



UNIWERSYTET MARII CURIE-SKŁODOWSKIEJ  
W LUBLINIE

Wydział Matematyki, Fizyki i Informatyki

Kierunek: **Informatyka**

Specjalność: -

**Oleksandr Mertsalov**

Nr albumu: 265513

**Oprogramowanie OCR w zastosowaniu  
praktycznym**

**Practical application of OCR methods**

Praca licencjacka

napisana w Katedrze Oprogramowania Systemów Informatycznych  
pod kierunkiem dr Marcina Denkowskiego

Lublin, rok 2021



# Spis treści

<b>Wstęp.....</b>	<b>5</b>
<b>Rozdział 1. Optyczne rozpoznawanie znaków .....</b>	<b>7</b>
1.1    Przetwarzanie wstępne (ang. Pre-processing).....	7
1.2    Segmentacja tekstu (ang. Text Segmentation) .....	8
1.3    Rozpoznawanie znaków (ang. Character Recognition) .....	10
1.3.1    Rozpoznawanie wzorów (ang. Pattern Recognition).....	11
1.3.2    Rozpoznawanie cech (ang. Feature Detection).....	12
1.4    Przetwarzanie końcowe (ang. Post-processing).....	13
1.5    Tesseract.....	13
<b>Rozdział 2. Sieci neuronowe w rozpoznawaniu znaków .....</b>	<b>15</b>
2.1    Struktura sieci neuronowej.....	15
2.2    Działanie sieci neuronowej .....	15
2.3    Uczenie sieci neuronowej .....	18
2.4    Długoterminowa pamięć krótkoterminowa (ang. Long short-term memory).....	18
<b>Rozdział 3. Aplikacja do rozpoznawania tekstu na paragonie fiskalnym .....</b>	<b>22</b>
3.1    Opis aplikacji .....	22
3.2    Proces uczenia Tesseract .....	22
3.3    Implementacja aplikacji .....	28
3.3.1    OpenCVService .....	28
3.3.2    ReceiptReaderService .....	35
3.3.3    HomeController .....	40
3.4    Testy i rezultaty .....	42
3.4.1    Test silnika Tesseract.....	42
3.4.2    Test aplikacji.....	42
<b>Podsumowanie.....</b>	<b>44</b>
<b>Bibliografia .....</b>	<b>46</b>
<b>Dodatek. Paragony użyte do testów .....</b>	<b>48</b>



## **Wstęp**

Obecnie coraz więcej dokumentów przechowujemy w sposób elektroniczny. Od 2020 roku pojawiła się nawet możliwość załatwienia spraw urzędowych przez Internet i składanie wniosków bez wychodzenia z domu. Trzymanie dokumentów na komputerze zabezpiecza nas od możliwej straty informacji z powodu fizycznych szkód, a w przypadku ich dużej ilości, wielką zaletą jest szybkość podczas sortowania, filtracji i wyszukiwania danych. Dokumenty mogą zawierać określone dane: tekst lub obrazy. Przez tekst rozumiemy zbiór zdań, zdania są zbiorem słów, a słowo składa się z liter.

Wprowadzanie tekstu zawartego w dokumencie do komputera odbywa się za pomocą klawiszy klawiatury, każda z których jest oznakowana pojedynczymi literami, cyframi lub symbolami interpunkcyjnymi. Komputer traktuje wciśnięty klawisz jako unikalny kod binarny lub liczbowy, a następnie używa wybranego przez system pliku czcionki w celu wyświetlenia powiązanego z nim znaku na ekranie monitora. Stworzenie elektronicznej kopii dokumentu poprzez ręczne wpisywania każdego znaku jest dość czasochłonne w przypadku, gdy składa się on z więcej niż jednej strony, ale takie rozwiązanie powoduje, że w wyniku dostajemy stworzony edytowalny tekst.

Szybszym sposobem na przekazanie dokumentu komputerowi jest stworzenie jego wersji cyfrowej za pomocą skanera lub kamery. Taki dokument zostanie umieszczony w pamięci komputera jako plik graficzny zapisany w odpowiednim formacie, najczęściej którymi są: JPEG, PDF, PNG albo TIFF. Tego typu plik przedstawia obraz na ekranie za pomocą zawartej informacji o ilości pikseli oraz ich położenia i koloru. Z tego wynika, że taka wersja dokumentu elektronicznego służy tylko do podglądu jego treści.

Od lat 1950 ciągle się rozwija rozwiązanie, za pomocą którego w dzisiejszych czasach można uzyskać tekst z każdego pliku graficznego. Jest to oprogramowanie komputerowe, które wykorzystuje technikę zwaną optycznym rozpoznawaniem znaków (ang. Optical Character Recognition, OCR). W swojej pracy przedstawię jak przy pomocy silnika OCR rozpoznać tekst z paragonu fiskalnego. Dane znajdujące się na takim dokumencie mogą być bardzo pomocnicze, ponieważ zawierają dużo informacji, za pomocą której można ułatwić robienie zakupów, planować własne finanse oraz regulować ilość kupowanych artykułów w markecie. Paragony fiskalne zawierają informację o czasie zakupu, o sklepie i o tym, jakie można kupić w nim artykuły oraz za jaką cenę. Na podstawie tych danych, można zrobić statystykę tego, jak często robią się zakupy i ile pieniędzy na to idzie. Istnieje duża ilość użytecznych zastosowań dla tego typu danych i pozostaje tylko rozwiązać problem tego, jak ich można uzyskać z papierowego dokumentu.

Celem tej pracy jest opracowanie i implementacja metod do rozpoznawania i segmentacji tekstu na obrazach paragonów fiskalnych hipermarketu „Auchan”. Do optycznego rozpoznawania znaków zostanie użyty silnik Tesseract oparty o sieci neuronowe LSTM (Long Short Therm Memory). Proces uczenia silnika Tesseract zostanie wykonany na platformie Ubuntu (wersja 20.04.3 LTS). W algorytmach przetwarzania obrazu i segmentacji tekstu będzie użyta biblioteka OpenCV. Algorytmy zostaną zaimplementowane jako część aplikacji w języku Java (wersja 12).

W rozdziale pierwszym zostaną omówione podstawowe procesy wykonywane podczas działania oprogramowania OCR oraz to, jakie problemy mogą wystąpić. Ponadto zostanie w nim przedstawione oprogramowanie Tesseract. Rozdział drugi jest poświęcony sieciom neuronowym, zostaną w nim opisane elementy, z których składa się zwykła sieć neuronowa, omówiony zostanie proces wyliczania wyniku działania sieci oraz proces uczenia, w kontekście rozpoznawania znaków. W ostatnim podrozdziale, zostanie opisana architektura rekurencyjnej sieci neuronowej LSTM. Rozdział trzeci ma na celu przedstawić zaimplementowane rozwiązanie do rozpoznania tekstu na paragonach. Zostanie w nim zaprezentowany proces przygotowywania modeli sieci neuronowej silnika Tesseract, opisane będą własne metody do przetwarzania obrazów i segmentacji linii tekstu oraz zostaną przedstawione rezultaty przeprowadzonych testów skuteczności rozpoznawania.

