

Laporan Tugas Besar I
IF-2123 Aljabar Linier Dan Geometri
Aplikasi Nilai Eigen dan Vektor Eigen dalam Kompresi
Gambar



Institut Teknologi Bandung
Semester I Tahun 2021/2022

Kelompok 17 - Jamur Crispy

13520002	Muhammad Fikri Ranjabi
13520141	Yoseph Alexander Siregar
13520147	Aloysius Gilang Pramudya

DAFTAR ISI

DAFTAR ISI	2
BAB 1 DESKRIPSI MASALAH.....	3
BAB 2 TEORI SINGKAT	4
Perkalian Matriks	4
Nilai Eigen	4
Vektor Eigen.....	5
Matriks SVD	5
BAB 3 IMPLEMENTASI PROGRAM.....	6
Tech Stack	6
Garis Besar Algoritma Kompresi	6
BAB 4 EKSPERIMEN	9
Hasil Eksekusi Program	9
BAB 5 KESIMPULAN	11
Saran	11
DAFTAR REFERENSI.....	12

BAB 1

DESKRIPSI MASALAH

Akan dibuat sebuah program kompresi gambar dengan memanfaatkan algoritma SVD dalam bentuk website lokal sederhana. Spesifikasi website adalah sebagai berikut:

1. Website mampu menerima file gambar beserta input tingkat kompresi gambar (dibebaskan formatnya).
2. Website mampu menampilkan gambar input, output, runtime algoritma, dan persentase hasil kompresi gambar (perubahan jumlah pixel gambar).
3. File output hasil kompresi dapat diunduh melalui website.
4. Kompresi gambar tetap mempertahankan warna dari gambar asli.
5. (Bonus) Kompresi gambar tetap mempertahankan transparansi dari gambar asli, misal untuk gambar png dengan background transparan.
6. Bahasa pemrograman yang boleh digunakan adalah Python, Javascript, dan Go.
7. Penggunaan framework untuk back end dan front end website dibebaskan. Contoh framework website yang bisa dipakai adalah Flask, Django, React, Vue, dan Svelte.
8. Kalian dapat menambahkan fitur fungsional lain yang menunjang program yang anda buat (unsur kreativitas diperbolehkan/dianjurkan).
9. Program harus modular dan mengandung komentar yang jelas.
10. Diperbolehkan menggunakan library pengolahan citra seperti OpenCV2, PIL, atau image dari Go.
11. Dilarang menggunakan library perhitungan SVD dan library pengolahan eigen yang sudah jadi.

BAB 2

TEORI SINGKAT

Perkalian Matriks

Notasi Matriks

- Matriks berukuran $m \times n$ (m baris dan n kolom):

$$A = [a_{ij}] = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

Jika $m = n$ maka dinamakan matriks persegi (*square matrix*) orde n

- Perkalian dua buah matriks $C_{m \times n} = A_{m \times r} \times B_{r \times n}$

Misal $A = [a_{ij}]$ dan $B = [b_{ij}]$

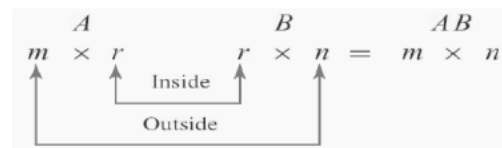
maka $C = A \times B = [c_{ij}]$, $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \dots + a_{in}b_{nj}$

syarat: jumlah kolom A sama dengan jumlah baris B

- Algoritma perkalian dua buah matriks $C_{m \times n} = A_{m \times r} \times B_{r \times n}$

```

for i ← 1 to m do
  for j ← 1 to n do
     $c_{ij} \leftarrow 0$ 
    for k ← 1 to r do
       $c_{ij} \leftarrow c_{ij} + a_{ik} * b_{kj}$ 
    end for
  end for
end for
  
```



Nilai Eigen

- Jika A adalah matriks $n \times n$ maka vektor tidak-nol \mathbf{x} di \mathbb{R}^n disebut **vektor eigen** dari A jika $A\mathbf{x}$ sama dengan perkalian suatu skalar λ dengan \mathbf{x} , yaitu

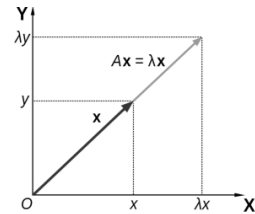
$$A\mathbf{x} = \lambda\mathbf{x}$$

Skalar λ disebut **nilai eigen** dari A , dan \mathbf{x} dinamakan vektor eigen yang berkoresponden dengan λ .

