SVD

July 7, 2024

0.1 A. Definisi dan Teori

Singular Value Decompositions (SVD) dari suatu matriks A dengan ukuran $m \times n$ adalah faktorisasi matriks tersebut menjadi tiga matriks berbeda yang dinotasikan dengan U, Σ, V. Pada umumnya matriks A dapat didefinisikan sebagai berikut:

$$A = U\Sigma V^T$$

Bentuk ini tentu tidaklah sepenuhnya berbeda dengan *Eigen Value Decompositions* (EVD) dari sebuah matriks persegi non-singuar A yang didefinisikan sebagai $A = X\Lambda X^{-1}$ atau bahkan sangat mirip dengan EVD dari sebuah matriks simetris real S yang didefinisikan sebagai $S = Q\Lambda Q^T$. Perbedaan paling fundamental pada konsep-konsep di atas hanyalah pada bentuk matriks A dan S. SVD tidaklah memiliki syarat khusus bagaimana bentuk matriks yang akan difaktorisasi, bahkan matriksnya dapat berupa matriks non-persegi.

Telah diketahui bentuk umum matriks yang didefinisikan oleh SVD, selanjutnya perlu diketahui pula apa yang dimaksud dengan tiga buah matriks hasil faktorisasi SVD yaitu U, Σ, V. Matriks V dapat didefinisikan sebagai matriks *orthonormal eigenvector* dari hasil operasi A^TA dengan ukuran $n \times n$, sedangkan matriks Σ adalah akar dari matriks diagonal yang berisi *eigenvalue* dari operasi A^TA dengan ukuran $m \times n$. Adapun matriks U dapat didefiniskan sebagai matriks *orthonormal eigenvector* dari operasi AA^T dengan ukuran $m \times m$, tetapi untuk meminimalisir kesalahan pada matriks U, dapat digunakan $U = AV\Sigma^{-1}$. Berdasarkan penjelasan tersebut, dapat diketahui bahwa matriks U dan V berturut-turut berkorespodensi dengan *columns spaces* serta *left nullspace* dan *rows spaces* serta *nullspace* dari matriks A.

Untuk lebih mempermudah mengetahui hubungan matriks U, Σ, V terhadap matriks A, berikut ini ringkasannya,

- $U \in \mathbb{R}^{m\times m} \implies$ kolom matriks U dari kolom 1 hingga kolom r adalah *columns space* A, sedangkan sisanya adalah *left nullspace* A
- $\Sigma \in \mathbb{R}^{m\times n} \implies$ diagonal matriks Σ merupakan *eigenvalue* A
- $V \in \mathbb{R}^{n\times n} \implies$ kolom matriks V dari kolom 1 hingga kolom r adalah *rows space* A, sedangkan sisanya adalah *nullspace* A

Dengan menyelesaikan definisi SVD yang telah dikemukakan sebelumnya, matriks A dapat direseprentasikan pula dengan kombinasi linear dari komponen-komponen matriks U, Σ, V^T sebanyak r atau ranking dari matriks A. Dengan begitu, matriks A dapat pula dituliskan sebagai berikut,

$$A = u_1\sigma_1 v_1^T + u_2\sigma_2 v_2^T + \cdots + u_r\sigma_r v_r^T$$

0.2 B. Konsep SVD dalam *Image Compression*

Dalam komputer, sebuah gambar sejatinya adalah sebuah matriks dengan ukuran tertentu yang elemen-elemennya mewakili gelap terang suatu warna. Tentu jika gambar adalah sebuah matriks, maka bukan mustahil untuk melakukan operasi-operasi matriks terhadap suatu gambar. Tak terkecuali operasi SVD ini.

Seperti yang dijelaskan pada poin A, SVD akan melakukan faktorisasi suatu matriks menjadi tiga matriks U, Σ, V, dimana U dan V merupakan *orthonormal eigenvector* dan matriks Σ merupakan *eigenvalue*. Dalam kasus matriks gambar A, elemen-elemen dari matriks A sangat dipengaruhi oleh matriks U, Σ, V. Hal itu terbukti dengan reseprentasi matriks A dalam bentuk linear kombinasi dari matriks U, Σ, V^T.

