令和 6 年度　卒業研究

文字認識 AI を活用したユーザ中心の家計簿アプリの開発

Developing a user-centric household budgeting app using text recognition AI.

函館工業高等専門学校
生産システム工学科　情報コース
佐藤稜隼
指導教員　東海林 智也

目次

1. 序論 3

1.1 概要 3

1.1.1 和文 3

1.1.2 英文 3

1.2 研究背景 4

1.3 研究目的 5

1.4 開発環境 5

2. 関連技術 6

2.1 PaddleOCR 6

2.2 OpenCV 7

2.3 Flask 8

2.4 LINE-SDK-BOT 9

3. 研究結果 10

3.1 実験内容 10

3.2 実験結果 12

3.2.1 文字認識機能 12

3.2.2 LINE アプリの機能 13

3.3 考察 14

4. 今後の展望 14

謝辞 15

参考文献 15

1.1　概要

1.1.1和文

近年、キャッシュレス化が進む中で、現金を使わない傾向が広がっている。このような状況において家計簿をつけることは、収支を把握し節約に役立つことができるため重要である。しかし、実際に家計簿をつけている人の割合は約半数にとどまり、つけていない人が多いのが現状である。その理由として「記録が面倒」「毎日続けるのが難しい」という声がよく挙げられている。このような課題を解決し、より多くの人が簡単に家計管理を行えるようにするため、AI技術を活用した写真を撮るだけで収支を自動的に記録し、整理できる家計簿アプリの開発を検討している。

1.1.2 英文

In recent years, the trend toward a cashless society has been spreading. In this situation, keeping a household account book is important because it can help people keep track of their income and expenditures and help them save money. However, only about half of the respondents actually keep a household account book, and many do not. Commonly cited reasons for this are that “keeping records is a hassle” and “it is difficult to keep up with it every day”. To solve these problems and make it easier for more people to manage their finances, we are considering developing a household account bookkeeping application that can automatically record and organize income and expenses by simply taking a photo using AI technology.

1.2 研究背景

キャッシュレス決済やQRコード決済の普及により、レシートの電子化が進んでいるものの、手書きでの家計管理は多くの人にとって負担となっている。家計簿を継続的に記録する人の割合は約50%にとどまり、記録を行わない理由として「手間がかかる」「モチベーションを維持できない」といった点が挙げられている。また、既存の家計簿アプリの多くは手入力が必須であり、利便性の面でユーザーのニーズに十分応えられていない状況がある。そこで本研究では、OCR（光学文字認識）技術とLINEプラットフォームを組み合わせた自動家計簿システムを開発する。このシステムは、ユーザーがレシートを撮影するだけで、収支データを自動で記録・整理する仕組みを提供する。これにより、家計管理の負担を大幅に軽減し、誰でも手軽に家計簿を継続できることを目指している。

1.3 研究目的

本研究の目的は、家計簿をつけることに負担を感じている人々、特に家計簿をつける習慣がない人や手書きで記録している人の手間を軽減するAIアプリを開発することである。このアプリはOCR（光学文字認識）技術を活用し、レシート情報を正確にテキスト化して家計簿として整理する仕組みを備えている。ユーザーはレシートの写真を撮るだけで収支データを自動的に解析・記録できるため、これまで煩雑で面倒だと感じていた家計管理を簡単に始められる。さらに、操作性の高い設計により、誰でもすぐに利用でき、日常生活での継続的な利用をサポートすることを目指している。

1.4 開発環境

PC : MacBook Air M2
OS : MacOS Sequoia
RAM : 16GB
使用言語 ： Python
開発環境 ： Anaconda、 Visual Studio Code
ライブラリ ： PaddleOCR、 OpenCV、 Flask、 LINE-SDK-BOTなど

2. 関連技術

2.1 PaddleOCR

PaddleOCRは、中国のBaiduが開発したオープンソースの光学文字認識ツールである。PaddlePaddleという深層学習プラットフォームを基盤として構築されており、文字認識を高精度かつ効率的に行うことができる。また、トレーニング済みのモデルを提供しており、ユーザは事前学習モデルを利用することで、高品質な認識結果を簡単に得られる。さらに、独自データを用いたファインチューニングも可能であり、特定の用途や言語に適応したモデルを作成することができる。

