Urdu Text Recognition from Images and PDFs

1. Project Overview and Approach

This project implements an end-to-end OCR system tailored for Urdu text from images or PDFs, providing not only text extraction but also text cleaning and accuracy measurement against reference transcriptions. The goal is to enable users to upload Urdu documents, perform OCR with preprocessing, and evaluate the OCR quality through standard metrics like Character Error Rate (CER) and Word Error Rate (WER).

Design Decisions

- **Modular Design:** Separation of OCR, preprocessing, metrics, and utilities in different modules (core/ocr.py, core/preprocessing.py, core/metrics.py, utils/file_utils.py) for maintainability and extensibility.
- **Preprocessing Pipeline:** Use of OpenCV for image denoising, thresholding, resizing, and optional deskewing to enhance OCR accuracy.
- **Flexible Input Support:** Ability to process both image files (PNG, JPG, TIFF) and multi-page PDFs.
- **Accuracy Metrics:** Implementation of Levenshtein distance based CER and WER metrics for quantitative evaluation.
- **User Interface:** Streamlit app for quick interactive demos and Gradio app for easy deployment on platforms like Hugging Face Spaces.
- **Cross-Platform Compatibility:** Auto-detection of Tesseract OCR binary paths and handling of Windows and Linux environments, including deployment considerations.

---

2. Tools, Frameworks, and Libraries Used

- **Python 3.8+** — main programming language
- **Tesseract OCR** — open-source OCR engine for text extraction
- **Pytesseract** — Python wrapper for Tesseract
- **OpenCV (cv2)** — image preprocessing (denoising, thresholding, resizing, deskewing)
- **Pillow (PIL)** — image loading and manipulation

- **pdf2image** — convert PDF pages to images for OCR
- **NumPy** — array manipulation and Levenshtein distance implementation
- **Streamlit** — building the interactive web dashboard UI for local testing and demo
- **Gradio** — lightweight web UI for easy deployment on Hugging Face Spaces
- **Poppler-utils** — system dependency for PDF rendering (needed on Linux)

---

3. Dataset(s) Used or Created

- **Sample Urdu Image and PDF files:** Synthetic Urdu text documents created to test OCR pipeline.
- **Reference Text Files:** Manually created ground-truth Unicode text files corresponding to the image or PDF inputs to measure OCR accuracy.
- **No public dataset dependency:** The project can work with any user-provided Urdu documents.

---

4. Preprocessing Steps

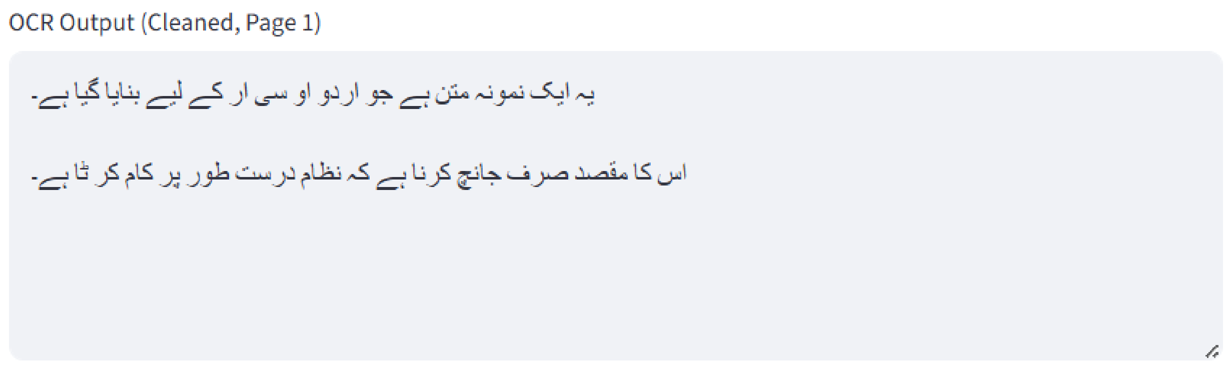
The preprocessing pipeline improves OCR quality by:

1. **Conversion to Grayscale:** Simplifies input image data.
2. **Resizing:** Limits the maximum image dimension to 1800 pixels to balance OCR performance and speed.
3. **Denoising:** Uses bilateral filtering to reduce noise while preserving edges.
4. **Thresholding:** Otsu's method binarizes the image to enhance character distinction.
5. **Deskewing (Optional):** Detects skew angle and rotates the image to align text horizontally.
6. **Conversion Back to PIL Image:** For compatibility with pytesseract.