# Rozdział 1. Optyczne rozpoznawanie znaków

Optyczne rozpoznawanie znaków (ang. Optical Character Recognition, OCR) to technologia, która automatycznie rozpoznaje liczby, litery oraz znaki interpunkcyjne w plikach graficznych (rastrowych) i konwertuje je do postaci danych tekstowych kodowanych w sposób maszynowy. Na wejściu podawany jest wcześniej przygotowany plik rastrowy, który poddaje się analizie w celu odnajdywania w nim zbioru pikseli, wizualizujących symbol, a następnie do każdego znalezionej przypisuje się kod binarny lub liczbowy z tablicy znaków. W wyniku zwracany jest tekst. Przy użyciu skanera lub urządzenia wyposażonego w kamerę, rozpoznawanie tekstu może odbyć się w trakcie generowania pliku graficznego. W innym przypadku, możliwe jest przekazanie przygotowanej wcześniej grafiki, zawierającej tekst do systemu OCR.

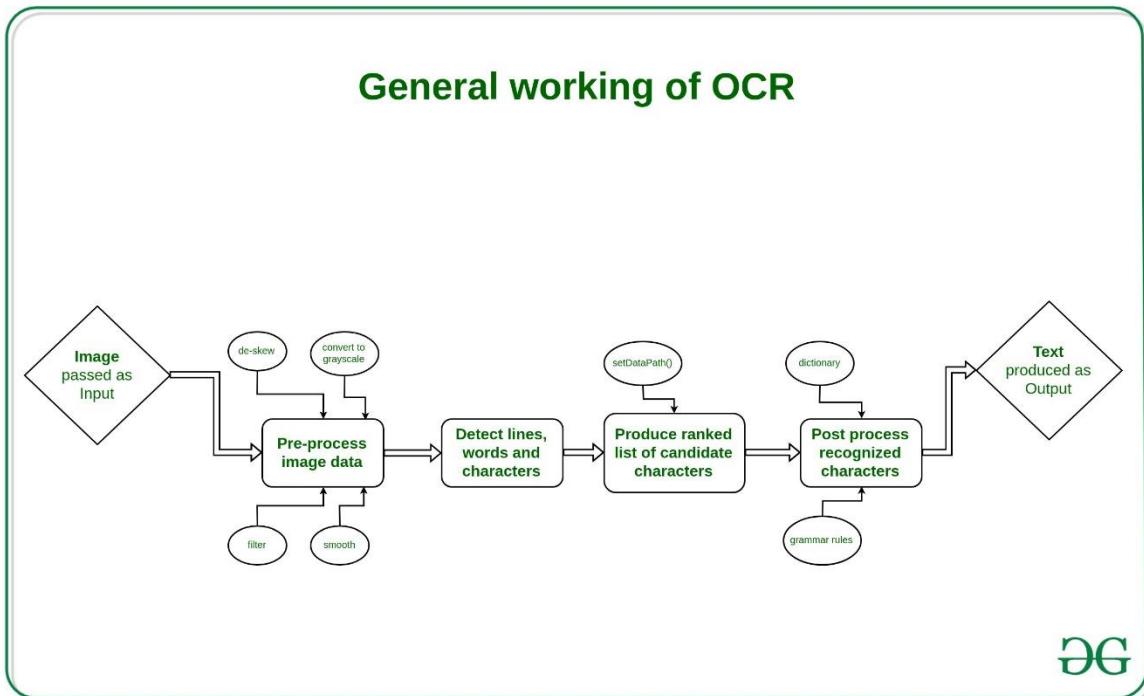
Obecnie nie istnieje idealne oprogramowanie, które zawsze potrafi rozpoznawać tekst z dokładnością do 100%. Prawdopodobieństwo tego, że OCR odniesie sukces podczas pracy zależy w pierwszej kolejności od jakości materiału wejściowego. Im większa jest rozdzielcość obrazu i jego jakość, tym większa szansa, że tekst zostanie poprawnie rozpoznany. Drugim ważnym czynnikiem jest samo oprogramowanie, składające się z algorytmów służących do poprawy obrazu, segmentacji tekstu, rozpoznania znaków i analizie wyniku. W dzisiejszych czasach większość systemów OCR używa silników opartych na sieciach neuronowych, co w rezultacie zwiększa skuteczność rozpoznawania znaków. Dzięki takiemu rozwiązaniu pojawia się możliwość dostosowania silnika do odpowiedniego problemu, a nawet nauczenia go rozpoznawania pisma ręcznego.

W celu wyodrębnienia ciągu znakowego z obrazu używane są różne techniki. Każda technika składa się z procesów, na wejście do których jest podawany wynik poprzedniego kroku, zaczynając od pliku graficznego i kończąc na sformatowanym, edytowalnym tekście (zob. Rysunek 1.1).

## 1.1 Przetwarzanie wstępne (ang. Preprocessing)

Przekazany do OCR obraz, po procesie skanowania może zawierać pewną ilość szumów. Plamy, brud, zagięty lub pomarszczony papier, uszkodzona matryca urządzenia – są to czynniki powodujące pojawienie w pliku rastrowym niepotrzebnych pikseli, utrudniających pracę OCR. Za duży poziom jasności, słaba jakość lub kontrast spowodują zniszczenie kształtu znaków lub ich zamazanie. W tym celu przed procesem identyfikacji

znaków stosuje się do obrazu metody programowe, które służą do tego, aby polepszyć jakość pliku rastrowego i usunąć wszystkie defekty.



Rysunek 1.1. Przebieg procesu OCR. Źródło [1]

Końcowym wynikiem tego etapu musi być obraz, w którym silnik OCR potrafi rozróżnić tekst od pozostałych obiektów. W tym celu do obrazu stosuje się różne algorytmy przetwarzania, które powodują, że tekst zostanie wyróżniony na nim. Na przykład, gdy silnik OCR przyjmuje założenie, że tekst jest zawsze białego koloru, a wszystko pozostałe jest czarne, to w fazie przetwarzania wstępnego, może się użyć algorytm progowania (ang. Threshold). Algorytm ten przekształca kolorowy obraz do skali szarości, zastępując każdy piksel wartościami w przedziale od 0 do 255. Jego działanie może wyglądać następująco: przyjmuje się, że piksel białego koloru jest równy wartości 255, a czarny 0. Dalej, odbywa się zaokrąglenie każdego pikselu według ustawionego progu (najczęściej jest to połowa wartości 255). Jeśli wartość pikselu jest mniejsza od wartości progu, to taki piksel jest uznawany za czarny, w przeciwnym przypadku jego kolor zmieni się na biały.