PaddleOCRには、主に以下の検出モデルと認識モデルの2つのモデルがある。

・検出モデル： 画像内から文字が含まれる領域を見つけるための仕組みである。DB(Differentiable Binarization)という技術を使い、文字領域を正確に切り出すことができる。また、ResNetやMobileNetなど、複数のバックボーンをサポートしている。

・認識モデル： 検出モデルが見つけた文字領域から実際の文字を読み取る役割を持つ。CRNN(Convolutional Recurrent Neural Network)という構造を採用しており、まず畳み込み層で画像から文字の特徴を抽出し、その後リカレント層で文字の並びを認識する仕組みとなっている。また、Attentionメカニズムを取り入れることで、複雑な配置や異なるフォントの文字でも高い精度で認識できる。

2.2 OpenCV

OpenCV（Open Source Computer Vision Library）は、画像処理やコンピュータビジョンの分野で広く使用されるオープンソースのライブラリであり、リアルタイム処理にも対応した高性能なツールを提供している。本システムでは、OCR（光学文字認識）の前処理を行うためにOpenCVを活用し、画像の品質向上を図ることで認識精度の向上を目指している。 OCR処理の前段階として、まずレシート画像のリサイズを行い、解析しやすい解像度に変換する。これにより、OCRモデルが適切に動作するための画像サイズを確保し、文字の歪みや欠落を最小限に抑えることができる。次に、グレースケール変換を適用し、カラー情報を削除することで、文字と背景のコントラストを強調し、認識の精度を向上させる。さらに、ノイズ除去処理を実施し、背景の影などOCRに悪影響を与える不要な情報を削減する。ノイズ除去にはガウシアンフィルタやメディアンフィルタなどの手法を用いる。 そして、画像の二値化処理を行い、文字部分と背景を明確に区別することで、OCRモデルが文字領域をより正確に認識できるようにする。

2.3 Flask

Flaskは、Pythonで開発された軽量なWebアプリケーションフレームワークであり、シンプルな設計と柔軟な拡張性を兼ね備えている。Flaskは必要に応じてさまざまなライブラリや拡張機能を組み合わせることで、多様なWebアプリケーションの開発が可能となる。そのため、比較的少ないコード量でAPIサーバーを構築でき、迅速な開発が求められるプロジェクトに適している。本システムでは、OCR（光学文字認識）処理をバックエンドで実行するためのWebサーバーの構築にFlaskを採用している。具体的には、クライアントがレシート画像を送信すると、Flaskを用いたバックエンドサーバーがその画像を受け取り、OCR処理を実行した後、解析結果をクライアントに返す仕組みとなっている。Flaskは、リクエストとレスポンスの管理を効率的に行うことができ、OCR処理を非同期的に実行することで、ユーザーの操作をスムーズにする役割を担っている。

2.4 LINE-SDK-BOT

LINE-SDK-BOTはLINE Messaging APIを利用するためのライブラリであり、LINEプラットフォームを通じたメッセージの送受信やデータのやり取りを可能にする。本システムでは、このライブラリを活用し、ユーザーが撮影したレシート画像をLINEアプリ上から送信し、OCR処理を実行した結果をLINEで受け取る仕組みを構築する。このシステムの導入により、家計簿管理を手軽に行うことができ、ユーザーの負担を大幅に軽減できる。

本システムでは、LINE-SDK-BOTを利用して、ユーザーとアプリケーションの間で以下の流れでデータをやり取りする。

1.画像の送信： ユーザーがLINE上でレシートの画像を撮影し、LINEボットに送信する。

2.サーバーでの画像処理： 送信された画像をバックエンドのFlaskサーバーが受け取り、PaddleOCRを用いたOCR処理を実行する。

3.結果の返信： OCR処理で取得したテキストデータを整形し、ユーザーにLINEメッセージとして返信する。 このプロセスにより、ユーザーはLINEアプリ内で家計簿データを確認でき、追加の専用アプリをインストールすることなく、手軽に家計管理ができるようになる。

3. 研究結果

3.1 実験内容

PaddleOCR の検出モデルおよび認識モデルの精度向上を目的として、独自データを用いたファインチューニングを実施した。本実験では、レシートの画像データを用いて、モデルの学習および評価を行い、文字領域の検出精度および認識精度の改善を試みた。学習データと適切なハイパーパラメータの調整により、学習済みである既存モデルと比較して精度向上を検証することを目的とした。