- Kata “eigen” berasal dari Bahasa Jerman yang artinya “asli” atau “karakteristik”.
- Dengan kata lain, nilai eigen menyatakan nilai karakteristik dari sebuah matriks yang berukuran $n \times n$.

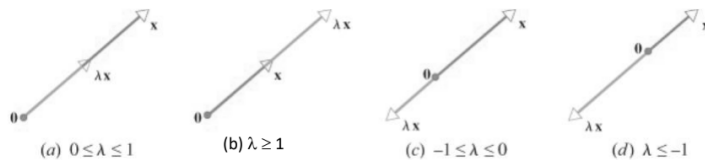
Vektor Eigen

- Vektor eigen \mathbf{x} menyatakan vektor kolom yang apabila dikalikan dengan sebuah matriks $n \times n$ menghasilkan vektor lain yang merupakan kelipatan vektor itu sendiri.



Sumber gambar: Wikipedia

- Dengan kata lain, operasi $A\mathbf{x} = \lambda\mathbf{x}$ menyebabkan vektor \mathbf{x} menyusut atau memanjang dengan faktor λ dengan arah yang sama jika λ positif dan arah berkebalikan jika λ negatif.



4

Matriks SVD

If A is an $m \times n$ matrix of rank k , then A can be factored as

$$A = U\Sigma V^T = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_k & \mathbf{u}_{k+1} & \cdots & \mathbf{u}_m \end{bmatrix} \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_k \\ 0_{(m-k) \times k} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_k^T \\ \mathbf{v}_{k+1}^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

in which U , Σ , and V have sizes $m \times m$, $m \times n$, and $n \times n$, respectively, and in which

- $V = [\mathbf{v}_1 \ \mathbf{v}_2 \ \cdots \ \mathbf{v}_n]$ orthogonally diagonalizes $A^T A$.
- The nonzero diagonal entries of Σ are $\sigma_1 = \sqrt{\lambda_1}, \sigma_2 = \sqrt{\lambda_2}, \dots, \sigma_k = \sqrt{\lambda_k}$, where $\lambda_1, \lambda_2, \dots, \lambda_k$ are the nonzero eigenvalues of $A^T A$ corresponding to the column vectors of V .
- The column vectors of V are ordered so that $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_k > 0$.
- $\mathbf{u}_i = \frac{A\mathbf{v}_i}{\|A\mathbf{v}_i\|} = \frac{1}{\sigma_i} A\mathbf{v}_i \quad (i = 1, 2, \dots, k)$
- $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\}$ is an orthonormal basis for $\text{col}(A)$.
- $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k, \mathbf{u}_{k+1}, \dots, \mathbf{u}_m\}$ is an extension of $\{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k\}$ to an ortho-normal basis for \mathbb{R}^m .

The vectors $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_k$ are called the *left singular vectors* of A , and the vectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_k$ are called the *right singular vectors* of A .

BAB 3

IMPLEMENTASI PROGRAM

Tech Stack

Program dibuat dengan menggunakan Flask sebagai micro web framework yang dihubungkan ke backend dengan bahasa pemrograman python. Flask memanfaatkan sebuah template untuk melakukan render pada plain HTML menjadi sebuah tampilan web pages. Adapun library python yang digunakan yaitu:

- numpy: untuk melakukan operasi matriks, mengubah input gambar menjadi bentuk array dengan komponen RGB
- PIL: untuk membaca input gambar
- math: menghitung operasi dasar matematika

Garis Besar Algoritma Kompresi

Gambar yang akan dikompresi akan diubah menjadi matriks menggunakan library numpy. Matriks yang dihasilkan dari proses ini adalah array 3 dimensi (berlaku untuk semua format kecuali format png karena png akan berbentuk array 2 dimensi).