## 1.2 Segmentacja tekstu (ang. Text Segmentation)

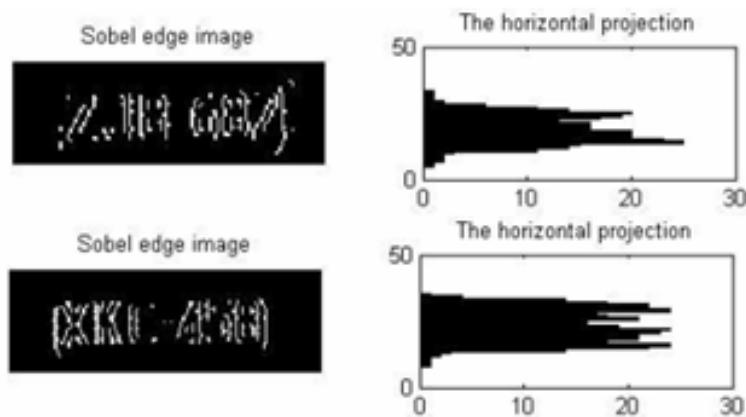
Zanim program zacznie rozpoznawać znaki, istotnie jest najpierw rozbić zlokalizowany na obrazie tekst na pojedyncze linie, a te na słowa oraz litery. Dla tego celu można użyć techniki histogramowej projekcji, która polega na wyliczaniu rzutów obrazu. Najpierw wykonuje się segmentacja linii tekstu, aby podzielić obszar tekstowy na pojedyncze linie. Odbywa się to za pomocą metody do wyliczania rzutu poziomowego (ang. horizontal projection).

Horizontal Histogram Projection). Ta metoda przyjmuje na wejście obraz binarny otrzymany z poprzedniego kroku, w którym białe pikseli - są uznawane za ważną informację, a czarne - za tło. Następnie oblicza się sumę białych pikseli w każdym wierszu obrazu (zob. Listing 1.1). Wynikiem działania metody jest lista o długości równej wysokości obrazu (zob. Rysunek 1.2). Jeśli wartość elementu listy jest duża pod względem pozostałych elementów, to znaczy, że jest to część linii tekstu. W przeciwnym wypadku, elementy o małej wartości mogą być uznane za obszary znajdujące się pomiędzy liniami tekstu i służyć jako wskaźniki na miejsca, gdzie należy podzielić obraz.

Listing 1.1. Wyliczanie rzutu poziomowego.

```

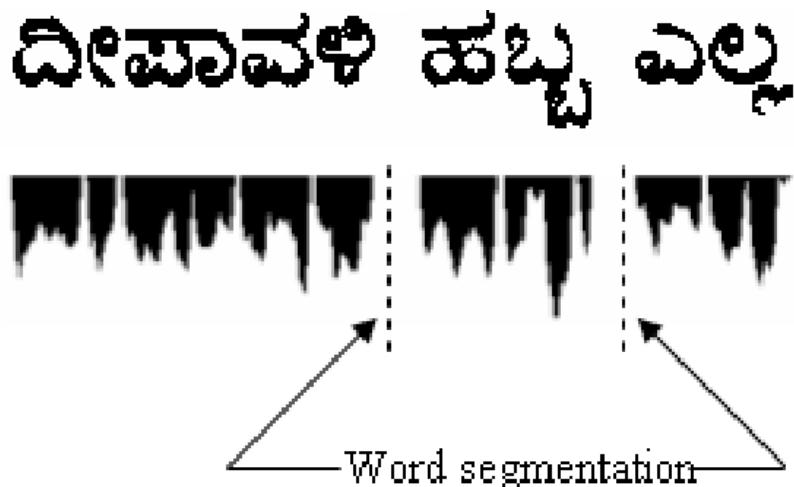
0 int[] getHorizontalHistogram (binaryImage) {
1     int[] imageRows = binaryImage.getRows();
2     int whitePixel = 255;
3     int[] separatorIndexArray = new int[binaryImage.getHeight()];
4     for(int i=0;i++; i < imageRows.size()){
5         int whitePixelSum = 0;
6         foreach(pixel in imageRows[i]){
7             if(pixel == whitePixel)
8                 whitePixelSum++;
9         }
10        separatorIndexArray[i]= whitePixelSum;
11    }
12    return separatorIndexArray;
13 }
```



Rysunek 1.2. Rzut poziomy obrazu. Źródło [2]

Kolejnym zadaniem jest segmentacja słów. Używa się do tego metoda wyliczania rzutu pionowego (ang. Vertical Histogram Projection), do której przekazywany jest wycięty

w poprzednim kroku obraz, zawierający pojedynczą linię tekstu. Jej działanie jest podobne do metody wykonywanej podczas segmentacji linii tekstu, jedyną różnicą jest to, że tutaj oblicza się sumę białych pikseli w każdej kolumnie obrazu a nie wierszu, jak to było wcześniej. Wynik przedstawia listę o długości równej szerokości obrazu, a jej elementy w zależności od posiadanej wartości mogą być częścią słów – jeśli są o dużej wartości lub polem pomiędzy nimi – w przypadku wartości niskich (zob. Rysunek 1.3). Odstęp pomiędzy słowami jest większy niż odstęp pomiędzy znakami, dlatego ważne jest ustalenie progu jego szerokości, na podstawie którego będą wybierane kolumny, rozdzielające słowa.



Rysunek 1.3 Rzut pionowy obrazu. Źródło [3]

Proces segmentacji znaków zależy od typu tekstu na obrazie. Jeśli znaki tworzące słowo są niezależne i nie łączą się pomiędzy sobą, to w takim wypadku używa się metoda z poprzedniego kroku segmentacji. W tym wypadku, wyliczanie pionowego rzutu odbywa się na podstawie obrazu, który zawiera pojedyncze słowo, składające się z ciągu znaków. Na wyjściu zwracana jest lista, z której należy wybrać kolumny o niskich wartościach, będące separatorami. W innym przypadku, jeżeli litery w tekście są połączone przez ligaturę, tak jak to często jest w piśmie ręcznym, to wtedy proces segmentacji słów jest wykonywany przy pomocy metod uczenia maszynowego.

### 1.3 Rozpoznawanie znaków (ang. Character Recognition)

Jak tylko wszystkie potencjalne znaki znajdujące się na obrazie zostaną wyodrębnione, silnik zawarty w oprogramowaniu OCR używa zaimplementowanej techniki do ich identyfikacji. Pierwsze programy stworzone do celów OCR stosowały technikę opartą o rozpoznawanie wzorów. W dzisiejszych czasach do rozpoznawania tekstu z obrazu są używane różne techniki.

### 1.3.1 Rozpoznawanie wzorców (ang. Pattern Recognition)

Rozpoznawanie wzorców polega na porównywaniu kształtów znajdujących się na obrazie z zestawem glifów w pamięci programu. Stosowanie tej techniki zaczęło się w latach 60-tych XX wieku do rozpoznawania dokumentów typu bankowych rachunków. Wtedy została stworzona czcionka o nazwie „OCR-A”, którą była używana podczas drukowania takich dokumentów. Zawarte w niej znaki można było łatwo rozróżnić, a ich szerokość była jednakowa. To spowodowało, że współczynnik rozpoznawania tekstu dla oprogramowania w tamtych czasach wzrósł niemal do 100%, a jego praca nie zajmowała dużo czasu. Nastepnym krokiem było dodanie do systemów OCR kolejnych wzorów do rozpoznawania innych popularnych czcionek.

Ta technika jest również znana jako dopasowywanie macierzy (ang. Matrix Matching). Ta nazwa lepiej opisuje jej działanie, ponieważ obraz przetwarzany w tym kroku jest macierzą, gdzie czarne pikseli reprezentują wartość 1, a białe oznakowane jako 0 (zob. Rysunek 1.4).



