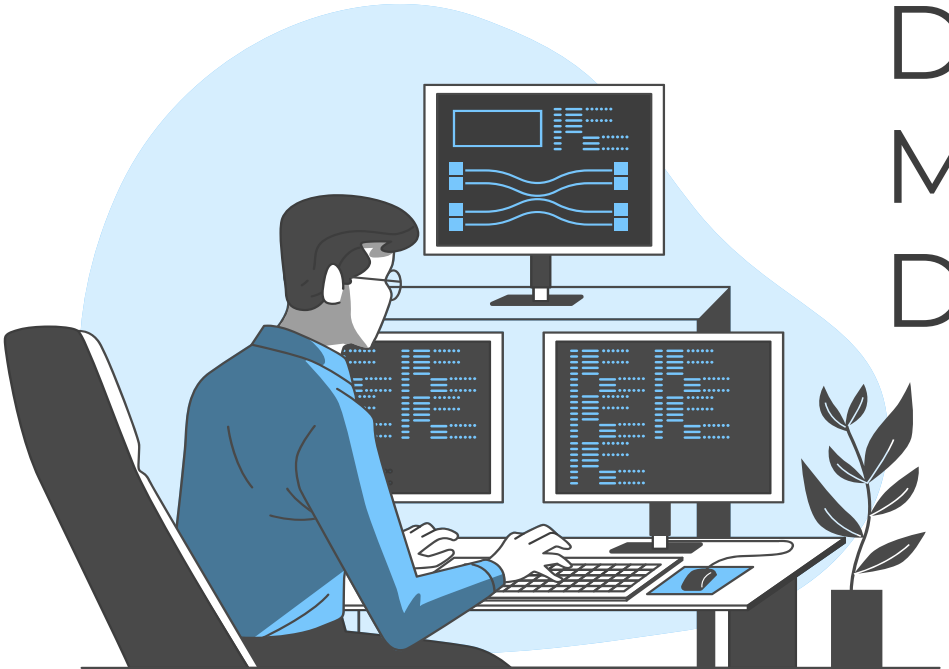
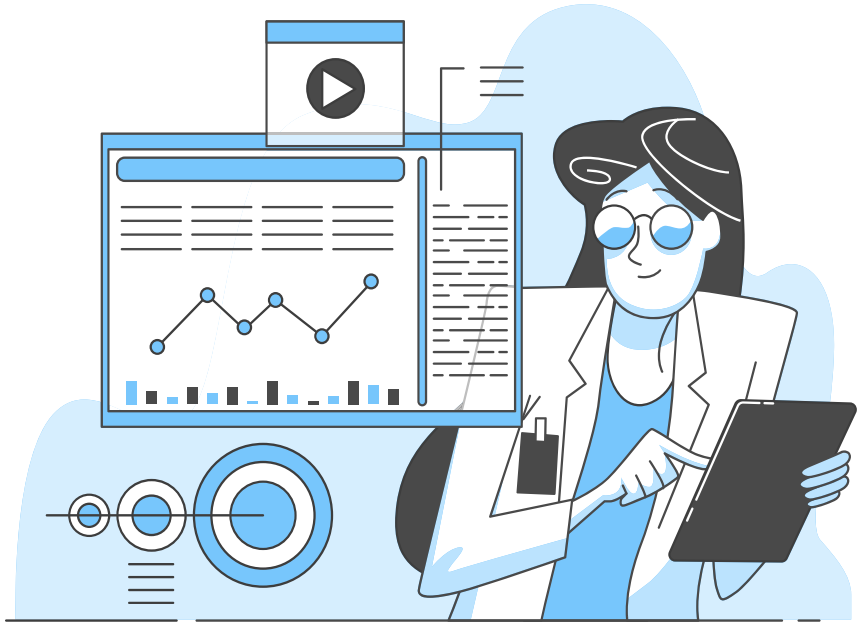


DenseNet Dengan MNIST Dataset Digit Recognizer

Table of Contents



Data Acquisition and Collection

Pada tugas ini data yang diambil adalah dari kaggle
https://www.kaggle.com/code/szaitseff/under-the-hood-a-dense-net-w-mnist-dataset/data

File data train.csv dan test.csv berisi gambar skala abu-abu dari angka yang digambar tangan, dari nol hingga sembilan.

Setiap gambar memiliki tinggi 28 piksel dan lebar 28 piksel, dengan total 784 piksel. Setiap piksel memiliki nilai piksel tunggal yang terkait dengannya, yang menunjukkan terang atau gelapnya piksel tersebut, dengan angka yang lebih tinggi berarti lebih gelap. Nilai piksel ini adalah bilangan bulat antara 0 dan 255, inklusif.

Kumpulan data pelatihan, (train.csv), memiliki 785 kolom. Kolom pertama, yang disebut "label", adalah angka yang ditarik oleh pengguna. Kolom lainnya berisi nilai piksel dari gambar terkait.

