Konvolusi pada Citra yang Dienkripsi Menggunakan Enkripsi Homomorfik CKKS

Jevant Jedidia Augustine / 13520133
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (gmail): 13520133@std.stei.itb.ac.id

***Abstrak*—Skema enkripsi homomorfik total memungkinkan dilakukannya operasi perkalian dan penjumlahan terhadap *ciphertext*. CKKS merupakan skema enkripsi homomorfik totalyang *state-of-the-art* akibat kemampuannya untuk beroperasi pada bilangan kompleks. Salah satu aplikasi dari skema enkripsi homomorfik total adalah enkripsi citra dan proses konvolusi pada citra terenkripsi tersebut. Proses konvolusi mencakup operasi perkalian dan penjumlahan dan skema enkripsi homomorfik total seperti CKKS mendukung kedua operasi tersebut pada *ciphertext*. Pada makalah ini, diusulkan 2 buah algoritma untuk melakukan hal tersebut, algoritma *naïve* dan algoritma *naïve* yang dimodifikasi. Hasil yang didapat menunjukkan bahwa kedua algoritma berhasil melakukan konvolusi terhadap citra terenkripsi dengan hasil yang serupa dengan citra yang tidak dienkripsi. Dari percobaan yang dilakukan, algoritma yang termodifikasi memakan waktu yang lebih sedikit bila dibandingkan denga algoritma *naïve*.**

***Kata kunci*—konvolusi, CKKS, enkripsi homomorfik**

I. Pendahuluan

Enkripsi homomorfik merupakan sebuah skema enkripsi di dalam bidang kriptografi yang memungkinkan dilakukannya komputasi terhadap *ciphertext*. Berdasarkan jumlah operasi yang dapat dilakukan, skema enkripsi homomorfik dibagi menjadi enkripsi homomorfik parsial dan enkripsi homomorfik total. Enkripsi homomorfik parsial hanya mendukung salah satu antara operasi perkalian atau penjumlahan, sedangkan enkripsi homomorfik total mendukung operasi penjumlahan dan perkalian. Dengan enkripsi homomorfik, data sensitif dapat diberikan kepada pihak ketiga untuk dioperasikan dan diolah tanpa harus mengorbankan privasi dan keamanan dari data tersebut.

Salah satu penerapan enkripsi homomorfik adalah pada konvolusi citra. Sebelum ditemukannya skema enkripsi homomorfik total, proses konvolusi pada citra terenkripsi tidak dapat dilakukan karena proses konvolusi melibatkan operasi penjumlahan dan perkalian. Skema enkripsi homomorfik parsial seperti RSA dan ElGammal hanya mendukung salah satu operasi tersebut pada *ciphertext*. Dengan enkripsi homomorfik total, seperti skema CKKS, yang mendukung operasi perkalian dan penjumlahan pada *ciphertext*, proses konvolusi dapat sepenuhnya dilakukan terhadap citra yang dienkripsi.

Kelemahan dari skema enkripsi homomorfik adalah lamanya waktu yang diperlukan untuk menjalankan proses enkripsi, penjumlahan, perkalian, dan dekripsi. Beberapa riset telah dilakukan untuk mempercepat proses-proses tersebut. Pada makalah ini, akan dibahas pengimplementasian enkripsi dan konvolusi pada citra secara *naïve* serta implementasi termodifikasi yang memanfaatkan kemampuan skema enkripsi homomorfik CKKS untuk mengenkripsi beberapa bilangan secara sekaligus untuk mempercepat proses konvolusi citra terenkripsi.

II. Landasan Teori

A. Konvolusi

Dalam pemrosesan citra, konvolusi merupakan operator khusus yang digunakan untuk menapis citra dengan tujuan mentransformasikan sebuah citra menjadi citra yang diinginkan. Konvolusi mengubah sebuah pixel pada citra berdasarkan pixel yang bertetanggaan, sehingga proses konvolusi merupakan operasi pengolahan citra dalam aras lokal. Pada fungsi diskrit, konvolusi didefinisikan sebagai berikut:

