

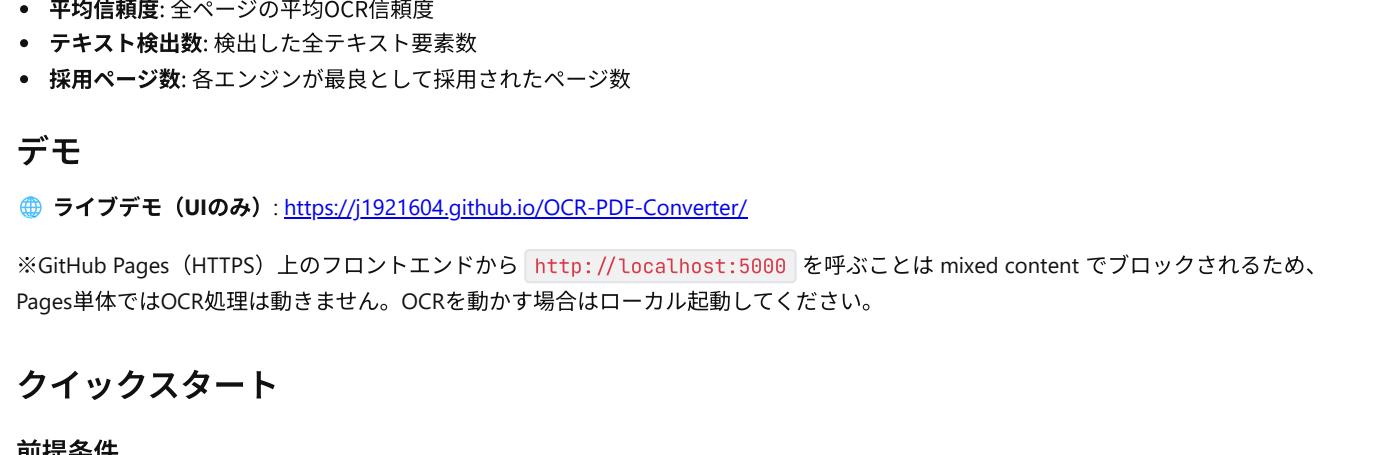
OCR検索可能PDF変換 Webアプリ

License MIT demo GitHub Pages version 1.0.0

スキャンしたPDFや画像を高精度OCRエンジン（OnnxOCR・PaddleOCR）で処理し、検索可能なテキストレイヤーを追加するWebアプリケーション

特徴

- 高精度OCRエンジン - OnnxOCR（高速CPU推論）、PaddleOCR（高精度）から選択可能
- OCRエンジン選択 - UI上でエンジンを切り替えて最適な結果を選択
- 画像ファイル選択 - JPEG、PNG、TIFF画像も直接OCR処理可能
- pyPDF2 + React + API操作 - パッケージでPythonフロントエンドでReact
- 日本語OCR最適化 - 日本語に特化した高精度認識
- 複数ページ対応 - バッチ処理でリアルタイム進捗表示
- ファイル制限 - 最大50MB対応
- 透明テキストレイヤー - ReportLabで完全透明なテキストレイヤーを合成
- ダークモードUI - 黒とオレンジを基調とした立体的で金属的なデザイン
- A4以外対応 - A3、Letter、Legal、カスタムサイズにも対応



技術スタック

バックエンド (Python 3.10.11)

- OnnxOCR 2025.5: 高速CPU推論OCRエンジン (PaddleOCRベース)
- PaddleOCR 2.7.0.3: 高精度多機能OCRエンジン
- pyPDF2 4.30: PDFレンダリング
- pyPDF3 3.7: PDF合成
- ReportLab 4.0: 透明テキストレイヤー生成
- Flask 3.0: REST APIサーバー
- OpenCV 4.6: 画像前処理
- NumPy 1.24: 数値計算
- Pillow 10.4: 画像処理

フロントエンド

- React 18: UIフレームワーク
- Webpack 5: モジュールバンドラー

OCRエンジン並列処理の詳細

本アプリケーションは、複数のOCRエンジンを並列実行し、最も高精度な結果を自動選択する独自アルゴリズムを実装しています。

処理フロー

- エンジン選択: UIでOnnxOCR、PaddleOCRから複数選択可能
- 並列OCR実行: 各PDFページで全選択エンジンを同時に実行
- 精度評価: 各エンジンの平均信頼度 (confidence) を計算
- 最高エンジン選択: 平均信頼度が最も高いエンジンの結果を採用
- 透明テキスト埋め込み: 最良エンジンの結果で検索可能なPDF生成