Pada dimensi ke 3 dari array tersebut, mewakili data dari r, g, dan b dari gambar tersebut pada indeks [0..2] secara berurutan.

Untuk mempermudah proses maka array tersebut dipecah menjadi 3 buah array 2 dimensi dari data pada r, g, dan b.

Pada ketiga array tersebut, dilakukan proses dekomposisi matriks dengan algoritma svd -- yang memanfaatkan metode dekomposisi QR juga -- menjadi 3 buah matriks U, sigma, dan Vtranspose.

Setelah itu dilakukan aproksimasi terhadap matriks U, sigma, dan Vtranspose tersebut (disinilah proses kompresi terjadi) berdasarkan nilai singular value / rank dari matriks (k) yang diinginkan berdasarkan dari input user.

Pada matriks U diambil sub-matriks dengan seluruh baris dan kolom 1-k, pada Sigma data nilai singular sebanyak k, dan pada Vtranspose diambil sub matriks dengan baris 1-k dan seluruh kolom.

Lalu hasil dari aproksimasi pada ketiga matrix tersebut digabungkan kembali dengan perkalian dot matriks yang hasilnya adalah matrix yang merepresentasikan data gambar yang sudah terkompres. Aproksimasi ini dilakukan pada masing-masing array r, g, dan b.

Ketiga array hasil aproksimasi tersebut, lalu digabungkan kembali menjadi sebuah array 3 dimensi.

Array hasil gabungan tersebut adalah representasi dari gambar yang sudah dikompresi.

Array tersebut lalu dikembalikan ke bentuk gambar dengan menggunakan library PIL pada python.

Dalam mencari nilai eigen dan vektor eigen kami menggunakan metode QR. Metode QR adalah metode iteratif untuk menemukan semua nilai eigen (tidak bersamaan dengan vektor eigen). Matriks yang serupa (*similar*) akan memiliki nilai eigen dan vector eigen yang sama jika :

$$A = C^{-1}BC$$

C adalah matriks yang *invertible*.

Metode QR mendekomposisi sebuah matriks menjadi dua matriks yaitu Q dan R, dimana Q adalah matriks ortogonal, dan R adalah matriks segitiga atas. Misalkan kita memiliki sebuah matriks A_0 . Pada step ke- k (mulai dari $k = 0$), kita dapat menggunakan dekomposisi QR untuk mendapatkan :

$$A_k = Q_k R_k$$

adalah matriks ortogonal, dan R_k adalah matriks segitiga atas. lalu kita buat

$$A_{k+1} = R_k Q_k$$

untuk mendapatkan

$$A_{k+1} = R_k Q_k = Q_k^{-1} Q_k R_k Q_k = Q_k^{-1} A_k Q_k$$

karena semua A_k serupa (*similar*) maka mereka mempunyai nilai eigen yang sama

Saat iterasi berlanjut, kita akhirnya akan membentuk matriks segitiga atas dengan bentuk:

$$A_k = R_k Q_k = \begin{bmatrix} \lambda_1 & X & \dots & X \\ 0 & \lambda_2 & \dots & X \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix},$$

yang elemen diagonalnya adalah nilai-nilai eigen eigen dan semua produk hasil kali dari Q berisikan vektor eigen.