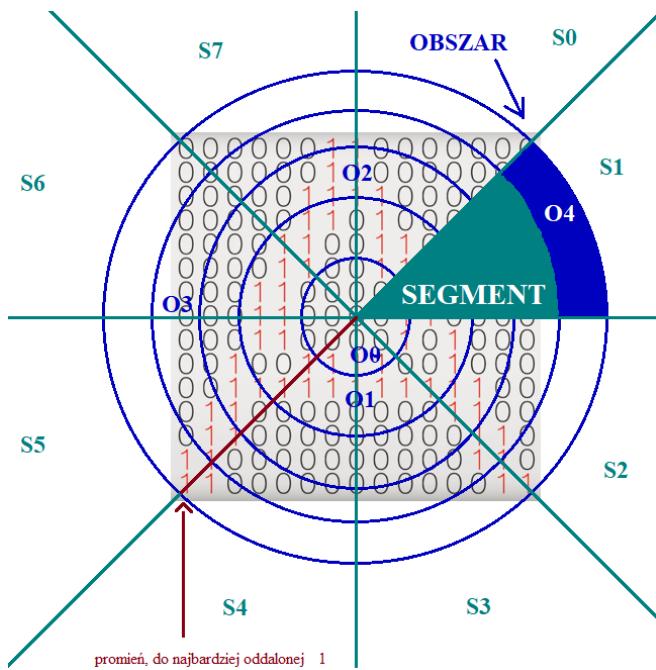
Rysunek 1.4. Binarna macierz litery A. Źródło [4]

Rozpoznawanie zaczyna się od znalezienia środka macierzy. Następnie wylicza się promień, od jej środka do najbardziej oddalonej od niego jedynki na podstawie wzoru (1.1).

$$dist = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \quad (1.1)$$

Na podstawie wyliczonej długości promienia, macierz dzieli się na 5 równych obszarów, a te z kolei są dzielone na 8 segmentów (zob. Rysunek 1.5). Końcowym wynikiem tego przetwarzania jest nowa macierz o wymiarach [8x5]. Jej wartości są wyliczane na podstawie ilości jedynek w każdym odcinku. Taki sam format mają szablony znajdujące się w pamięci programu. Końcowym etapem jest porównywanie wyniku z obecnymi w

programie szablonami. Znak oznacza się jako zidentyfikowany wtedy, gdy parametry porównywalnych macierze są do siebie podobne.



Rysunek 1.5. Dzielenie macierzy binarnej na obszary i segmenty. Źródło: opracowanie własne

### 1.3.2 Rozpoznawanie cech (ang. Feature Detection)

Rozpoznawanie cech polega na klasyfikacji kształtów znajdujących się na obrazie, a następnie porównywaniu ich z zaimplementowanymi w programie określonymi cechami każdego ze znaków. Rozwiążanie to opiera się na geometrycznych i topologicznych właściwościach znaku, co pozwala na rozpoznawanie każdego typu tekstu, niezależnie od kroju pisma, jego rozmiaru i bez znaczenia czy jest on wytłuszczone czy pochylony. Podczas identyfikacji znaków biorą się pod uwagę taki właściwości jak np.:

- proporcje wysokości i szerokości znaku,
- ilość miejsc, gdzie linii przecinają się (X),
- ilość pętli (O),
- punkty rozgałęzienia (T),
- punkty przegięcia,
- kreski oraz ich kierunek,
- krzywe oraz ich położenie.

Na ten moment istnieje olbrzymia ilość stosowanych czcionek i ta liczba ciągle się zwiększa. Dodatkowo na wejście do systemu OCR często trafiają teksty napisane pismem

ręcznym. Dlatego w nowoczesnych programach OCR, technika rozpoznawania wzorców jest rzadko implementowana.

## 1.4 Przetwarzanie końcowe (ang. Post-processing)

Wynikiem pracy poprzedniego etapu jest rozpoznany tekst. Jednak niektóre znaki w nim mogą zostać niepoprawnie rozpoznane. Powodem może być podobieństwo niektórych znaków do siebie, jak na przykład litera „O” czasami się myli z cyfrą „0” (zero) lub litera „I” może być rozpoznana jako litera „l” czy też jako cyfra „1” (jeden). Aby wyeliminować takie pomyłki, ten tekst poddaje się przetwarzaniu końcowemu, podczas którego program uwzględnia to, w jakim miejscu na obrazie muszą znajdować się cyfry, a gdzie słowa. Dodatkowo, w programie umieszczone są właściwe dla każdego języka słowniki, według których sprawdzane jest czy rozpoznane słowo jest poprawne.

## 1.5 Tesseract

Aby rozwiązać problem optycznego rozpoznawania znaków we własnej aplikacji, można użyć dowolnego z obecnie dostępnych na rynku oprogramowania OCR. Wybranie odpowiedniego produktu zależy w pierwszej kolejności od tego, czy można go dostosować do rozwiązywania konkretnego problemu. Jeżeli aplikacja ma umieć rozpoznawać tekst napisany pismem ręcznym w języku polskim, należy wybrać taki system, za pomocą którego można to zrobić. Innym czynnikiem, który należy wziąć pod uwagę jest kompatybilność z własną aplikacją. Dodatkowo warto wybrać taki produkt, który jest ciągle rozwijany.

Do rozwiązania problemu swojej pracy licencjackiej użyłem pakietu Tesseract [5]. Jest to ciągle rozwijane oprogramowanie OCR udostępniane na licencji open source, stworzone w latach 1980 przez Hewlett-Packard, a od 2006 roku jest sponsorowane przez Google. Jego implementacja była napisana w języku C++, co pozwalało na jego użycie jako API w innych aplikacjach. Do pracy z oprogramowaniem można użyć wiersza poleceń dostępnego w takich systemach operacyjnych jak Windows, Linux, macOS i Android.

Pierwotnie Tesseract umiał rozpoznawać tylko język angielski, a z czasem został rozbudowany o funkcjonalność do rozpoznawania takich języków jak francuski, niemiecki, włoski, hiszpański i holenderski. Obecna (oficjalna) wersja Tesseract (4.1.1) potrafi rozpoznać ponad 100 języków, w tym języki, w których teksty się piszą od prawej strony do lewej. Dodatkowo jest możliwość nauczenia go rozpoznawania innych języków, a także pisma ręcznego. Do pracy z obrazami i ich wstępного przetwarzania, oprogramowanie

używa zewnętrzną bibliotekę Leptonika [6], która po rozpoznawaniu znaków udostępnia takie formaty wyjściowe jak: zwykły tekst, hOCR (HTML for OCR), PDF i TSV.

Od wersji 4.0.0 Tesseract zawiera dodatkowy silnik do rozpoznawania tekstów, oparty na sieci neuronowej typu LSTM (ang. Long Short Term Memory OCR Engine). Jego skuteczność rozpoznawania jest większa niż poprzedniego silnika, a dodatkowo on ma możliwość uczenia się. Silnik OCR używany w wersjach 3.0.0, nazywany silnikiem klasycznym (ang. Legacy OCR Engine), działa na podstawie rozpoznawania wzorców i również może być trenowany.