さらに、LINE BOTを用いて家計簿アプリを構築し、ユーザがレシート画像を送信することで自動的に家計簿が作成されるシステムの実現を目指した。

1. データセットの準備

独自の学習データを作成するために、レシート画像を収集し、文字領域のアノテーションを実施した。データセットは以下の2種類に分類した。また、学習データは訓練用と評価用に分割し、検出モデル・認識モデルの精度を独立して評価できるようにした。

・検出モデル用データ：画像全体に対し、文字領域を矩形または多角形でラベリングする。PPOCRLabelを使用し、テキストの位置情報とテキストを作成する。図1に検出モデル用学習データの一部を示す。


```
images/receipt_030.png  [{"transcription": "FamilyMart", "points": [[1121, 1608], [1782, 1573], [1791, 1715], [1134, 1740]], "difficult": false},
images/receipt_031.png  [{"transcription": "FamilyMart", "points": [[1260, 1480], [1914, 1522], [1907, 1667], [1251, 1613]], "difficult": false},
images/receipt_032.png  [{"transcription": "FamilyMart", "points": [[1186, 1819], [1849, 1775], [1863, 1904], [1194, 1952]], "difficult": false},
images/receipt_033.png  [{"transcription": "FamilyMart", "points": [[1216, 1668], [1886, 1667], [1888, 1803], [1219, 1796]], "difficult": false},
images/receipt_034.png  [{"transcription": "FamilyMart", "points": [[1281, 1714], [1939, 1717], [1935, 1849], [1274, 1839]], "difficult": false},
```


図1:検出モデル用学習データの一部

・認識モデル用データ：検出モデルによって切り出された文字領域を小画像として保存する。各画像に対し、対応する文字列ラベルを付ける。図2に認識モデル用学習データの一部を示す。


```
crop_images/receipt_030_0.png   FamilyMart
crop_images/receipt_030_1.png   函館港町店
crop_images/receipt_030_2.png   北海道函館市港町３丁目２番５号
crop_images/receipt_030_3.png   電話：0138-62-2008
crop_images/receipt_030_4.png   登録番号：T2440001008665
crop_images/receipt_030_5.png   2024年5月23日(木) 17:34
```


図2:検出モデル用学習データの一部

2. 設定ファイルの作成

モデルの学習に必要なyml設定ファイルを作成し、データセットのパス、モデルアーキテクチャ、バッチサイズ、学習率、エポック数などのパラメータを設定した。

3. モデルのトレーニング

設定ファイルを元に検出モデルと認識モデルをそれぞれ学習した。

4. モデルの評価

学習後の各モデルの精度を検証するため、以下の指標を用いてテストした。

・検出モデル: IoU（Intersection over Union）を指標とし、文字領域の検出精度を測定した。

・認識モデル: Accuracy（正確度）を指標とし、認識の正確性を評価した。モデルの出力結果を既存の学習済みモデルと比較し、精度向上の度合いを分析した。

5. 家計簿アプリの作成

LINEBOTを利用した家計簿アプリを開発した。本アプリは以下の機能を提供しており、ユーザが手軽に家計管理を行うことを実現している。

・レシート画像の送信による情報登録機能： ユーザがLINEアプリ上でレシートの画像を送信すると、OCR処理が自動的に実行され、画像内の文字情報から購入日、購入金額などの必要なデータが抽出され、家計簿に自動登録される仕組みを構築した。

・レシート履歴の表示機能： 登録されたレシート情報は、日付および購入金額とともに履歴として一覧表示される。これにより、ユーザは過去の支出状況を容易に確認できるようになっている。

・目標金額の設定とグラフ表示機能：ユーザは、月ごとの支出目標金額を設定可能であり、実際の支出と比較した円グラフで視覚的に確認できる。これにより、自身の支出傾向が一目で把握でき、家計管理の改善に役立つ情報を提供している。

3.2 実験結果

3.2.1 文字認識機能

・検出モデルについて

文字領域の検出精度が改善されたことが確認された(図3)。特に、文字間隔が狭い領域や複雑なフォーマットのレシートに対してもより正確に文字領域を囲むことが可能となった。しかし、一部の背景が複雑なレシートでは誤検出が発生するケースも見受けられた。