実装詳細

コード位置: [backend/main.py L630-L672](#)

```
# 各ページで全エンジンを実行し、最良の結果を選択
for page_num in range(page_count):
    # 全エンジンでOCR実行
    engine_results = {}
    for eng in engines_to_use:
        ocr_results = run_ocr(pil_img, eng)
        ocr_items = normalize_ocr_results(ocr_results, confidence_threshold)
        if eng:
            avg_confidence = sum(item['confidence'] for item in ocr_items) / len(ocr_items)
            engine_results[eng] = {
                'items': ocr_items,
                'avg_confidence': avg_confidence,
            }
    # 最高エンジン結果を選択 (平均信頼度が最も高いもの)
    best_engine = None
    best_confidence = -1.0
    for eng, res in engine_results.items():
        if res['avg_confidence'] > best_confidence:
            best_confidence = res['avg_confidence']
            best_engine = eng
            best_result = res

    # 透明テキストレイヤーPDF作成 (最良エンジンの結果を使用)
    if best_result:
        overlay_bytes = create_overlay_pdf(
            page_w_pt, page_h_pt, best_result['items'],
            scale_x=scale_x, scale_y=scale_y,
        )

```

統計情報表示

処理完了後、各エンジンの精度統計をUIに表示：

- 平均信頼度: 全ページの平均OCR信頼度
- テキスト件数: 検出した全テキスト要素数
- 採用ページ数: 各エンジンが最もとして採用されたページ数

デモ

➊ ライブデモ (UIのみ) : <https://192.1604.github.io/OCR-PDF-Converter/>

※GitHub Pages (HTTPS) 上のフロントエンドから <http://localhost:5000> を呼ぶことは mixed content でブロックされるため、Pages単体ではOCR処理は動きません。OCRを動かす場合はローカル起動してください。

クイックスタート

前提条件

- [Python 3.10.11](#)
- [Node.js 18以上](#)
- [npm](#) または [yarn](#)

ワンコマンド起動 (PowerShell - 推奨)

```
.\start-full.ps1
```

このスクリプトは以下を自動実行します：

- Python 3.10.11 or Node.jsのインストール確認
- 依存関係のインストール (Python + npm)
- Pythonバックエンド起動 (<http://localhost:5000>)
- Reactフロントエンド起動 (<http://localhost:8080>)

サーバーを停止するには [Ctrl+C](#) を押します。

手動セットアップ

1. バックエンド起動

```
py -3.10 -m venv .venv
.venv\Scripts\Activate.ps1
py -3.10 -m pip install --upgrade pip
py -3.10 -m pip install -r requirements.txt
python backend\app.py
```

2. フロントエンド起動 (別ターミナル)

```
npm install
npm start
```

ブラウザで <http://localhost:8080> を開きます。

使い方

1. ファイルを選択

「ファイルを選択」ボタンをクリックし、スキャンしたPDFファイルまたは画像ファイル (JPEG、PNG、TIFF、50MB以下) を選択します。

対応形式: PDF / JPEG / PNG / TIFF (画像はフロント側でPDFに変換してから送信します)

2. OCR変換開始

OCRエンジン (OnnxOCR または PaddleOCR) を選択し、「OCR変換開始」ボタンをクリックすると、Pythonバックエンドで以下の処理が実行されます：

- 選択されたOCRエンジンでPDFの各ページをOCR処理
- OCR精度 (平均信頼度) をリアルタイム表示
- 進捗バーでリアルタイムに処理状況を確認

3. PDFファイル読み込みエラー

症状: PDFをアップロード後、エラーメッセージが表示される

原因:

- ファイルサイズが50MBを超えてる
- 別のPDF形式
- 暗号化されたPDFファイル

解決方法:

- ファイルサイズを確認 (50MB以下)
- 別のPDF形式
- 暗号化されたPDFファイルを解除してから再試行

4. OCR精度が低い

原因:

- スキャン解像度が低い (推奨: 300dpi以上)
- 画像が斜めになっている
- 低品質なスキャン画像

解決方法:

- OCR処理は選択したエンジン (OnnxOCR/PaddleOCR) で自動的に300dpiで処理します
- 高解像度でスキャンし直す
- 暗号化されたPDFファイルを解除してから再試行