## Rozdział 2. Sieci neuronowe w rozpoznawaniu znaków

### 2.1 Struktura sieci neuronowej

Jak wskazuje sama nazwa, sieć neuronowa jest zbiorem połączonych neuronów. Taki zbiór może zawierać od kilkuset do kilku tysięcy elementów tworzących sieć za pomocą powiązań zwanych wagami. Neuronem można nazwać obiekt przechowujący jakąś wartość, a wagą jest liczba określająca powiązanie dwóch neuronów [7]. Elementy sieci neuronowej są zgrupowane i podzielone na warstwy, z których ona się składa. Każdy neuron pierwszej warstwy, jest połączony z elementami drugiego, a te z kolei połączone z elementami trzeciego i tak do ostatniej warstwy. Ilość warstw może być różna, w zależności od implementacji, jednak ważnym jest ich przeznaczenie. Pierwsza warstwa każdej sieci neuronowej jest zdefiniowana jako wejściowa. Wartości neuronów w niej, są ustawiane według tego, co zostało w danym kroku przekazano do sieci. Kolejne warstwy poza ostatnią, są warstwami ukrytymi i przeznaczone do rozwiązywania problemu. Wynik ich pracy, przedstawia się w postaci wartości elementów ostatniej warstwy (inaczej warstwy wyjściowej). Ogólne, sieć może składać się nawet z trzech warstw, jednak użycie takiej implementacji jest sensowne tylko podczas rozwiązywania nieskomplikowanych problemów [8].

### 2.2 Działanie sieci neuronowej

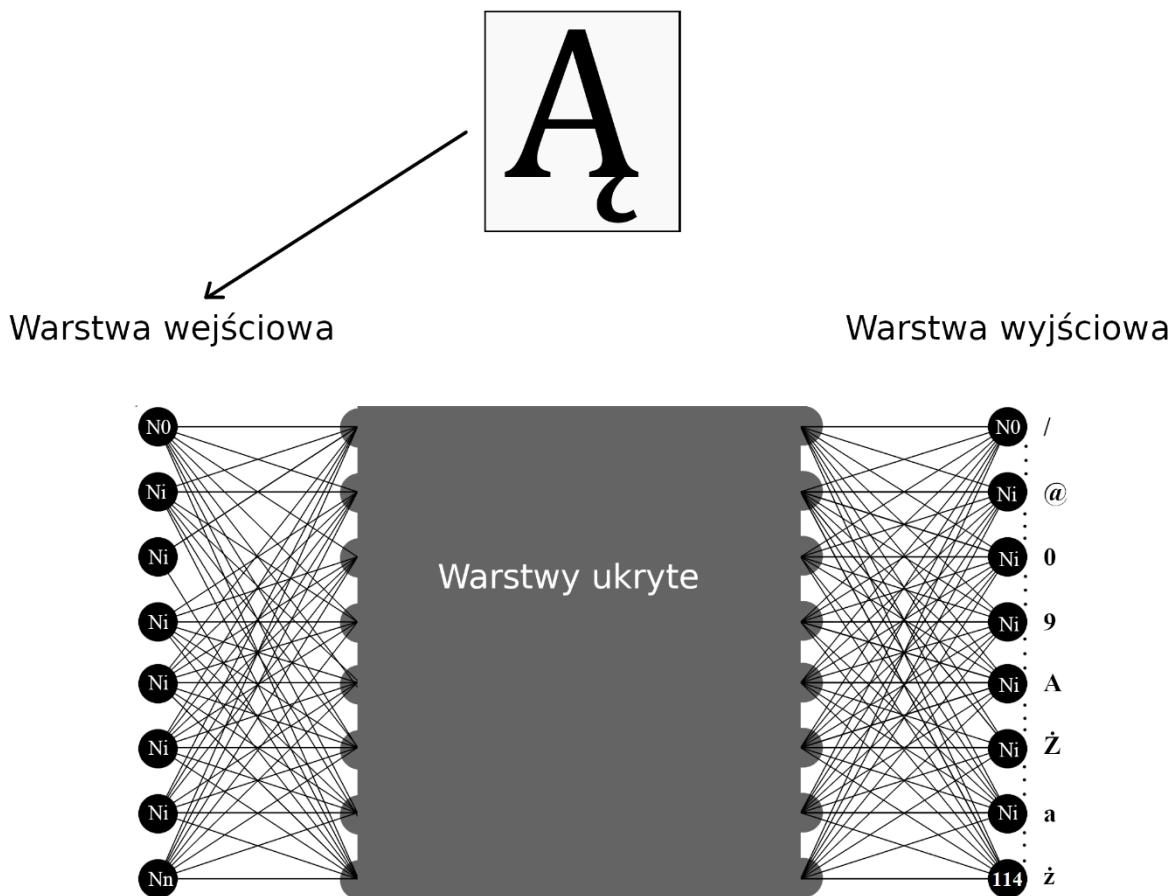
Ilość neuronów w każdej warstwie jest uzależniona od implementacji i od rozwiązywanego problemu. W przypadku rozpoznawania znaków, warstwa wejściowa będzie zawierała tyle samo elementów, ile znajduje się pikseli na obrazie przekazanym do sieci neuronowej. Warstwa wyjściowa wtedy będzie informowała, na ile obraz wejściowy jest podobny do każdego z możliwych znaków, a każdy neuron w niej, byłby odpowiednikiem konkretnego znaku [9]. Dla przykładu języka polskiego, alfabet, który zawiera też znaki diakrytyczne (tj.: ą, ć, ę, ł, ñ, ó, ś, ž, ż), warstwa ostatnia składałaby się z 114 elementów (zob. Rysunek 2.1).

W przypadku przekazywanych pikseli do pierwszej warstwy, neurony mogą reprezentować ich poziom szarości. Białe piksele będące tłem w postaci wartości 0, czarne jako 1, a wartości pomiędzy nimi są pikseli szare. Można wtedy określić, że wszystkie wartości bliskie zera nie są istotne, ponieważ nie są częścią znaku. W neuronie, jest to ustalane za pomocą bloku aktywacji. Ten blok ustala stan aktywacji neuronu. Neurony mogą

posiądać wartości w różnych zakresach, dlatego blok aktywacji normalizuje ich, aby wartość każdego neuronu mieściła się w zakresie zdefiniowanym. Dzięki temu można ustawić przedział, po przekroczeniu którego neuron staje się aktywnym. Aktywne neurony (w tym przypadku: czarne pikseli reprezentujące znak) są istotniejsze podczas wyliczania wyniku.

Stosuje się do tego funkcję aktywacji, którą może być jedna z poniższych:

- funkcja sigmoidalna,
- tangens hiperboliczny,
- sinusoida i cosinusoida,
- funkcja liniowa,
- funkcja Gaussa.