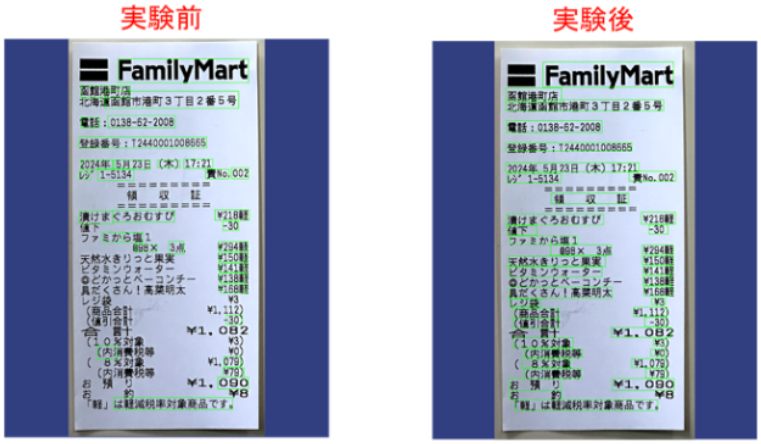

図3:検出モデル用の学習結果

・認識モデルについて

Accuracy（正解率）を評価指標として用いた結果、ファインチューニング前のモデルと比較して、正確に文字を読み取る割合は向上した(図4)。しかし、評価用データ以外のレシートに対して文字認識を行った場合、精度が著しく低下する傾向があった。さらに、学習を繰り返す中で精度の向上は見られず、Accuracyの数値は高いものの、実際の認識結果が十分に正確ではない状況が明らかとなった。

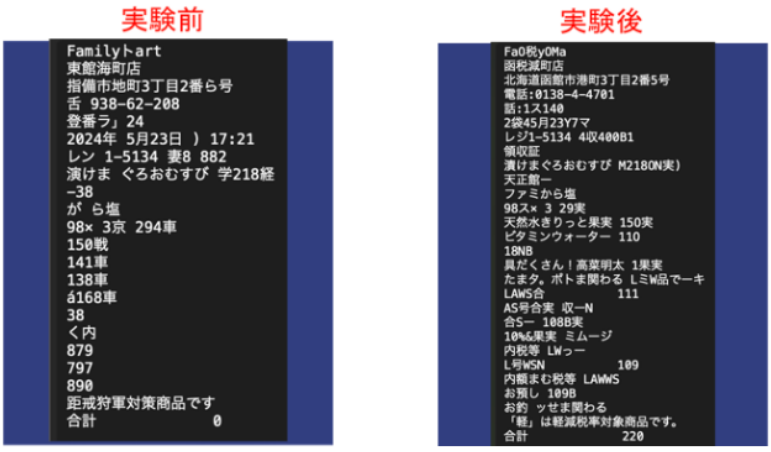

図4:認識モデル用の学習結果

3.2.2 LINE アプリ機能

本研究では、ユーザビリティの向上と実用性を目的として、LINEアプリ上にリッチメニューを活用したインターフェースを実装した(図5)。リッチメニューを通じて、主に以下の3つの機能を提供することができた。

・OCR処理と家計簿情報登録機能：ユーザはLINEアプリ上でレシート画像を送信することで、バックエンドで動作するOCRシステムにより画像内の文字情報が抽出される。その後、抽出された情報を元に、家計簿データが自動的に登録される仕組みを構築した。これにより、従来の手動入力による家計簿作成の手間が大幅に軽減される。

・レシート履歴の確認機能：登録されたレシート情報は、日付や購入金額といった基本情報とともに履歴として一覧表示される。ユーザはリッチメニューから過去の支出データを簡単に確認でき、家計の管理状況を一目で把握できるようになっている。

・目標金額設定および支出状況の可視化機能：ユーザは、月ごとの支出目標金額を設定することが可能である。実際の支出状況は、円グラフなどの視覚的なグラフとして表示され、目標達成度が一目で確認できるよう設計された。これにより、ユーザは自分の使用金額を直感的に理解し、家計管理の改善に役立てることができる。