5. PaddleOCR SSL証明書検証エラー

症状: PaddleOCRエンジン使用時に以下のエラーが表示される

```
SSLError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self signed certificate in certificate chain
```

原因:

• 証明書環境やファイアウォール配下で、PaddleOCRのモデルダウンロード時に自己署名証明書が使用されている

• モルダルランドー元 (<http://aorl.bjcebos.com>) の接続でSSL検証が失敗する

実装済み対応: backend/main.py (L1-L40) で以下の対応が実装されています：

1. SSL証明書検証を無効化:

```
import ssl
ssl._create_default_https_context = ssl._create_unverified_context
```

2. urllib3警告を抑制:

```
import urllib3
urllib3.disable_warnings(InsecureRequestWarning)
```

3. 環境変数を設定:

```
os.environ['REQUESTS_CA_BUNDLE'] = ''
os.environ['CURL_CA_BUNDLE'] = ''
```

4. requests.getをバッティング:

```
o.ensure_paddleocr_available() 関数でrequests.getに verify=False を自動注入
```

手動対応 (上記が効かない場合):

```
# PowerShellで環境変数を設定
$env:REQUESTS_CA_BUNDLE = ""
$env:CURL_CA_BUNDLE = ""

# ポート起動
py -3.10 backend/app.py
```

セキュリティ注意事項:

- この設定はローカル開発環境でのみ使用してください
- プロダクション環境では適切なCA証明書を設定することを推奨します

トラブルシューティング

1. サーバーが起動しない

症状: `npm start` または `.\start-full.ps1` 実行後、サーバーが自動停止する

解決方法:

- まず `.\start-full.ps1` の利用を推奨します (同一ウィンドウ内で安定起動/停止します)。
- 8080/5000番ポートが他のサービスで使用中の場合は、先に停止してから再実行してください。

2. CSP violation エラー

症状: ブラウザでエラーが表示される

原因:

- CSP設定が未定義している
- `public/index.html` の CSP meta tag を確認

補足: API接続先 (`connect-src`) やワーカー (`worker-src`) の許可が不足していると発生します。

3. PDFファイル読み込みエラー

症状: PDFをアップロード後、エラーメッセージが表示される

原因:

- ファイルサイズが50MBを超えてる

解決方法:

- ファイルサイズを確認 (50MB以下)

4. OCR精度が低い

原因: スキャン解像度が低い (推奨: 300dpi以上)

解決方法:

- 高解像度でスキャンし直す

5. PaddleOCR SSL証明書検証エラー

症状: PaddleOCRエンジン使用時に以下のエラーが表示される

```
SSLError: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: self signed certificate in certificate chain
```

原因:

- 証明書環境やファイアウォール配下で、PaddleOCRのモデルダウンロード時に自己署名証明書が使用されている

• モルダルランドー元 (<http://aorl.bjcebos.com>) の接続でSSL検証が失敗する

実装済み対応: backend/main.py (L1-L40) で以下の対応が実装されています：

1. SSL証明書検証を無効化:

```
import ssl
ssl._create_default_https_context = ssl._create_unverified_context
```

2. urllib3警告を抑制:

```
import urllib3
urllib3.disable_warnings(InsecureRequestWarning)
```

3. 環境変数を設定:

```
os.environ['REQUESTS_CA_BUNDLE'] = ''
os.environ['CURL_CA_BUNDLE'] = ''
```

4. requests.getをバッティング:

```
o.ensure_paddleocr_available() 関数でrequests.getに verify=False を自動注入
```

手動対応 (上記が効かない場合):

```
# PowerShellで環境変数を設定
$env:REQUESTS_CA_BUNDLE = ""
$env:CURL_CA_BUNDLE = ""

# ポート起動
py -3.10 backend/app.py
```

セキュリティ注意事項:

- この設定はローカル開発環境でのみ使用してください
- プロダクション環境では適切なCA証明書を設定することを推奨します

5. オフラインでOCRエンジンをインストール:

• Pythonパッケージ (`onnxocr` / `paddleocr`) のインストール

解決方法:

- Pythonパッケージ (`onnxocr` / `paddleocr`) のインストール

6. ファイルサイズ制限:

• ファイルサイズが50MBを超えてる (推奨: 300dpi以上)

解決方法:

- ファイルサイズを確認 (50MB以下)

7. デバイス接続:

• デバイス接続が確立していない (USB接続など) または、USB接続が正常でない

解決方法:

- USB接続が確立していない場合は、USB接続を確認