

**ANALISIS TEKS DAN DOKUMEN: MENGGUNAKAN OCR  
(*OPTICAL CHARACTER RECOGNITION*) UNTUK  
MENGKONVERSI TEKS DALAM GAMBAR MENJADI TEKS  
YANG DAPAT DI EDIT**

Disusun sebagai Salah Satu Syarat untuk Memenuhi Projek Ulangan Akhir  
Semester Mata Kuliah Pengolahan Citra Digital



oleh:

Dwi Miftahussalamah      2110631170010

Nana Casmana Ade Wikarta      2110631170127

**PROGRAM STUDI INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS SINGAPERBANGSA KARAWANG  
KARAWANG  
2024**

## DAFTAR ISI

DAFTAR ISI .....	i
DAFTAR GAMBAR .....	ii
DAFTAR TABEL.....	iii
1. Latar Belakang.....	1
2. Tujuan .....	2
3. Manfaat Penelitian .....	2
3.1. Manfaat Teoritis.....	2
3.2. Manfaat Praktik.....	3
4. Metodologi.....	3
4.1. <i>Optical Character Recognition</i> (OCR).....	3
4.2. Peralatan dan Perangkat Lunak.....	5
5. Hasil dan Pembahasan .....	6
5.1. Pengujian <i>Pre-Processing</i> .....	6
5.2. Pengujian Ekstraksi Teks.....	6
5.3. Hasil Pengujian ke Teks .....	8
6. Kesimpulan dan Saran .....	8
DAFTAR PUSTAKA .....	9
LAMPIRAN .....	10

## DAFTAR GAMBAR

<b>Gambar 1</b> Tahapan OCR.....	4
<b>Gambar 2</b> Tahap <i>Otsu Thresholding</i> .....	6
<b>Gambar 3</b> <i>Zoning</i> .....	7
<b>Gambar 4</b> <i>Tokenization</i> .....	7
<b>Gambar 5</b> Output.....	8

## DAFTAR TABEL

<b>Tabel 1</b> <i>Pattern Recognition</i> .....	7
---	---

## 1. Latar Belakang

Pengolahan citra digital adalah disiplin ilmu yang mempelajari teknik dan metode untuk mengolah gambar diam (foto) dan gambar bergerak (video) yang dihasilkan oleh perangkat akuisisi citra seperti kamera digital, *webcam*, atau *smartphone*, menggunakan pemrosesan digital dengan computer (Ibnutama et al., 2019). Salah satu teknologi yang sangat penting dalam pengolahan citra adalah *Optical Character Recognition* (OCR). OCR memungkinkan konversi teks dalam gambar menjadi teks yang dapat diedit, yang sangat berguna dalam banyak konteks, seperti digitalisasi dokumen, pemrosesan data, dan pengarsipan elektronik.

Proses digitalisasi dokumen telah banyak membantu dalam hal pengarsipan, mulai dari dokumen berupa buku, manuskrip kuno, hingga surat menyurat atau administrasi. Digitalisasi dokumen dapat menghasilkan efisiensi dalam hal tempat penyimpanan dan pengorganisasiannya. Selain itu, dokumen yang diarsipkan secara digital akan lebih awet karena tidak tergantung pada bentuk fisik yang rawan rusak akibat bencana seperti banjir atau kebakaran, serta meminimalisir risiko kehilangan karena proses backup dapat dengan mudah dilakukan.

Namun, dokumen yang telah disimpan dalam bentuk digital seringkali berbentuk file gambar hasil proses pemindaian. Pencarian terhadap dokumen tersebut hanya bisa dilakukan melalui metadata yang di inputkan sebelumnya, seperti judul dokumen, tahun pembuatan, penulis, dan sebagainya. Pencarian terhadap isi dokumen itu sendiri tidak bisa dilakukan secara langsung karena format dokumen yang berupa gambar (Rismanto et al., 2020).