---

5. Accuracy Results and Sample Outputs

Sample OCR Output (Cleaned Text)

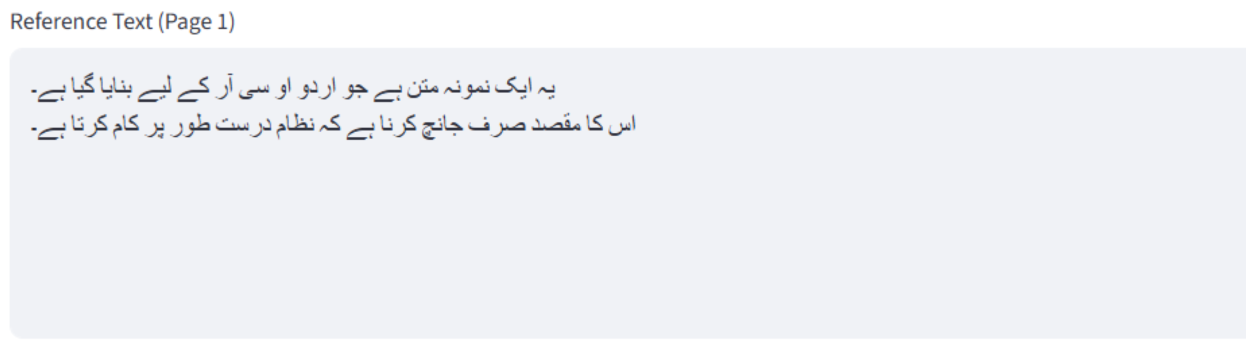


OCR Output (Cleaned, Page 1)

یہ ایک نمونہ متن ہے جو اردو او سی ار کے لیے بنایا گیا ہے۔

اس کا مقصد صرف جانچ کرنا ہے کہ نظام درست طور پر کام کر ٹا ہے۔

Sample Reference Text



Reference Text (Page 1)

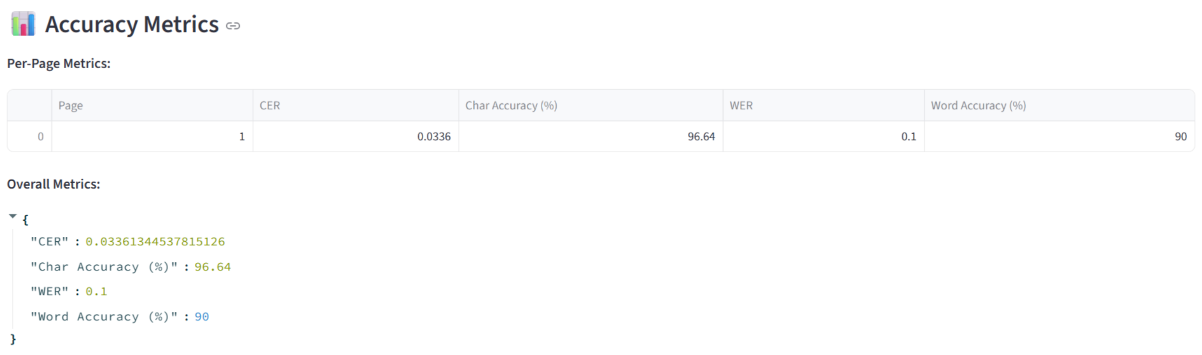
یہ ایک نمونہ متن ہے جو اردو او سی آر کے لیے بنایا گیا ہے۔
اس کا مقصد صرف جانچ کرنا ہے کہ نظام درست طور پر کام کرتا ہے۔

Accuracy Metrics

Page	CER	Char Accuracy (%)	WER	Word Accuracy (%)
1	0.0336	96.64	0.1	90.0

Overall:

- Character Error Rate (CER): 3.36%
- Word Error Rate (WER): 10%
- Character Accuracy: 96.64%
- Word Accuracy: 90%


📊 Accuracy Metrics

Per-Page Metrics:

	Page	CER	Char Accuracy (%)	WER	Word Accuracy (%)
0	1	0.0336	96.64	0.1	90

Overall Metrics:

```
{
  "CER" : 0.03361344537815126
  "Char Accuracy (%)" : 96.64
  "WER" : 0.1
  "Word Accuracy (%)" : 90
}
```

6. Challenges Faced

- **Urdu Language Support:** Ensuring the correct Tesseract Urdu language data is installed and accessible.
- **PDF Processing:** Handling PDFs on Windows requires external Poppler utilities and path configuration.
- **Text Normalization:** Urdu text has complex Unicode forms and optional diacritics that complicate normalization and cleaning.
- **OCR Errors:** Urdu script's cursive nature, diacritics, and noise in scanned documents affect OCR accuracy.
- **Deployment Constraints:** Ensuring the system works cross-platform and on hosted environments like Hugging Face Spaces where installing system dependencies may be limited.

7. Future Improvements

- **Enhanced Preprocessing:** Add adaptive contrast enhancement, morphological operations, and noise removal tuned for Urdu.
- **Better Language Models:** Integrate language models or neural OCR for Urdu to reduce recognition errors.
- **Error Correction:** Post-process OCR output with spelling correction and linguistic rules.
- **UI Improvements:** Add annotation tools and batch export options.

- **Support More Formats:** Add support for handwritten Urdu text, scanned books, and real-time mobile capture.
- **Multi-lingual Support:** Extend the system to other languages and scripts with minimal changes.

Riyyan Muhyouddin Kharal