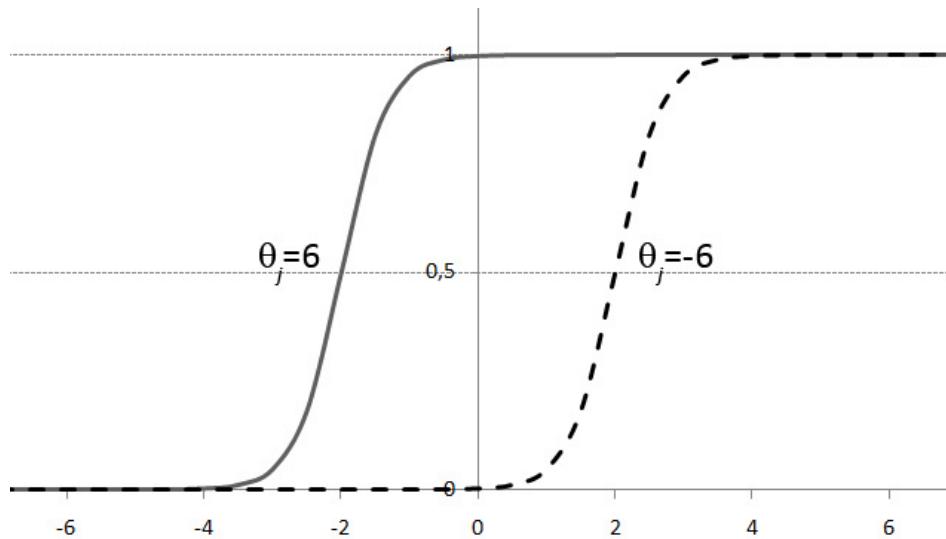
Rysunek 2.1. Przykładowy model sieci neuronowej. Źródło: opracowanie własne

Dobór odpowiedniej funkcji, zależy od tego jaki problem rozwiązuje sieć neuronów. Często stosuje się funkcję sigmoid [10]. Jej wynikiem jest wartość w przedziale od 0 do 1, która jest wyliczana według wzoru (2.1).

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2.1)$$

Neuron aktywuje się w przypadku, gdy wartość jest większa od zera. Na podstawie tego, wszystkie sygnały wejściowe, reprezentujące piksele białe, zostaną zignorowane w procesie rozpoznawania znaku.

Czasami, do rozwiązania stawianego przed siecią problemu, niezbędne jest, aby aktywacja neuronu odbywała się po przekroczeniu innej wartości niż zero. Próg aktywacji można przesunąć, uwzględniając w funkcji obciążenie (ang. *bias*). Ten składnik jest dodawany do wartości przekazanej do funkcji, tym samym przesuwając próg wzdłuż osi X. Ujemny *bias*, spowoduje przesunięcie w prawo, a dodatni w lewo (zob. Rysunek 2.2).

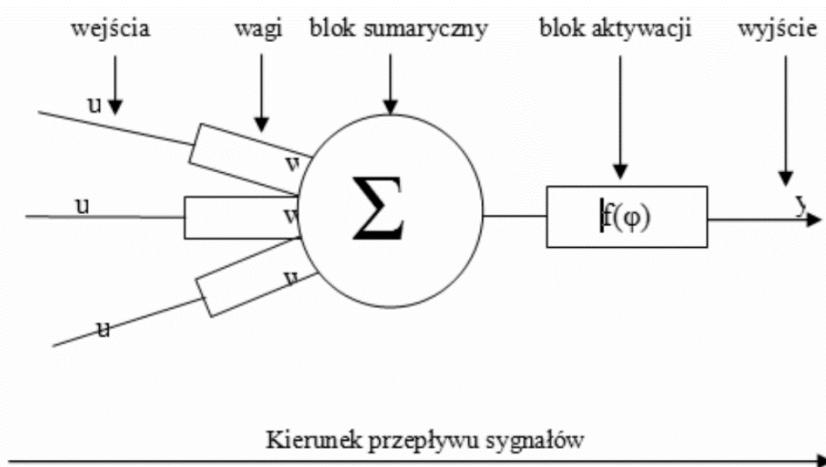


Rysunek 2.2. Przesunięcie progu aktywacji funkcji sigmoid, według bias. Źródło [11]

W warstwach ukrytych i warstwie wyjściowej, wartość neuronu jest wyliczana na podstawie sygnałów z warstwy poprzedzającej. Siła sygnałów jest wyrażona wagami, przez które odbywa się połączenie neuronów. Wagi zerowe oznaczają, że powiązania pomiędzy elementami nie ma. Sygnały wejściowe neuronu, są sumowane uwzględniając wartości wag, przez które one są mnożone. Odbywa się to w bloku sumarycznym, który z kolei przekazuje wyliczoną wartość do bloku aktywacji, a ten przekazuje przetworzony sygnał do neuronów kolejnego warstwa [12] (zob. Rysunek 2.3). Uwzględniając *bias* jako wejście do neuronu oraz jego wagę o wartości jeden, wyliczanie sygnału wyjściowego ‘y’ można opisać wzorem (2.2):

$$y = f \left( \sum_{i=0}^n W_i u_i \right) \quad (2.2)$$

Gdzie:  $f$  jest funkcją aktywacji,  $\langle u_0, u_1 \dots u_n \rangle$  są wejściami do neuronu, a  $\langle W_1, W_0 \dots W_n \rangle$  to wagi.



Rysunek 2.3 Model neuronu warstwy ukrytej. Źródło [13]

### 2.3 Uczenie sieci neuronowej

Rezultatem działania sieci neuronowej jest aktywność elementów warstwy ostatniej. Na stopień aktywności neuronów wpływają wagi, przekazujące sygnały z warstwy poprzedzającej. Na etapie implementacji sieci neuronowej, wartości wag są ustawiane losowo. To powoduje, że podczas jej działania, w neuronach błędnie się wylicza poziom ich aktywności. Proces dobierania odpowiednich wag dla sieci neuronowej, przy których jej wartości wynikowe są przybliżone do oczekiwanych, nazywany jest uczeniem nadzorowanym (bądź z nauczycielem). W tej metodzie uczenia do sieci przekazują się dwa zbiory: sygnały wejściowe i oczekiwane sygnały wyjściowe. Na podstawie wyliczonych i oczekiwanych wartości warstwy ostatniej, liczona jest różnica pomiędzy nimi. Zadaniem algorytmu uczenia, jest uregulowanie wag tak, aby ta różnica była jak najmniejsza. Przykładem takiego algorytmu uczenia jest *algorytm wstecznej propagacji błędów* (ang. Back Propagation), należący do gradientowych metod optymalizacji [14].