Di sinilah peran OCR menjadi sangat penting. OCR dapat mengekstrak teks dari gambar menjadi teks yang dapat disimpan dan diolah lebih lanjut oleh program komputer. Teknologi ini dapat mempercepat proses digitalisasi dengan tingkat akurasi yang tinggi, sehingga menghemat waktu dan sumber daya. Namun, performa OCR dapat bervariasi tergantung pada kualitas gambar, jenis font, ukuran teks, dan berbagai faktor lainnya. Oleh karena itu, penelitian ini

bertujuan untuk menganalisis efektivitas OCR dalam mengkonversi teks dari gambar menjadi teks yang dapat diedit, serta mengidentifikasi faktor-faktor yang mempengaruhi akurasi dan efisiensi teknologi ini. Dengan pemahaman yang lebih baik tentang bagaimana OCR bekerja dan tantangan yang dihadapinya, diharapkan dapat diberikan rekomendasi untuk meningkatkan kinerja dan aplikasi OCR di masa mendatang.

## **2. Tujuan**

Tujuan dari penelitian ini adalah untuk menganalisis efektivitas dan akurasi *Optical Character Recognition* (OCR) dalam mengkonversi teks dari gambar menjadi teks yang dapat diedit. Penelitian ini juga bertujuan untuk:

1. Mengidentifikasi faktor-faktor yang mempengaruhi performa OCR, seperti kualitas gambar, jenis font, ukuran teks, dan resolusi.
2. Mengevaluasi kemampuan OCR dalam berbagai kondisi dan karakteristik gambar.
3. Memberikan rekomendasi praktis untuk meningkatkan akurasi dan efisiensi penggunaan OCR dalam proses digitalisasi dokumen.

## **3. Manfaat Penelitian**

Manfaat dari penelitian ini dikategorikan menjadi dua bagian, yakni manfaat teoritis dan manfaat praktis.

### **3.1. Manfaat Teoritis**

- **Pengembangan Ilmu Pengetahuan:** Penelitian ini memberikan kontribusi dalam pengembangan ilmu pengetahuan di bidang pengolahan citra digital, khususnya mengenai penggunaan dan optimasi teknologi OCR.
- **Pemahaman Mendalam tentang OCR:** Menambah wawasan tentang faktor-faktor yang mempengaruhi performa OCR, sehingga dapat digunakan sebagai referensi untuk penelitian selanjutnya.

- **Framework Evaluasi OCR:** Menghasilkan framework atau metode evaluasi yang dapat digunakan oleh peneliti lain untuk mengukur efektivitas dan akurasi OCR dalam berbagai kondisi.

### 3.2. Manfaat Praktik

- **Efisiensi Pengarsipan:** Dengan menggunakan OCR, proses digitalisasi dokumen akan menjadi lebih efisien, mengurangi kebutuhan penyimpanan fisik, dan mempercepat pencarian dokumen berdasarkan isi teks.
- **Keamanan dan Keawetan Dokumen:** Dokumen digital lebih aman dari kerusakan fisik seperti banjir atau kebakaran, dan risiko kehilangan dapat diminimalkan melalui proses backup yang mudah.
- **Aksesibilitas Data:** Meningkatkan aksesibilitas dan kemampuan pencarian data pada dokumen yang telah dipindai, memungkinkan pengguna untuk mencari informasi spesifik langsung dari teks yang diekstrak.
- **Peningkatan Produktivitas:** Mengurangi waktu dan tenaga yang diperlukan untuk memproses dan mengelola dokumen, sehingga meningkatkan produktivitas di berbagai sektor yang memerlukan digitalisasi dokumen, seperti perkantoran, perpustakaan, dan arsip pemerintahan.

## 4. Metodologi

### 4.1. *Optical Character Recognition (OCR)*

*Optical Character Recognition (OCR)* merupakan suatu teknologi yang digunakan untuk mengenali citra huruf dan angka dengan tujuan mengubahnya menjadi format teks. Implementasi teknologi ini dapat meningkatkan fleksibilitas serta kemampuan sistem komputer. Secara umum proses OCR meliputi file input, preprocessing, segmentasi, normalisasi, ekstraksi ciri dan *recognition*.



**Gambar 1** Tahapan OCR

Seperti yang terlihat pada gambar diatas, penjelasan mengenai proses OCR adalah sebagai berikut:

a. File Input

File input berupa file citra digital dengan format \*.bmp atau \*.jpg.

b. *Preprocessing*

*Preprocessing* merupakan suatu proses untuk menghilangkan bagian-bagian yang tidak diperlukan pada gambar input untuk proses selanjutnya.