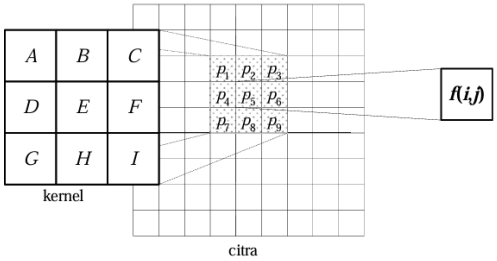
$$h(x) = f(x) * g(x) = \sum_{a=-\infty}^{\infty} f(a)g(x-a)$$

Pada citra, yang merupakan sinyal dwimantra, operasi konvolusi didefinisikan sebagai berikut:

$$h(x,y) = f(x,y) * g(x,y) = \sum_{a=-\infty}^{\infty} \sum_{b=-\infty}^{\infty} f(a,b)g(x-a,y-b)$$

Pada persamaan berikut, $g(x,y)$ dinamakan *kernel* atau *convolution mask*.

Dalam ranah diskrit, *kernel* konvolusi dinyatakan dalam bentuk matriks. Ukuran dari *kernel* bebas, akan tetapi 3x3 merupakan ukuran yang sering digunakan.



Gambar II.1 Ilustrasi Konvolusi (R. Munir, 2023)

Operasi konvolusi dapat dipandang sebagai proses perkalian dot antara vektor pixel dan vektor kernel. Mengacu pada Gambar II.1, proses konvolusi pada sebuah pixel *p* didefinisikan dengan persamaan berikut:

$$p = f(i,j) = (Ap_1 + Bp_2 + Cp_3 + Dp_4 + Ep_5 + Fp_6 + Gp_7 + Hp_8 + Ip_9)$$

Apabila jumlah dari nilai di kernel melebihi 1, maka *p* dibagi dengan jumlah tersebut. Bila tidak, *p* dibagi dengan 1 atau tidak dilakukan apa-apa.

Proses konvolusi menyeluruh pada sebuah citra biasanya dimulai dari pixel ujung kiri atas kemudian menggeser *region of interest* (ROI) ke kanan sebanyak 1 pixel. Apabila sampai di ujung citra, maka ROI akan bergeser ke ujung kiri dan bergeser ke bawah sebanyak 1 pixel. Sesuai dengan prosedur tersebut, tepi dari citra tidak akan terkonvolusi yang menyebabkan citra hasil konvolusi menjadi lebih kecil dari citra awal. Solusi dari permasalahan ini adalah:

1. Mengabaikan pixel pinggir, sehingga citra terkonvolusi memiliki nilai pixel pinggir yang sama dengan citra awal.
2. Menambahkan *padding* pada tepi citra dengan nilai 0 sebelum dilakukan konvolusi. Jumlah *padding* pada kolom dan baris yang diperlukan bergantung dengan ukuran *kernel*.
3. Menduplikasikan nilai tepi pada citra sebagai *padding* sebelum dilakukan konvolusi. Jumlah *padding* pada kolom dan baris yang diperlukan bergantung dengan ukuran *kernel*.

Berbagai proses pengolahan citra menggunakan konvolusi untuk menjalankan prosesnya. Contoh dari proses pengolahan citra tersebut antara lain: penghilangan derau, mengurangi erotan, penghalusan/pelembutan citra, dan deteksi tepi.

B. Skema Enkripsi Homomorfik

Dalam aljabar abstrak, bila terdapat dua grup, $(G,\diamond)$ dan $(H,\circ)$, sebuah homomorfisme grup dari $(G,\diamond)$ ke $(H,\circ)$ adalah sebuah fungsi $f = G \rightarrow H$ dimana untuk semua *g* dan *g'* di *G* memenuhi:

$$f(g \diamond g') = f(g) \circ f(g')$$

Berdasarkan definisi tersebut, sebuah skema enkripsi homomorfik merupakan sebuah skema enkripsi *E* dengan kunci *k* dimana untuk semua *a* dan *b*, memenuhi:

$$Ek(a) \circ Ek(b) = Ek(a \diamond b)$$

Persamaan tersebut mengatakan bahwa skema enkripsi homomorfik memungkinkannya dilakukan operasi penjumlahan dan/atau perkalian terhadap *ciphertext* yang bila didekripsi hasilnya akan memberikan hasil penjumlahan dan/atau perkalian dari *plaintext* yang bersesuaian dengan *ciphertext* tersebut.