Di sisi lain, dalam konsep EVD, tidak semua *eigenvalue* berdampak signifikan terhadap suatu matriks. Dengan begitu, matriks Σ, yang berisi *eigenvalue* dari matriks A, bisa diatur sedemikian rupa untuk hanya memiliki *eigenvalue* yang signifikan dan itu berarti akan ada komponen-komponen matriks U dan V yang dihilangkan.

Jika menggunakan reseprentasi A sebagai linear kombinasi, maka kita dapat hanya melakukan penambahan hingga komponen ke-n saja dan tak perlu sampai komponen ke-r karena bisa jadi komponen dari $r - n$ tidaklah signifikan. Seperti itulah konsep umum *image compression* menggunakan SVD.

Jadi, pada dasarnya proses yang dilakukan hanyalah mengeliminasi *eigenvector* dan *eigenvalue* dari sebuah matriks gambar yang dianggap tidak signifikan, sehingga gambar yang dihasilkan akan menjadi lebih ringan dan efisien tanpa banyak mengurangi kualitas gambar asli.

0.3 C. Implementasi dan Analisis

Implementasi *image compression* dengan konsep SVD akan dilakukan menggunakan bantuan bahasa pemrograman Python. Adapun langkah-langkahnya adalah sebagai berikut:

0.3.1 Mengimport seluruh library yang dibutuhkan

Pada program ini akan digunakan empat library, yaitu `Pillow`, `numpy`, `pandas`, `matplotlib`, dan `requests`. Library `Pillow` digunakan untuk membaca dan menyimpan gambar ,library `numpy` digunakan untuk operasi matriks, library `pandas` untuk pemrosesan data, library `matplotlib` untuk visualisasi data, dan library `requests` untuk mengambil data dari internet.

[ ]:
```
from PIL import Image
import numpy as np
import requests as rq
import matplotlib.pyplot as plt
```

0.3.2 Mengimport gambar dan menyimpannya ke dalam bentuk array (matriks)

Gambar yang akan digunakan adalah gambar dari website milik UGM. Untuk dapat mengambil gambar dari internet, digunakan library `requests`. Setelah url dari gambar telah diambil, gambar tersebut akan disimpan dalam sebuah variabel bernama `img` dan juga mengubahnya menjadi hitam putih dengan menggunakan library `Pillow`.

Pada program ini, gambar yang diambil akan ditampilkan ke dalam bentuk array (matriks) dengan menggunakan library `numpy` dan disimpan ke dalam variabel atau matriks A.

[ ]:
```
# import gambar
req = rq.get("https://ugm.ac.id/wp-content/uploads/2023/04/About-Hero.jpg",␣
↪stream=True)
img = Image.open(req.raw)
img = img.convert("L")

# menyimpan ke dalam bentuk array
A = np.array(img)
print(f"Bentuk Array dari Gambar:\n\n {A}")
print(f"\nUkuran Array dari Gambar: {np.shape(A)}")
```

```
Bentuk Array dari Gambar:

 [[58 58 57 … 38 38 40]
 [59 59 58 … 32 33 35]
 [58 58 58 … 36 37 38]
 …
 [35 43 57 … 38 39 40]
 [29 37 52 … 38 39 40]
 [25 30 42 … 36 37 38]]

Ukuran Array dari Gambar: (709, 1260)
```

[ ]:
```
# menampilkan gambar
plt.figure(figsize=(10, 10))
plt.imshow(A, cmap='gray')
plt.axis('off')
plt.title('Gambar Asli', fontsize=20)
plt.show()
```

Gambar Asli

0.3.3 Menghitung SVD dari matriks gambar

Seperti yang telah dijelaskan pada poin B, *image compression* dapat diterapkan menggunakan SVD. Untuk itu, gambar yang telah dijadikan array (matriks) akan difaktorisasi secara SVD menggunakan library `numpy` atau lebih tepatnya fungsi `np.linalg.svd` yang akan mengembalikan tiga matriks U, Σ, V^T yang masing masing disimpan dalam variabel U, S, V_t.

```
[ ]: U, S, Vt = np.linalg.svd(A)

# menampilkan ukuran matriks U, S, Vt
print(f"Ukuran matriks U: {np.shape(U)}, ukuran matriks S: {np.shape(S)},␣
 ↪ukuran matriks Vt: {np.shape(Vt)}")
```

```
Ukuran matriks U: (709, 709), ukuran matriks S: (709,), ukuran matriks Vt:
(1260, 1260)
```