Pada tahap ini, gambar yang di input akan diproses untuk meningkatkan kualitasnya dengan menghilangkan noise dan bagian-bagian yang tidak diperlukan. Langkah-langkah yang digunakan termasuk:

- Mengubah gambar menjadi *grayscale*.
- Mengurangi *noise* dengan *Gaussian blur*.
- Menggunakan *Thresholding Otsu* untuk mengubah gambar *grayscale* menjadi biner.

c. Segmentasi

Segmentasi melibatkan pemisahan bagian-bagian penting dari gambar, seperti karakter teks, dari latar belakang atau elemen lain yang tidak relevan.



d. Normalisasi

Normalisasi mengubah dimensi dan ketebalan karakter agar seragam, yang membantu dalam proses pengenalan karakter selanjutnya. Normalisasi biasanya dilakukan dengan merubah ukuran gambar atau objek tersegmentasi ke ukuran yang seragam

e. Ekstraksi Ciri

Pada tahap ini, fitur-fitur penting dari setiap karakter diidentifikasi dan diekstraksi untuk memudahkan pengenalan.

f. *Recognition*

Tahap terakhir adalah pengenalan karakter, di mana fitur-fitur yang diekstraksi dibandingkan dengan basis data karakter yang sudah ada untuk mengidentifikasi teks.

#### 4.2. Peralatan dan Perangkat Lunak

- **Perangkat Keras:** Komputer atau laptop dengan spesifikasi yang memadai untuk menjalankan perangkat lunak OCR.
- **Perangkat Lunak:** Menginstal Tesseract OCR dan library yang diperlukan seperti `pytesseract` dan `opencv`.
- **Dataset:** Mengumpulkan berbagai gambar dokumen dengan karakteristik yang berbeda, seperti jenis font, ukuran teks, dan resolusi gambar.
- **Instalasi Tesseract:** Mengikuti langkah-langkah instalasi dari sumber yang terpercaya  
(misalnya, <https://github.com/UB-Mannheim/tesseract/wiki>).
- **Konfigurasi Tesseract:** Menentukan jalur ke executable Tesseract pada sistem.

## 5. Hasil dan Pembahasan

### 5.1. Pengujian *Pre-Processing*

Pada tahap *pre-processing*, data hasil scan dokumen diproses sebelum masuk ke modul OCR. *Pre-processing* yang dilakukan termasuk *Thresholding* pada gambar hasil scan dengan menggunakan metode *Otsu Thresholding*.



Gambar 2 Tahap *Otsu Thresholding*

Tujuannya adalah untuk membersihkan gambar agar hasil ekstraksi karakter menjadi lebih baik.

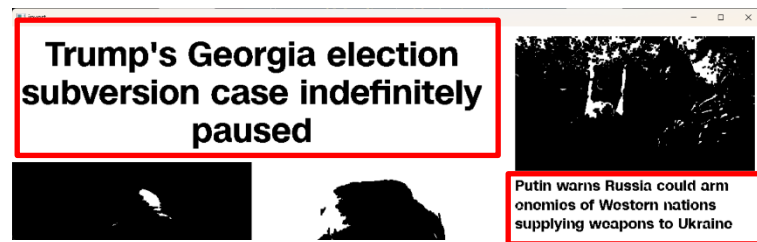
### 5.2. Pengujian Ekstraksi Teks

#### a. *Line Removal*

Mengidentifikasi dan menghapus garis dan tanda yang bukan karakter untuk meningkatkan pengenalan karakter.

b. *Zoning*

Memisahkan bagian gambar menjadi partisi-partisi teks, seperti mendeteksi kolom-kolom jika dokumen yang diproses adalah dokumen multi-kolom.



Gambar 3 *Zoning*

c. *Tokenization*

Menentukan bagian-bagian dari setiap baris pada gambar hasil *Zoning*, mengidentifikasi jarak antar karakter dan menandainya sebagai “token” yang merepresentasikan satu karakter.