Berdasarkan jumlah operasi yang dapat dilakukan pada *ciphertext*, enkripsi homomorfik dibagi menjadi 2 macam:

1. Enkripsi homomorfik parsial

   Memungkinkan dilakukan satu operasi saja terhadap *ciphertext*, penjumlahan atau pengurangan. Contoh algoritma enkripsi homomorfik parsial adalah RSA, ElGamal, dan Pallier.

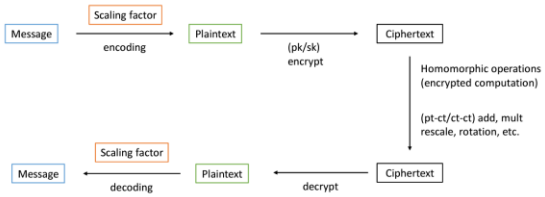
2. Enkripsi homomorfik total

   Memungkingkan dilakukan operasi penjumlahan dan perkalian terhadap *ciphertext*. Contoh algoritma enkripsi homomorfik total adalah BGV, BFV, dan CKKS.

C. CKKS

Skema Cheon-Kim-Kim-Song (CKKS) merupakan skema enkripsi homomorfik total yang pertama kali dikemukakan pada tahun 2017. CKKS merupakan skema enkripsi yang menggunakan aritmatik aproksimasi yang berarti hasil dekripsi dari sebuah *ciphertext* tidak akan menghasilkan pesan yang sama persis akan tetapi memiliki sebuah *eror*. *Eror* ini penting untuk menjaga keamanan skema CKKS dan apabila proses enkripsi dan dekripsi dilakukan dengan baik, nilai *eror* yang dihasilkan tidak 2signifikan sehingga dapat dihiraukan.

Skema CKKS terdiri atas beberapa tahap, yaitu: *encoding*, *encryption*, *ciphertext operation*, *decryption*, dan *decoding*.



Gambar II.2 Skema CKKS

Sumber: https://yongsoosong.github.io/files/slides/intro_to_CKKS.pdf

CKKS menerima sebuah vektor yang berisikan bilangan kompleks sebagai masukan. Vektor ini kemudian di-*encode* untuk memetakan vektor menjadi polinomial CKKS berdasarkan parameter yang ditentukan. Polinomial ini kemudian dienkripsi menjadi *ciphertext* yang dapat dioperasikan baik dengan *ciphertext* yang lainnya maupun sebuah *plaintext*. Untuk mendapatkan pesan asli, *ciphertext* didekripsi dan di-*decode* untuk memetakan polinomial CKKS ke vektor bilangan yang merupakan pesan asli.

Skema CKKS mendukung 3 operasi pada *ciphertext*, yaitu:

1. Penjumlahan

   Sebuah *ciphertext* dapat dijumlahkan dengan *ciphertext* yang lain, atau dengan sebuah *plaintext*.

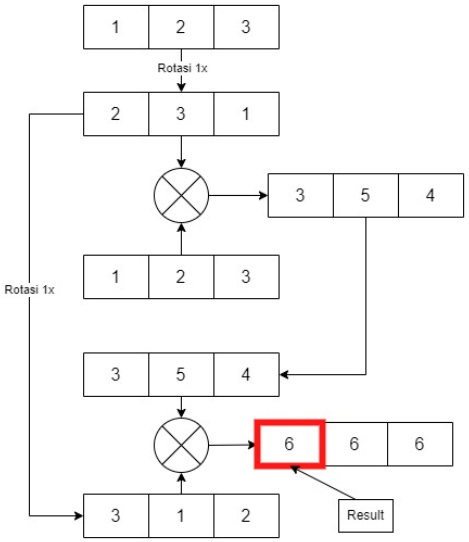
Untuk $pt1 = [1,2,3,4]$ dan $pt2 = [2,3,4,5]$, operasi $c(pt1) + c(pt2)$ dan $c(pt1) + pt2$ akan menghasilkan penjumlahan kedua vektor berdasarkan elemennya, yaitu $res = [3,5,7,9]$.