Setiap kolom piksel dalam set pelatihan memiliki nama seperti piksel, di mana x adalah bilangan bulat antara 0 dan 783, inklusif. Untuk menempatkan piksel ini pada gambar, misalkan kita telah menguraikan x sebagai x = i * 28 + j, di mana i dan j adalah bilangan bulat antara 0 dan 27, inklusif. Kemudian piksel terletak pada baris i dan kolom j dari matriks 28 x 28, (diindeks dengan nol).

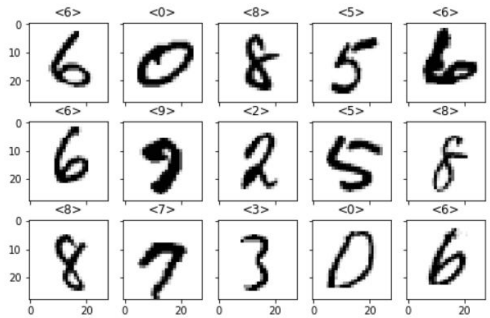
Data Preprocessing

Melakukan shaping pada data train yang sudah diberi label

```
train_pixels, test_pixels = train.iloc[:,1:].values.astype('float32'),
test.values.astype('float32')
train_labels = train.iloc[:,0].values.astype('int32')
train_labels = train_labels.reshape(-1, 1)
```

Melakukan visualisasi data yang sudah dilabeli

```
nrows, ncols = 3, 5  # number of rows and colums in subplots
fig, ax = plt.subplots(nrows, ncols, sharex=True, sharey=True, figsize=(8,5))
for row in range(nrows):
    for col in range(ncols):
        i = np.random.randint(0, 30000)  # pick up arbitrary examples
        ax[row, col].imshow(train_pixels[i,:,:,0], cmap='Greys')
        ax[row, col].set_title(f'<{train.label[i]}>');
```



Data Validate

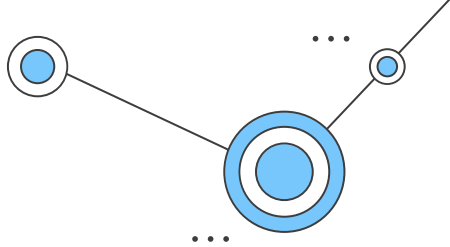
Melakukan split untuk data train dan validatenya

```
((37800, 28, 28, 1),
 (37800, 10),
 (4200, 28, 28, 1),
 (4200, 10),
 (28000, 28, 28, 1))
```

Melakukan fixing untuk angka dengan membantu modelling nanti

`m_train = 37800 / m_val = 4200 / m_test = 28000 / n_x = 784 / n_y = 10`

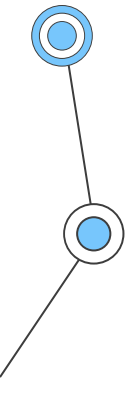
Neural Network Models



DenseNet adalah jaringan konvolusi yang terhubung secara padat. Ini sangat mirip dengan ResNet dengan beberapa perbedaan mendasar. ResNet menggunakan metode aditif yang berarti mereka mengambil output sebelumnya sebagai input untuk lapisan masa depan, & di DenseNet mengambil semua output sebelumnya sebagai input untuk lapisan masa depan seperti yang ditunjukkan pada gambar di atas.

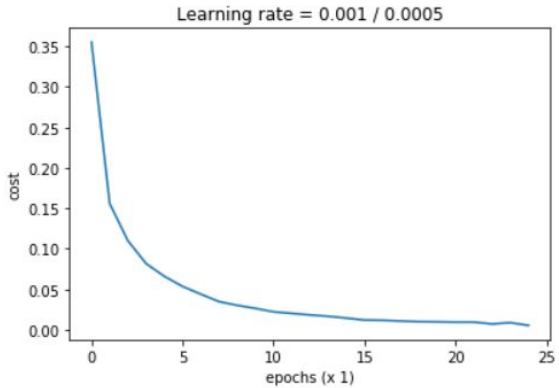
Pada kode ini DenseNet yang digunakan adalah dari modul Keras, diperoleh hasilnya adaah:

```
37800/37800 [==============================] - 1s 36us/step
4200/4200 [==============================] - 0s 38us/step
k_model: train accuracy = 99.9894%
k_model: val accuracy = 98.1429%
k_model: val error = 78.0 examples
```



Training dan Testing Model DenseNet

- Training




```
c_model: train accuracy: 100.0%
c_model: val accuracy = 98.0%
c_model: val error = 84.0 examples
```

	acc_train	acc_val	batch_size	keep_prob	lambda	layer_dims	lr	min_lr	num_epochs	val_error
0	1.0	0.98	128	0.75	0	[784, 512, 10]	0.001	1.000000e-08	25	84.0