Gambar 4 *Tokenization*

d. *Pattern Recognition*

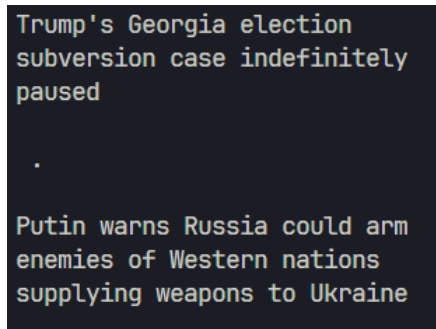
Setiap token dibandingkan per pixel dengan dataset karakter yang disediakan, termasuk angka, tanda baca, dan simbol-simbol lainnya. Teknik ini dikenal sebagai *matrix matching*.

Tabel 1 *Pattern Recognition*

Token	<b>T</b>	<b>r</b>	<b>u</b>	<b>m</b>	<b>p</b>	<b>'</b>	<b>s</b>
Karakter	T	r	u	m	p	,	s

### 5.3. Hasil Pengujian ke Teks

Berikut adalah hasil yang terlihat setelah melewati tahapan-tahapan sebelumnya.



```
Trump's Georgia election
subversion case indefinitely
paused

.

Putin warns Russia could arm
enemies of Western nations
supplying weapons to Ukraine
```

*Gambar 5 Output*

## 6. Kesimpulan dan Saran

Penelitian ini menyoroti pentingnya *Optical Character Recognition* (OCR) dalam digitalisasi dokumen, khususnya dalam mengonversi teks dari gambar menjadi teks yang dapat diedit. Teknologi OCR terbukti efektif dengan tingkat akurasi yang tinggi, namun performanya sangat dipengaruhi oleh kualitas gambar, jenis font, ukuran teks, dan resolusi gambar. Metode *pre-processing* yang tepat seperti konversi ke *grayscale*, pengurangan noise, dan *Thresholding Otsu* sangat penting untuk meningkatkan hasil ekstraksi karakter.

Untuk meningkatkan akurasi dan efisiensi OCR, disarankan untuk menggunakan gambar dengan resolusi optimal dan font yang sederhana serta seragam. Mengadopsi teknik *pre-processing* yang lebih canggih dan memperbarui perangkat lunak OCR dengan algoritma pengenalan yang lebih baik juga akan memberikan hasil yang lebih baik. Dengan memahami faktor-faktor yang mempengaruhi performa OCR dan mengimplementasikan rekomendasi ini, proses digitalisasi dokumen dapat menjadi lebih efisien, aman, dan mudah diakses, mendukung pengarsipan digital yang lebih baik dan meningkatkan produktivitas di berbagai sektor.

## DAFTAR PUSTAKA

- Ibnutama, K., Panjaitan, Z., & Ginting, E. F. (2019). J-SISKO TECH Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD Modifikasi Metode Template Matching pada OCR Untuk Meningkatkan Akurasi Deteksi Plat Nomor Kendaraan. v, *21*(2), 21–29.
- Rismanto, R., Prasetyo, A., & Irawati, D. A. (2020). Optimalisasi Image *Thresholding* pada *Optical Character Recognition* Pada Sistem Digitalisasi dan Pencarian Dokumen. *PETIR*, *13*(1), 1–11. <https://doi.org/10.33322/petir.v13i1.659>

## LAMPIRAN

### Step 1. Install Tessereact on your device.

Kunjungi link berikut, <https://github.com/UB-Mannheim/tesseract/wiki> dan download *Tessereact installer* for *Windows* dari link tersebut.

#### Tesseract installer for Windows

Normally we run Tesseract on Debian GNU Linux, but there was also the need for a Windows version. That's why we have built a Tesseract installer for Windows.


**WARNING:** Tesseract should be either installed in the directory which is suggested during the installation or in a new directory. The uninstaller removes the whole installation directory. If you installed Tesseract in an existing directory, that directory will be removed with all its subdirectories and files.