2. Perkalian

   Sebuah *ciphertext* dapat dikali dengan *ciphertext* yang lain, atau dengan sebuah *plaintext*. Untuk $pt1 = [1,2,3,4]$ dan $pt2 = [2,3,4,5]$, operasi $c(pt1) * c(pt2)$ dan $c(pt1) * pt2$ menghasilkan perkalian kedua vektor berdasarkan elemennya, yaitu $res = [2,6,12,20]$.

3. Rotasi

   Untuk $pt1 = [1,2,3]$, operasi rotasi terhadap *pt1* sebanyak 2 kali akan menghasilkan $res = [3,1,2]$. Proses rotasi dapat digunakan untuk menjumlahkan semua elemen yang ada pada sebuah vektor.

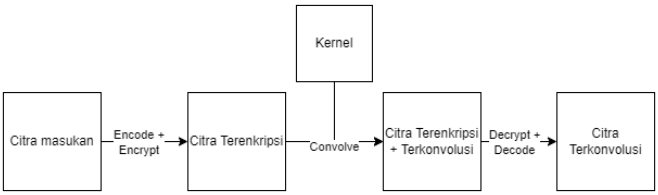


Gambar II.3 Ilustrasi Penjumlahan Setiap Elemen pada Vektor Menggunakan Operasi Rotasi

Skema CKKS merupakan skema enkripsi berlevel yang berarti terdapat batas untuk proses perkalian yang dapat dilakukan terhadap *ciphertext*. Apabila level ini habis, maka proses yang dapat dilakukan hanyalah dekripsi, rotasi, atau penjumlahan. Apabila dilakukan proses perkalian lebih lanjut lagi terhadap *ciphertext*, hasil dekripsi dari *ciphertext* tidak akan menghasilkan pesan yang benar, sehingga perlu diperhatikan jumlah level dan jumlah perkalian yang akan dilakukan terhadap *ciphertext.*

III. Implementasi

Diusulkan 2 algoritma enkripsi serta konvolusi terhadap sebuah citra masukan, yaitu *naïve* dan termodifikasi. Kedua metode memiliki alur yang serupa, seperti yang ditampilkan pada Gambar III.1. Perbedaan dari kedua metode terdapat pada tahap enkripsi dan konvolusi yang akan dijelaskan lebih dalam pada bagian selanjutnya.

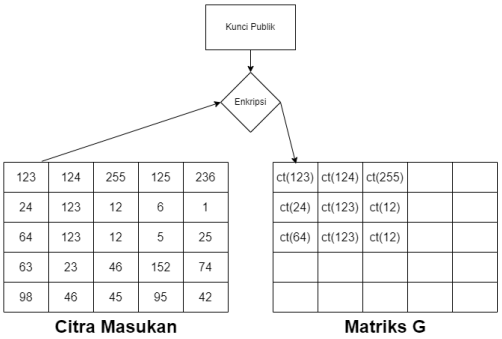


Gambar III.1 Alur Umum Konvolusi pada Citra Terenkripsi

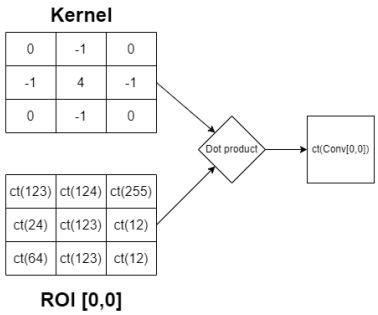
A. Algoritma *Naïve*

Secara umum, algoritma *naïve* mengenkripsi setiap pixel pada citra kemudian melakukan konvolusi tradisional pada setiap pixel terenkripsi dan mendekripsi hasil konvolusi untuk mendapatkan citra hasil. Tahap-tahap detil dari implementasi konvolusi pada citra terenkripsi secara *naïve* adalah sebagai berikut:

1) Definisikan sebuah skema CKKS dengan parameter yang sesuai, kemudian bangun kunci publik dan kunci privat yang dibutuhkan untuk proses enkripsi dan dekripsi.
2) Inisialisasi sebuah matriks G dan H dengan dimensi yang sama dengan citra masukan sebagai citra terenkripsi.
3) Tambahkan *padding* pada citra masukan (opsional).
4) Untuk setiap pixel [i][j] pada citra masukan, *encode* dan enkripsi nilai dari pixel tersebut dengan kunci publik yang dibangun dan masukkan hasil enkripsi ke matriks G pada indeks [i][j].
5) Lakukan proses konvolusi terhadap matriks G dengan menggunakan *kernel* yang telah ditetapkan.
6) Hasil konvolusi untuk pixel [i][j] dimasukkan ke matriks H pada indeks [i][j].
7) Untuk setiap elemen pada matriks H, lakukan dekripsi dan *decoding* terhadap *ciphertext* dengan kunci privat.
8) Pembulatan serta normalisasi nilai pixel pada matriks H akan menghasilkan citra terkonvolusi.