Fungsi `np.linalg.svd` didesain hanya akan mengembalikan matriks Σ dengan ukuran $m \times 1$ padahal seharusnya matriks Σ memiliki ukuran yang sama dengan matriks A. Dengan begitu, matriks Σ yang disimpan dalam variabel S perlu untuk dikonstruksi kembali menjadi matriks Σ dengan ukuran yang sama dengan matriks A.

Program dibawah ini akan mengkonstruksi matriks Σ dengan ukuran yang sama dengan matriks A dengan cara membuat matriks 0 seukuran matriks A dan mengisi diagonal matriks tersebut dengan elemen dari matriks Σ sebelumnya. Hasil dari proses itu akan disimpan dalam variabel baru bernama Sigma

[ ]:
```python
# konstruks diagonal Sigma
Sigma = np.zeros(np.shape(A))
np.fill_diagonal(Sigma, S)

# menampilkan ukuran matriks U, Sigma, Vt
print(f"Ukuran matriks U: {np.shape(U)}, ukuran matriks Sigma: {np.
 ↪shape(Sigma)}, ukuran matriks Vt: {np.shape(Vt)}")
```

```
Ukuran matriks U: (709, 709), ukuran matriks Sigma: (709, 1260), ukuran matriks
Vt: (1260, 1260)
```

0.3.4 Mengompresi gambar

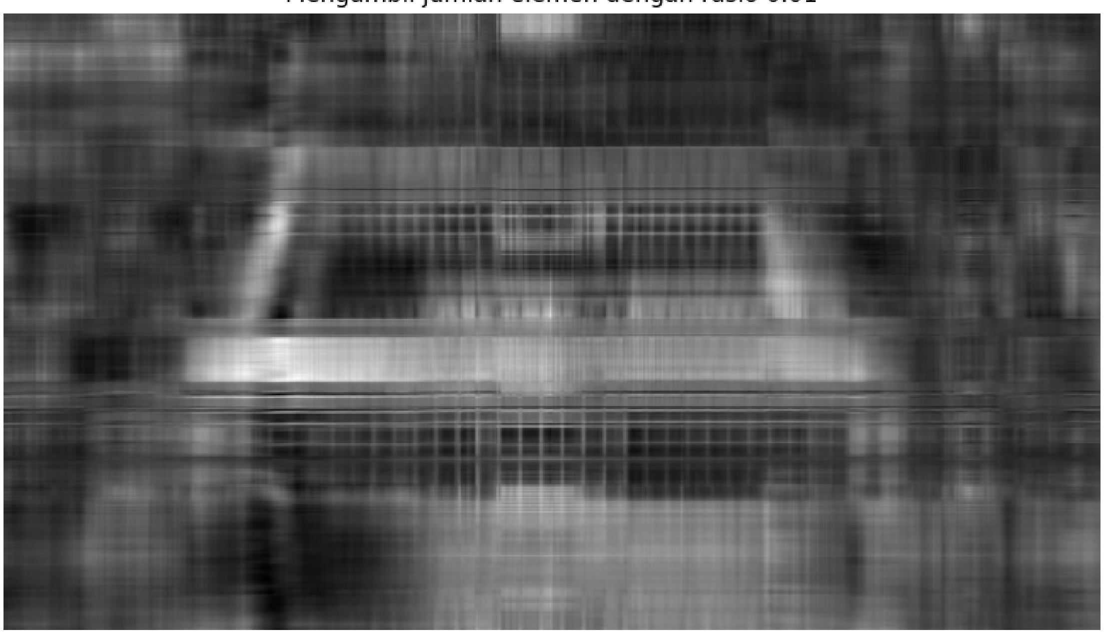
Seperti yang telah dijelaskan pada poin B, bahwa pada dasarnya *image compression* menggunakan SVD akan mengeliminasi *eigenvalue* dan *eigenvector* yang tidak terlalu signifikan. Maka, pada program dibawah ini, elemen-elemen matriks U, Σ, V^T akan diambil dengan rasio yang ditentukan. Dengan kata lain, akan ada elemen yang dihilangkan dan tidak digunakan.