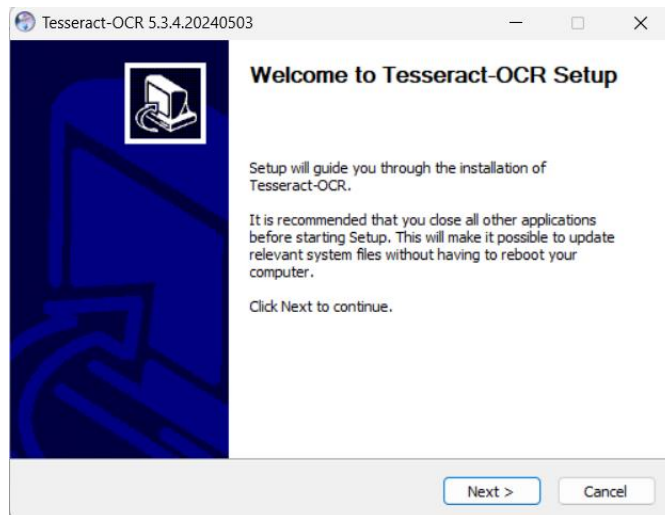
The latest installers can be downloaded here:

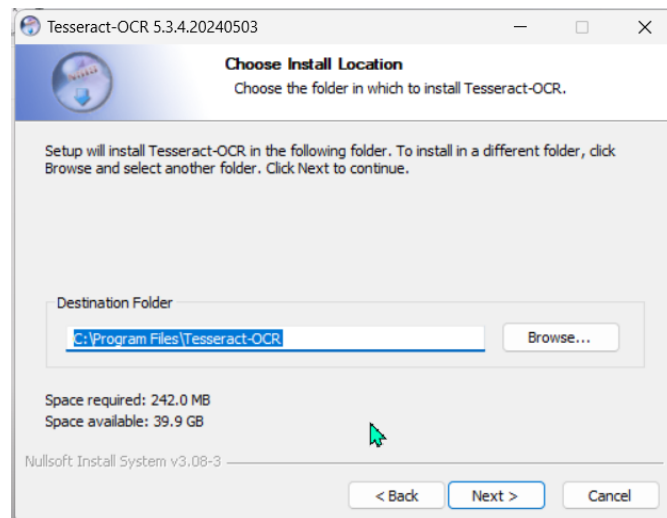
- [tesseract-ocr-w64-setup-5.3.4.20240503.exe \(64 bit\)](#)
- [tesseract-ocr-w64-setup-5.4.0.20240519-1-ga5ff320e.exe \(64 bit\)](#)

There are also [older versions](#) for 32 and 64 bit Windows available.

Kemudian, jalankan *Tesseract setup* dan selesaikan instalasinya

 tesseract-ocr-w64-setup-5.3.4.20240503....	06/06/2024 10:00	Application	49.600 KB
--	------------------	-------------	-----------





Note: Pastikan settingan default dan klik **next**, **next**, **next**, dan **complete installation**.

## Step 2. Install Tessereact OCR dan libraries yang dibutuhkan

Tesseract adalah mesin OCR sumber terbuka yang dikelola oleh Google. Sebelum menjalankan Tessereact, kita harus install library pytesseract terlebih dahulu.

Pip install pytesseract

```
PS C:\Users\Nana Casmana\Documents> pip install pytesseract
Requirement already satisfied: pytesseract in c:\users\nana casmana\appdata\local\programs\python\python310\lib\site-packages (0.3.10)
Requirement already satisfied: packaging>=21.3 in c:\users\nana casmana\appdata\local\programs\python\python310\lib\site-packages (from pytesseract) (23.1)
Requirement already satisfied: Pillow>=8.0.0 in c:\users\nana casmana\appdata\local\programs\python\python310\lib\site-packages (from pytesseract) (9.1.0)
[notice] A new release of pip available: 22.1.2 -> 24.0
[notice] To update, run: python.exe -m pip install --upgrade pip
```

### Step 3. Persiapkan Libraries dan Code

#### 1. Import libraries yang dibutuhkan

```
import cv2
import pytesseract
```

- `cv2` adalah modul OpenCV yang digunakan untuk pemrosesan gambar.
- `pytesseract` adalah wrapper untuk Tesseract OCR, digunakan untuk ekstraksi teks dari gambar.

#### 2. Menentukan Lokasi Tesseract

```
pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
```

Mengatur jalur ke *executable* Tesseract. Sesuaikan dengan lokasi instalasi Tesseract di sistem Anda.