Gambar III.2 Enkripsi Citra pada Algoritma *Naive*

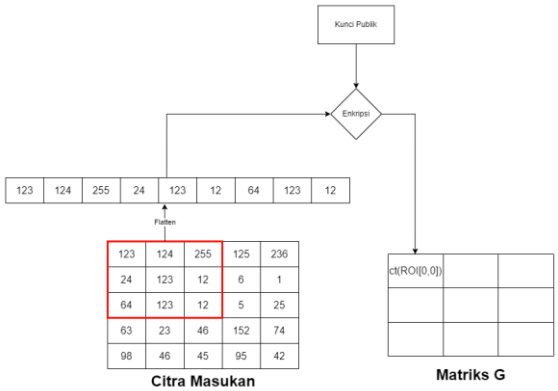


Gambar III.3 Konvolusi Citra Terenkripsi pada Algoritma *Naïve*

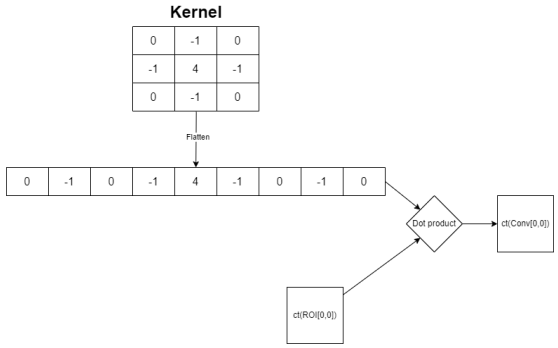
B. Algoritma Termodifikasi

Algoritma termodifikasi memanfaatkan fakta bahwa operasi konvolusi merupakan perkalian produk antara *kernel* dan ROI pada citra serta memanfaatkan kemampuan CKKS untuk mengenkripsi sebuah vektor atau *array* yang berisikan bilangan kompleks & mengalikan vektor tersebut secara sekaligus, sehingga perbedaan metode ini dengan metode *naïve* terdapat pada proses enkripsi dan konvolusi. Tahap-tahap detil dari implementasi konvolusi pada citra terenkripsi yang dimodifikasi adalah sebagai berikut:

1) Definisikan sebuah skema CKKS dengan parameter yang sesuai, kemudian bangun kunci publik dan kunci privat yang dibutuhkan untuk proses enkripsi dan dekripsi.
2) Inisialisasi sebuah matriks G dan H dengan dimensi yang sama dengan citra masukan sebagai citra terenkripsi.
3) Tambahkan *padding* pada citra masukan (opsional).
4) Untuk setiap pixel [i][j] pada citra masukan yang bukan *padding* (atau pixel tepi yang tidak terkonvolusi), ambil ROI konvolusi dari pixel tersebut dan ratakan menjadi vektor atau *array* 1D.
5) *Encode* dan enkripsi vektor tersebut dengan kunci publik dan masukkan hasil enkripsi ke matriks G pada indeks [i][j].
6) Untuk setiap elemen pada matriks G, lakukan perkalian dot dengan *kernel* yang telah diratakan menjadi *array* 1D.
7) Masukkan hasil yang didapat pada langkah 6 ke matriks H pada indeks[i][j].
8) Untuk setiap elemen pada matriks H, lakukan dekripsi dan *decoding* terhadap *ciphertext* dengan kunci privat.
9) Pembulatan serta normalisasi nilai pixel pada matriks H akan menghasilkan citra terkonvolusi.