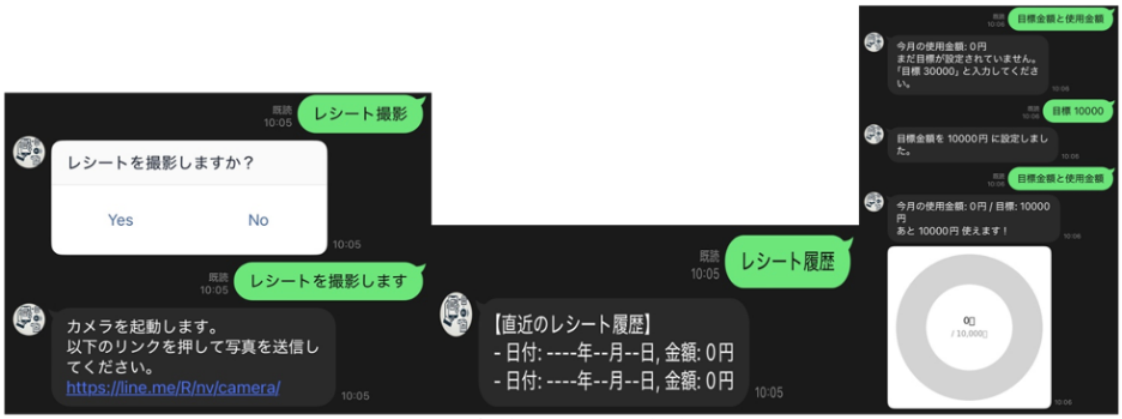

図5:実装した3つの機能

3.3 考察

本実験では、PaddleOCRの検出モデルおよび認識モデルに対し、独自データを用いたファインチューニングを実施した。その結果、検出モデルは十分に学習が進み、文字領域の検出精度が向上したことが確認された。一方で、認識モデルに関しては、評価用データセット上ではAccuracyの値が高いものの、別のデータセットを用いて実際に文字認識を行った場合、期待した精度が得られず、正しい認識結果が得られなかった。この現象は、過学習の影響を受けた可能性が高いと考えられる。

また、本実験ではコンビニレシートを中心とした学習データを使用したため、特定のフォーマットに対しては認識精度が向上したが、飲食店やドラッグストアなど、フォントやレイアウトが異なるレシートに対しては認識精度が低下する傾向が見受けられた。今後は、より多様なフォーマットを含む大規模なデータセットの収集とそれに基づいたモデルの再学習を行い、モデルの認識性能を向上させる必要がある。

さらに、今回の学習では約30枚のレシートを使用した結果、枚数が少ないため、十分な精度向上が得られなかった可能性が示唆される。また、GPUによるトレーニング環境の整備を計画していたにもかかわらず、PaddleOCRのMac環境への構築がうまくいかず、CPU環境での実施となったため、学習時間が大幅に延長された。このことから、AIモデルの開発およびトレーニングには、適切なGPU環境の整備が不可欠であると認識した。

最後に、OCRモデルの精度は向上したものの、誤認識を完全にゼロにすることは困難であるため、実用化に向けてはユーザが誤認識箇所を容易に修正できる仕組みの導入が求められる。例えば、OCR結果をリアルタイムでLINE Botに送信し、ユーザが即座に編集・修正できるインターフェースを設計することにより、システム全体の実用性をさらに向上させることができると考えられる.

4. 今後の展望

現在、データベースのプログラムが十分に構築されておらず、ユーザーの判別が行えていない。この点を改善し、ユーザーごとに適切なデータ管理ができるようにする。また、検出モデルおよび認識モデルのトレーニングをさらに進め、特に文字認識の精度向上を図る。さらに、アプリのユーザーインターフェースを整備し、直感的で使いやすいデザインを目指してアプリ全体を改良する。

謝辞

本研究を進めるにあたり、ご指導くださいました指導教員の東海林智也准教授に感謝いたします。

参考文献

[1] LINE Developers
https://developers.line.biz/ja/

[2] PaddleOCR
https://github.com/PaddlePaddle/PaddleOCR

[3] PaddleOCR/PPOCRLabel
https://github.com/PaddlePaddle/PaddleOCR/tree/release/2.3/PPOCRLabel.

[4] PaddleOCR model
https://paddlepaddle.github.io/PaddleOCR/latest/ppocr/model_list.html