Setelah didapat nilai eigen dan vektor eigen dari langkah di atas, berikutnya adalah untuk mencari bentuk SVD dari matriks. Pada program ini kami menggunakan algoritma

pencarian SVD yang berbeda dari yang diajarkan di kelas. Ingat kembali bahwa SVD memiliki bentuk seperti pada gambar berikut.

$$A = U\Sigma V^T = [\mathbf{u}_1 \quad \mathbf{u}_2 \quad \cdots \quad \mathbf{u}_k | \mathbf{u}_{k+1} \quad \cdots \quad \mathbf{u}_m] \left[\begin{array}{cccc|ccc} \sigma_1 & 0 & & \cdots & 0 & & & \\ 0 & \sigma_2 & & \cdots & 0 & & & \\ \vdots & \vdots & & \ddots & \vdots & & & \\ 0 & 0 & & \cdots & \sigma_k & & & \\ & 0_{(m-k) \times k} & & & & & & \end{array} \right] \left[\begin{array}{c} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_k^T \\ \hline \mathbf{v}_{k+1}^T \\ \vdots \\ \mathbf{v}_n^T \end{array} \right]$$

Langkah pertama adalah menghitung nilai dari V . Jika kita lihat, perkalian matriks $A^T A$ jika didekomposisi akan memiliki bentuk

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

kita hitung nilai $V = A^T A$. Kemudian didapat nilai V_i yang merupakan vektor eigen dari $A^T A$. Dari rumus di atas kemudian kita langsung bisa mendapatkan nilai singular karena $(\Sigma^T \Sigma)$ merupakan matriks dengan elemen diagonal yaitu σ_i^2 , dimana $\sigma_i, \dots, \sigma_n$ adalah nilai singular dari A .

Setelah itu kita akan mencari matriks U . Matriks U bisa didapatkan menggunakan rumus

$$U_i = \frac{A v_i}{\sigma_i}$$

dengan nilai i sampai dengan jumlah baris dari U . Matriks U diisi dengan kolom per kolom. Ingat bahwa nanti nilai U yang didapat berkorespondensi dengan nilai V dan σ .

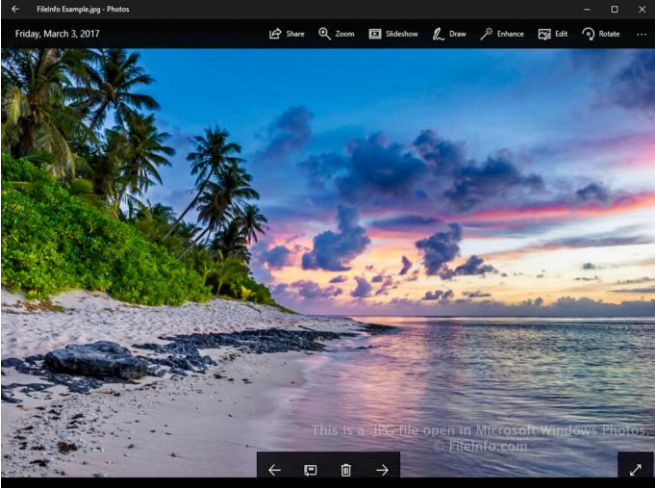
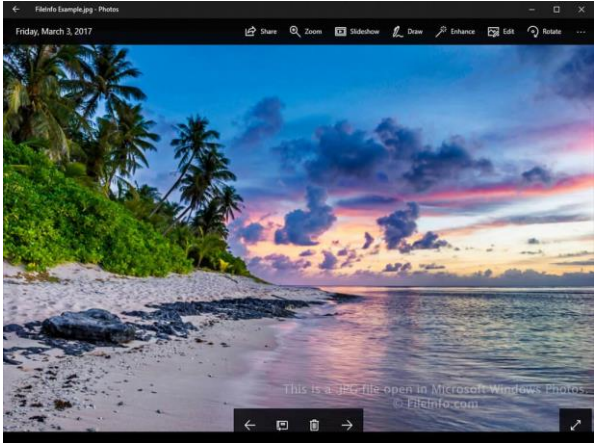



Langkah terakhir adalah menghitung SVD dengan rumus utama





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

Algoritma ini digunakan untuk mencari nilai SVD dari setiap channel gambar (red, green, dan blue). Efisiensi yang dimiliki cukup besar dibandingkan dengan mencari nilai U menggunakan matriks AA^T karena kita hanya perlu mencari sekali nilai eigen dan vektor eigennya.