Gambar III.4 Enkripsi Citra pada Algoritma Termodifikasi



Gambar III.4 Konvolusi Citra Terenkripsi pada Algoritma Termodifikasi

Modifikasi pada implementasi *naïve* diperkirakan akan melakukan konvolusi terhadap citra dalam waktu yang lebih cepat. Hal tersebut didasari oleh proses perkalian produk yang dilakukan dalam proses konvolusi. Pada metode *naïve*, untuk mengonvolusi sebuah pixel, perlu dilakukan iterasi pada setiap elemen ROI pada pixel tersebut untuk dilakukan perkalian dengan elemen pada *kernel* yang kemudian akan dijumlahkan hasilnya, sehingga proses konvolusi untuk sebuah pixel pada metode *naïve* memiliki kompleksitas O(2n).

Pada metode yang termodifikasi, ROI yang diratakan dapat langsung dikalikan dengan *kernel* yang diratakan. Proses perkalian ini tidak bergantung dengan ukuran *kernel*, sehingga proses perkalian memiliki kompleksitas O(1). Hasil perkalian kemudian dapat dijumlahkan dengan operasi rotasi untuk menghasilkan pixel terkonvolusi dengan kompleksitas O(n), sehingga proses konvolusi pada algoritma yang termodifikasi memiliki kompleksitas O(n). Pada beberapa *library* CKKS, pengimplementasian dari kedua operasi ini disederhanakan dengan menggabungkan kedua operasi ke sebuah fungsi perkalian produk.

Implementasi dari kedua algoritma dilakukan dalam Python dengan menggunakan bantuan *library* TenSEAL dan Google Colab. Hasil implementasi dapat dilihat pada pranala https://github.com/JevantJedidia/Convolution-on-Encrypted-Image.

IV. HASIL DAN ANALISIS

Percobaan kedua algoritma akan dilakukan dengan menggunakan citra lena *grayscale* dengan ukuran 100x100 dan sebuah *kernel* laplace berukuran 3x3 dengan isi [-1 -1 -1; -1 8 -1; -1 -1 -1].

Gambar IV.1 Citra Lena

Sebagai tolak ukur, citra lena dikonvolusi secara langsung dengan menggunakan program Matlab.

Gambar IV.2 Citra Lena Terkonvolusi

Perbandingan antara hasil konvolusi citra tolak ukur dan citra terenkripsi dilakukan secara kuantitatif dengan menggunakan *peak signal-to-noise ratio* (PSNR) dan secara kualitatif dengan observasi kasat mata. PSNR dihitung dengan menggunakan persamaan

$$PSNR = 20 * log_{10}(\frac{b}{rms})$$

dengan *b* sebagai nilai sinyal terbesar (b=255 untuk citra dengan 256 derajat keabuan), dan *rms* sebagai akar pangkat dua selisih dari antara citra tolak ukur dan citra yang diobservasi. *rms* dihitung dengan menggunakan persamaan berikut.

$$rms = \sqrt{\frac{1}{N * M}\sum_{i=1}^{N}\sum_{j=1}^{M}(f_{ij} - f'_{ij})^2}$$

Nilai PSNR yang diatas atau sama dengan 30 menyatakan kualitas citra yang baik, sedangkan untuk nilai PSNR dibawah 30 menyatakan bahwa citra mengalami degradasi yang semakin besar dengan menurunnya PSNR.

Pengujian dilakukan dengan menjalankan masing-masing algoritma sebanyak 3 kali kemudian mencatat waktu yang diperlukan untuk menjalankan algoritma serta nilai PSNR dari citra yang dihasilkan. Ketiga nilai tersebut kemudian dirata-ratakan sebagai metrik yang digunakan untuk membandingkan performa setiap algoritma. Berikut hasil yang didapat dari pengujian kedua algoritma dengan menggunakan citra lena yang sama:

Tabel IV.1 Hasil Algoritma Naive

Citra Hasil	Waktu (s)	PSNR (dB)
	330.016	31.88
	339.909	31.88
	339.99	31.88
Rata-rata	336.638	31.88

Tabel IV.2 Hasil Algoritma Termodifikasi

Citra Hasil	Waktu (s)	PSNR (dB)
	257.38	31.871
	225.17	31.88
	232.558	31.8456
Rata-rata	238.369	31.87