Program dibawah ini akan mengambil elemen-elemen matriks U, Σ, V^T dengan beberapa rasio yang ditentukan dan akan menampilkan hasil gambar dari operasi tersebut. Dengan begitu, kita akan mengetahui sampai mana elemen-elemen matriks gambar yang signifikan.

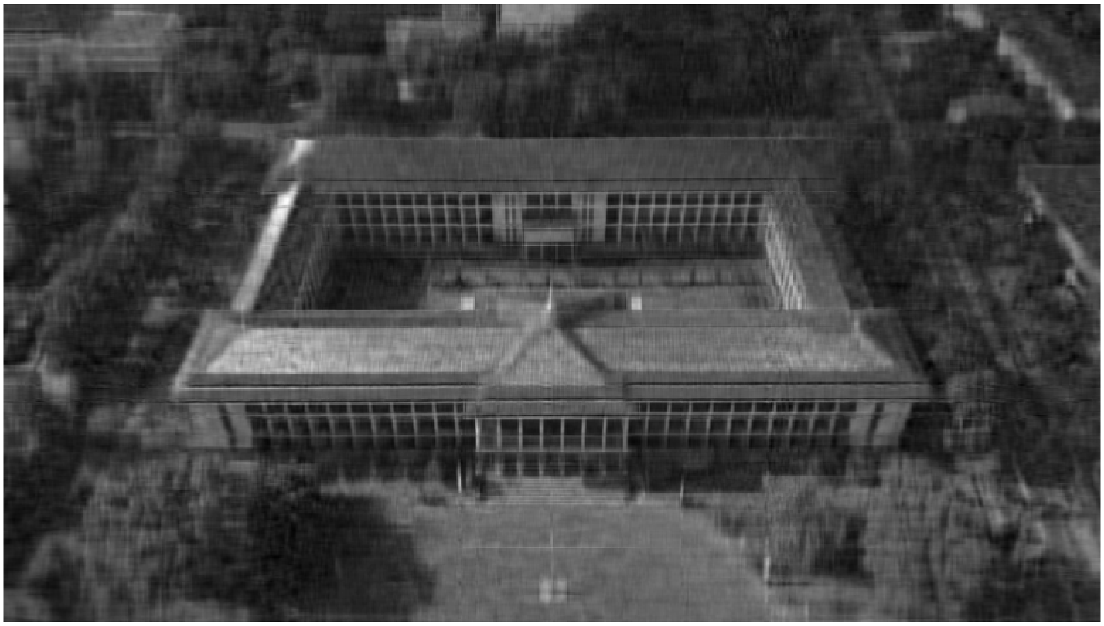
[ ]:
```python
# mengompresi gambar
for ratio in (0, 0.01, 0.05, 0.1, 0.5, 1):

    i=0

    # mengambil elemen dengan rasio yang ditentukan
    compressedA = U[:,:int(np.shape(A)[0]*ratio)] @ Sigma[:int(np.
 ↪shape(A)[0]*ratio),:int(np.shape(A)[1]*ratio)] @ Vt[:int(np.
 ↪shape(A)[1]*ratio),:]

    # menampilkan gambar
    plt.figure(i+1, figsize=(10, 10))
    i += 1
    img = plt.imshow(compressedA)
    img.set_cmap('gray')
    plt.axis('off')
    plt.title('Mengambil jumlah elemen dengan rasio ' + str(ratio))
    plt.show()
```

Mengambil jumlah elemen dengan rasio 0

Mengambil jumlah elemen dengan rasio 0.01



Mengambil jumlah elemen dengan rasio 0.05



Mengambil jumlah elemen dengan rasio 0.1

Mengambil jumlah elemen dengan rasio 0.5

Mengambil jumlah elemen dengan rasio 1