BAB 4 EKSPERIMEN

Hasil Eksekusi Program

Keterangan Gambar	Persentase Kompresi	Hasil Kompresi
Colorful1.jpg 129kb 1024 x 768 	50	Waktu : 12.285 sekon, 112 kb, 1024 x 768 
Colorful2jpeg.jpeg 128kb 1000 x 667 	60	Waktu : 09.923 sekon, 91kb, 1000 x 667 
Colorful3.jpg 745kb 1920 x 1200 	70	Waktu : 36.134, 113kb, 1920 x 1200 

<div>grayscale679kb.jpg 679kb 1600 x 665</div> <div></div>	<div>80</div>	<div>Waktu : 18.316, 225kb, 1600 x 665</div> <div></div>
<div>Grayscale2.jpg 50kb 640 x 640</div> <div></div>	<div>90</div>	<div>Waktu : 04.631, 36kb, 640 x 640</div> <div></div>

BAB 5

KESIMPULAN

Singular Value Decomposition (SVD) merupakan metode dekomposisi matriks menjadi 3 bagian yang dapat digunakan untuk kompresi ukuran gambar dengan mengambil nilai K tertentu untuk mengambil sebagian komponen penting yang menyimpan nilai pixel dari gambar dalam bentuk matriks. Pada Tugas Besar 2 IF-2123 Aljabar Linier dan Geometri ini, telah dibuat suatu website kompresi gambar dengan persentase kompresi tertentu memanfaatkan perhitungan SVD dari gambar sehingga pada website dapat dihasilkan gambar dalam bentuk terkompresi beserta keterangan durasi dan rasio kompresi gambar.

Saran

Dalam pengerjaan tugas besar ini, kami menyadari dengan sepenuhnya kendala dan keterbatasan pada website kompresi yang kami buat. Tentunya terdapat berbagai hal yang kedepannya dapat dikembangkan untuk optimalisasi lebih lanjut. Untuk itu, kami mengharapkan kritik dan saran dari pihak-pihak yang turut mengevaluasi program dan laporan ini. Terima kasih kepada tim dosen IF-2123 Aljabar Linier Dan Geometri bersama tim asisten atas segala bimbingannya.

DAFTAR REFERENSI

“Nilai Eigen dan Vektor Eigen Bagian 1” by Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-18- Nilai-Eigen-dan-Vektor-Eigen-Bagian1.pdf>

“Singular Value Decomposition (SVD)” by Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-19b-Singular-value-decomposition.pdf>

“Review Matriks” by Rinaldi Munir

<https://informatika.stei.itb.ac.id/~rinaldi.munir/AljabarGeometri/2020-2021/Algeo-01-Review-Matriks.pdf>

“Image Compression using Singular Value Decomposition (SVD)”

http://www.math.utah.edu/~goller/F15_M2270/BradyMathews_SVDImage.pdf

“Understanding Singular Value Decomposition and its Application in Data Science” by Reza Bagheri

<https://towardsdatascience.com/understanding-singular-value-decomposition-and-itsapplication-in-data-science-388a54be95d>

“The Singular Value Decomposition”

https://math.mit.edu/~gs/linearalgebra/linearalgebra5_7-1.pdf

“Image Compression with Singular Value Decomposition”

<http://timbaumann.info/svd-image-compression-demo/>

“Developing Web Applications with Flask”

https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/Python3_Flask.html

“Singular Value Decomposition (SVD)” by MIT OpenCourseWare

<https://www.youtube.com/watch?v=rYz83XPxiZo>

“Lecture 6: Singular Value Decomposition (SVD) Transcript Notes”

<https://ocw.mit.edu/courses/mathematics/18-065-matrix-methods-in-data-analysis-signal-processing-and-machine-learning-spring-2018/video-lectures/lecture-6-singular-value-decomposition-svd/>

“Youtube Playlist Learning Flask”

https://www.youtube.com/playlist?list=PLF2JzgCW6-YY_TZCmBrbOpgx5pSNBD0_L

“ML Wiki: Power Iteration”

http://mlwiki.org/index.php/Power_Iteration