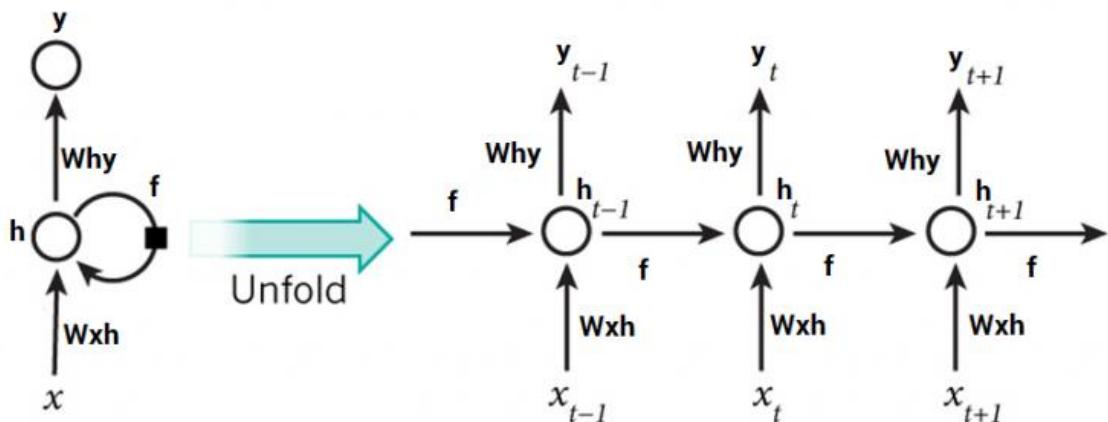
Do celów uczenia stosuje się różne metody, wybór których zależy od rozwiązywanego problemu. Pomimo uczenia z nauczycielem, istnieją też wiele algorytmów uczenia bez nauczyciela (lub uczenie samoorganizujące), podczas których sieć sama wypracowuje funkcję przetwarzania sygnałów [10].

### 2.4 Długoterminowa pamięć krótkoterminowa (ang. Long short-term memory)

W trakcie rozpoznawania tekstu, znaki w nim mogą być połączone ligaturami. To może spowodować, że na etapie segmentacji, słowo zostanie podzielone nie na pojedyncze

symbole, a na losowe części (kawałki znaków). Użycie zwykłej sieci neuronowej nie rozwiąże tego problemu, ponieważ aby zidentyfikować znak należały na wejście sieci, naraz przekazać kilka obrazów, które ten znak reprezentują. Pomocnicze w takim przypadku będą sieci rekurencyjne.

Sieci rekurencyjne RNN (ang. Recurrent Neural Network) stosuje się do rozwiązywania problemów, w których należy wziąć pod uwagę sekwencję danych wejściowych, a nie jeden konkretny element. Cechą wyróżniającą taką strukturę, jest duża ilość sprzężeń zwrotnych, które mogą łączyć wyjście neuronu z wejściem do innych warstw poprzedzających lub tego samego [10]. Każdy element, będący wektorem w sekwencji danych wejściowych, jest przekazywany do sieci w okresie czasowym ' $t$ ', którego wartość maksymalna jest równa ilości elementów w zbiorze wejściowym. Z tego wynika, że wektor ' $x$ ' określający obecne wejście, można oznakować jako ' $x_t$ '. Natomiast wektor ' $x_{t-1}$ ', opisywałby poprzedni element ze zbioru wejściowego, a ' $x_{t+1}$ ' - następny. W procesie działania RNN, wektory są przekazywane do warstwy ukrytej, uwzględniając przy tym wartości wag i wektory będące wynikiem poprzedniego kroku ' $h_{t-1}$ '. Rezultat pracy każdego kroku ' $h$ ', jest przekazywany na wyjście ' $y$ ' i jednocześnie na wejście następnego kroku ' $h_{t+1}$ '. Opisane działanie sieci RNN, jest zwizualizowane na Rysunku 2.4.

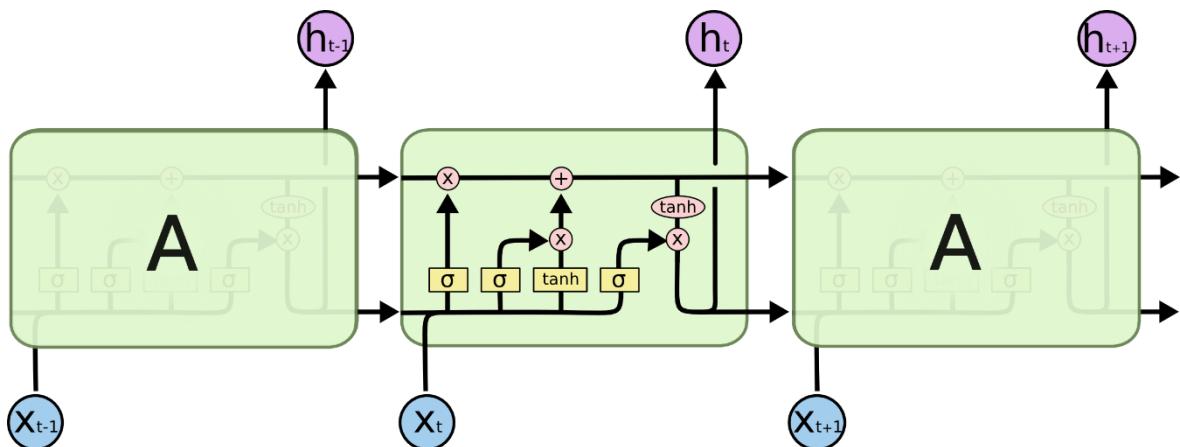


Rysunek 2.4. Rozwinęta rekurencyjna sieć neuronowa. Źródło [15]

Uczenie rekurencyjnej sieci neuronowej, tak samo jak i w zwykłych sieciach, wykonywane jest przy pomocy algorytmów wstępnej propagacji. Ich zadaniem jest zmodyfikowanie wartości wag każdego kroku tak, aby różnica błędu była jak najmniejsza. W trakcie aktualizacji wag, może powstać problem zanikającego gradientu (ang. Vanishing Gradient). Jest to problem najczęściej pojawiający się w sytuacjach dużej ilości elementów zbioru wejściowego. Podczas wyliczania nowych wartości wag, uwzględniane są wyniki kroków następnych, co w rezultacie powoduje to, że wagi łączące pierwsze wektory zbioru wejściowego, podlegają minimalnym zmianom. Zadaniem uczenia, jest znalezienie

każdemu wektorowi takich minimalnych wartości, przy których sieć neuronowa lepiej rozwiązuje postawiony problem. Jeżeli postępy w odnajdywaniu minimum dla wagi są nieznaczne, to czas nauczenia takiej sieci może się wydłużyć do nieskończoności. Rozwiązanie tego problemu leży w architekturze ulepszzonego modelu RNN, którym jest sieć LSTM.

W sieciach LSTM ważność przekazywanych danych jest kontrolowana. Jeżeli okaże się, że jakieś dane z poprzednich kroków są nieistotne podczas rozwiązywania postanowionego problemu, to sieć LSTM te dane zapomni, natomiast ważną informację z wektorów wejściowych, sieć zapisze w pamięci. Zwykła sieć rekurencyjna też zawiera pamięć, są to dane przekazywane pomiędzy krokami. Przy dużym zbiorze danych wejściowych, rezultat działania pierwszych kroków zanika, co oznacza, że pamięć ta jest krótkoterminowa [16]. Sieć LSTM, jak brzmi sama nazwa, zawiera dodatkowo pamięć długoterminową, która zarządza przepływem danych. Dodanie takiej pamięci wymaga uwzględnienia w architekturze sieci trzech warstw (zob. Rysunek 2.5).