Tabel IV.3 Perbandingan Metode *Naïve* dan Termodifikasi

Metode	Waktu (s)	PSNR (dB)
Naive	336.638	31.88
Termodifikasi	238.369	31.87

Berdasarkan hasil yang didapatkan, proses konvolusi pada citra terenkripsi menghasilkan citra yang serupa dengan hasil konvolusi terhadap citra yang tidak dienkripsi. Secara kualitatif, citra yang dihasilkan tidak bisa dibedakan dengan penglihatan kasat mata dan secara kuantitatif, kedua metode menghasilkan nilai PSNR yang serupa dan nilai PSNR yang didapat berada di atas 30 dB sehingga dapat dibilang citra yang dihasilkan memiliki kualitas yang baik. Antara 2 algoritma yang digunakan, hasil yang didapat sesuai dengan hipotesa yang menyatakan bahwa algoritma yang termodifikasi memakan waktu yang lebih sedikit dibandingkan dengan algoritma *naïve*.

Permasalahan utama yang ditemui pada kedua algoritma adalah lamanya waktu yang diperlukan untuk melakukan konvolusi. Untuk citra *grayscale* dengan ukuran 100x100, algoritma terbaik membutuhkan kira-kira 238.369 detik atau sekitar 4 menit untuk dikonvolusi. Pada citra RGB dengan ukuran yang jauh lebih besar, sekitar 1000x1000, waktu yang dibutuhkan dapat mencapai kira-kira 20 jam. Hal ini tentunya tidak *feasible* maupun praktis bila diaplikasikan dalam dunia nyata. Hal utama yang dapat dilakukan untuk mempercepat proses tersebut adalah paralelisasi menggunakan GPU. Algoritma yang diusulkan memiliki sifat paralelisme yang tinggi, seperti pada tahap konvolusi, sehingga paralelisasi program dapat dengan mudah dilakukan untuk mendapatkan program yang berjalan jauh lebih cepat.

V. KESIMPULAN

Diusulkan 2 algoritma untuk melakukan konvolusi terhadap citra yang dienkripsi menggunakan skema enkripsi homomorfik CKKS. Algoritma *naïve* mengenkripsi setiap pixel pada citra kemudian melakukan konvolusi dengan cara yang tradisional terhadap citra yang terenkripsi, sedangkan algoritma yang termodifikasi mengambil ROI dari setiap pixel, mengubahnya menjadi sebuah *array* 1D, dan mengenkripsi *array* tersebut sehingga proses konvolusi hanya mencakup perkalian dot antara *kernel* dan *array* ROI. Dari segi kualitas dan kuantitas, kedua algoritma berhasil memberikan citra yang serupa dengan hasil konvolusi terhadap citra yang tidak dienkripsi. Dari segi waktu, algoritma yang termodifikasi memakan waktu yang lebih sedikit bila dibandingkan dengan algoritma *naïve.*

LINK VIDEO

https://youtu.be/rG893dqk_pg

UCAPAN TERIMA KASIH

Penulis mengucapkan terima kasih kepada Tuhan Yang Maha Esa karena atas berkat dan karunia-Nya lah penulis dapat menyelesaikan makalah "Konvolusi pada Citra yang Dienkripsi Menggunakan Enkripsi Homomorfik CKKS" dengan tepat waktu, Penulis juga mengucapkan terima kasih kepada oran tua penulis yang selalu mendukung penulis selama masa perkuliahan penulis di ITB. Terakhir, penulis ingin mengucapkan terima kasih kepada bapak Rinaldi Munir yang telah membimbing penulis selama satu semester ini dalam menjalankan kelas IF4073 Interpretrasi dan Pengolahan Citra.

REFERENSI

[1] Munir, Rinaldi. (2023). *Bahan Kuliah IF4073 Interpretasi dan Pengolahan Citra.* Program Studi Informatika ITB.

[2] Yi, X., Paulet, R., & Bertino, E. (2014). "Homomorphic Encryption and Applications". Spinger.

[3] Cheon, J. H., Kim, A., Kim, M., & Song, Y. (2017). Homomorphic Encryption for Arithmetic of Approximate Numbers. Advances in Cryptology – ASIACRYPT 2017, 409-437.

[4] https://sefiks.com/2023/04/10/a-step-by-step-fully-homomorphic-encryption-example-with-tenseal-in-python/

[5] https://github.com/OpenMined/TenSEAL

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Desember 2023

Jevant Jedidia 13520133