#### 3. Membaca dan Mengkonversi Gambar

```
# Grayscale, Gaussian blur, Otsu's threshold
image = cv2.imread("C:/Users/Nana Casmana/Downloads/object4.png")
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (1, 1), 0)
thresh = cv2.threshold(
    blur, 0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1]
```

- `cv2.imread` membaca gambar dari file yang ditentukan.
- `cv2.cvtColor` mengkonversi gambar berwarna ke *grayscale*.
- `cv2.GaussianBlur` mengurangi noise pada gambar dengan Gaussian blur menggunakan kernel 3x3.
- `cv2.threshold` menerapkan *Thresholding* Otsu untuk mengubah gambar *grayscale* menjadi gambar biner (hitam-putih), dengan pembalikan (inverse) warna. Hasilnya adalah gambar biner yang disimpan dalam `thresh`.



#### 4. Menghilangkan Noise dan Membalik Warna

```
# Morph open to remove noise and invert image
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=1)
invert = 255 - opening
```

- `cv2.getStructuringElement` membuat elemen struktur untuk operasi morfologi (kernel 3x3 dalam hal ini).
- `cv2.morphologyEx` menerapkan operasi morfologi 'open' untuk menghilangkan noise kecil dari gambar biner.
- `255 - opening` membalikkan warna gambar (hitam menjadi putih dan sebaliknya).

#### 5. Mengekstraksi Teks dari Gambar

```
# Perform text extraction
data = pytesseract.image_to_string(invert, config='--psm 4')
print(data)
```

- `pytesseract.image_to_string` menggunakan Tesseract OCR untuk mengekstrak teks dari gambar yang telah diproses (`invert`).
- `config='--psm 4'` menentukan mode segmentasi halaman (PSM). Mode 4 berarti Tesseract menganggap gambar sebagai satu blok teks tunggal.
- `print(data)` mencetak teks yang diekstraksi ke konsol.

#### 6. Tampilkan Gambar yang Diproses

```
cv2.imshow('thresh', thresh)
cv2.imshow('opening', opening)
cv2.imshow('invert', invert)
cv2.waitKey()
```

- `cv2.imshow` menampilkan gambar hasil setiap tahap pemrosesan (`thresh`, `opening`, dan `invert`) dalam jendela terpisah.
- `cv2.waitKey()` menunggu penekanan tombol di jendela tampilan

gambar. Ini diperlukan agar jendela tampilan tetap terbuka sampai pengguna menekan tombol.

#### Step 4. Jalankan Program

Siapkan gambar atau object yang ingin di convert ke dalam teks.

**Trump's Georgia election  
subversion case indefinitely  
paused**



Putin warns Russia could arm  
enemies of Western nations  
supplying weapons to Ukraine

Kemudian, jalankan *code* nya.

```
uas.py x
uas.py > ...
1 import cv2
2 import pytesseract "pytesseract": Unknown word.
3
4 pytesseract.pytesseract.tesseract_cmd = r"C:\Program Files\Tesseract-OCR\tesseract.exe"
5
6 # Grayscale, Gaussian blur, Otsu's threshold "Otsu's": Unknown word.
7 image = cv2.imread("C:/Users/Nana Casmana/Downloads/object4.png") "imread": Unknown w
8 gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
9 blur = cv2.GaussianBlur(gray, (1, 1), 0)
10 thresh = cv2.threshold(
11     blur, 0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)[1] "OTSU": Unknown word.
12
13 # Morph open to remove noise and invert image
14 kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3, 3))
15 opening = cv2.morphologyEx(thresh, cv2.MORPH_OPEN, kernel, iterations=1)
16 invert = 255 - opening
17
18 # Perform text extraction
19 data = pytesseract.image_to_string(invert, config='--psm 4') "pytesseract": Unknown w
20 print(data)
21
22 cv2.imshow('thresh', thresh) "imshow": Unknown word.
23 cv2.imshow('opening', opening) "imshow": Unknown word.
24 cv2.imshow('invert', invert) "imshow": Unknown word.
25 cv2.waitKey()
```

Hasilnya akan seperti ini:



Dapat dilihat bahwa program berhasil mengconvert teks yang ada di dalam foto menjadi teks biasa.