Rysunek 2.5. Budowa architektury LSTM. Źródło [17]

W pierwszej kolejności, przed przekazywaniem danych do każdej z warstw, odbywa się konkatenacja wektora wejściowego wraz z wektorem wyjściowym poprzedniego kroku. Otrzymany wynik jest mnożony przez wagi obu wektorów i przekazywany na wejście do każdej warstwy. Pamięcią długoterminową też jest wektor, który jest przekazywany w każdym kroku czasowym.

Warstwa bramki zapomnienia (ang. Forget gate layer) decyduje o tym, jakie dane należy wymazać z pamięci długoterminowej. Przekazane do tej warstwy wejście, trafi do funkcji *sigmoid*, która zwraca wektor z wartościami w przedziale  $[0,1]$ . Wartości bliskie zeru są elementami, które należy usunąć z pamięci. W tym celu otrzymany wektor jest mnożony przez wektor stanu pamięci [18].

Warstwa bramki wejściowej (ang. Input gate layer) decyduje o tym, jakie dane wejściowe należy dodać do pamięci długoterminowej. W pierwszej kolejności przy pomocy funkcji *sigmoid*, wybierane są elementy, które zostaną zaktualizowane. Funkcja *sigmoid* jest używana, aby określić jak dużo informacji każdego elementu wektora wejściowego musi przejść. W przypadku jedynki, oznacza, że wartość elementu, zostanie w całości przekazana do pamięci, a jeżeli zero, to nic. Do określenia przekazywanej wartości, używa się funkcja aktywacji *tangens hiperboliczny*. Następnie, obydwa wektory są mnożone przez siebie, co w rezultacie daje wektor zawierający nową informację. W kolejnym kroku stan pamięci długoterminowej uzupełnia się nowym wektorem [17].

Warstwa bramki wyjścia (ang. Output gate layer) - jest końcowym etapem przepływu danych, w którym uwzględnia się pracę dwóch poprzednich warstw. Nowy stan pamięci długoterminowej jest używany w celu wyliczenia wyniku na podstawie wektora, przekazanego w danym kroku czasowym. Wektor pamięci może zawierać elementy różnych wartości. Dlatego jest on przekazywany do funkcji *tangens hiperboliczny*, która scal i te wartości w przedziale [-1, +1]. Następnie, należy zdefiniować jakie dane z pamięci są niezbędne do rozwiązania problemu tego kroku. W tym celu, dane wejściowe tej warstwy są przekazywane do funkcji *sigmoid*, a następnie są mnożone przez stworzony na podstawie pamięci wektor. Wynik jest przekazywany na wyjście oraz do następnego kroku [15].

Pamięć długoterminowa jest przekazywana pomiędzy krokami. W każdym z nich, jest aktualizowana na podstawie wektora wejściowego, przekazywanego w tym kroku.

# Rozdział 3. Aplikacja do rozpoznawania tekstu na paragonie fiskalnym

## 3.1 Opis aplikacji

W swoim rozwiązaniu tego problemu zakładam potrzebę przeniesienia danych, znajdujących się na paragonie, do systemu komputerowego za pomocą kamery lub skanera, a następnie przetworzenia ich do postaci tekstowej za pomocą oprogramowania OCR. Podczas druku dokumentu, urządzenie fiskalne używa unikalnej czcionki stworzonej do tego celu. Aby skuteczność rozpoznawania tekstu była wysoka, system OCR musi być zapoznany z tym, jakie kształty znaków mogą się pojawić na obrazie. Dlatego przed implementacją aplikacji, należy przeprowadzić proces nazywany “uczeniem” i przekazać do systemu OCR nową czcionkę jako zbiór uczący.

## 3.2 Proces uczenia Tesseract

Rozpoznawanie tekstu oprogramowaniem Tesseract, odbywa się na podstawie pliku o rozszerzeniu „traineddata”, który należy wcześniej wskazać. Tego typu pliki zawierają zestaw przygotowanych dla konkretnego wypadku danych, używanych silnikiem OCR. Na oficjalnej stronie Tesseract [19] dostępne są pliki dla większości języków, wśród których jest plik używany do rozpoznawania języka polskiego. Ma on nazwę “pol.traineddata”. Od wersji 4.00 te pliki zawierają w sobie model sieci neuronowej LSTM za pomocą, której odbywa się rozpoznawanie. Oprogramowanie ma funkcjonalność, na podstawie której dobrany model LSTM można dociążyć nowymi danymi. Według oficjalnej instrukcji [20] dotyczącej uczenia Tesseract, proces ten najlepiej przeprowadzać na maszynach z zainstalowanym systemem typu Linux. Dla szybszej pracy, zaleca się mieć procesor wielordzeniowy.

Uczenie należy rozpocząć od przygotowania danych, na podstawie których będzie odbywał się ten proces. W tym celu należy przekazać plik czcionki używanej do druku paragonów na wejście do skryptu „tesstrain.sh” (pobierany ze strony Tesseract [21]), na podstawie którego wygeneruje się gotowy zestaw danych do uczenia. Możliwym problemem jest to, że może nie być niezbędnego pliku czcionki, jednak Tesseract posiada funkcjonalność, dzięki której można samemu wygenerować zestaw uczący na podstawie zwykłego obrazu z tekstem. Kolejnym problemem jest to, że czcionka może się różnić w

zależności od sklepu i oprogramowania zainstalowanego na urządzeniu fiskalnym. W swojej pracy licencjackiej przygotuję rozwiązywanie tylko dla czcionki pochodzącej z jednej sieci handlowej, ponieważ uwzględnienie innych wymaga dużej ilości danych do uczenia. Wybrałem do tego celu hipermarket Auchan z tego względu, że oferuje dużą ilość artykułów do sprzedania i dzięki temu kształty znaków można spotkać na paragonie, połączone w różnych kombinacjach.

Przygotowane przeze mnie dane są uzyskane za pomocą skanera. Są to obrazy paragonów ze sklepu Auchan w postaci plików o rozszerzeniu „png”. Podczas skanowania, za pomocą właściwego oprogramowania, wyodrębniałem miejsca, gdzie znajdowały się dokumenty na szybie urządzenia, aby w wyniku otrzymać tylko same obrazy paragonów bez tła. Przykładowy obraz paragonu można zobaczyć na Rysunku 3.1.

Aby poprawić jakość procesu rozpoznawania przeprowadzanego przez Tesseract, twórcy sugerują wstępne przetworzenie i przygotowanie danych. Podczas rozpoznawania należy stosować tą samą metodykę przetwarzania, co i dla danych uczących. Stworzone przeze mnie algorytmy, poprawiające obraz, będą opisane w dalszej części rozdziału, ponieważ są używane też podczas działania aplikacji. Dla lepszego rezultatu rozpoznawania, postanowiłem zaimplementować własny algorytm, który dzieli paragon na pojedyncze obrazy każdego wiersza. Jest on też częścią zaimplementowanej aplikacji i zostanie opisany w późniejszej części pracy. Dla procesu uczenia, zapisałem wycięte linie tekstu w formacie „png” i oznaczyłem liczbami w zależności od ich kolejności. Następnie, użyłem program „jTessBoxEditor” [22]. Posiada on funkcjonalność, która upraszcza proces uczenia Tesseract. Dzięki wbudowanym metodom scaliłem te obrazy w jeden plik formatu „tiff”. Według konwencji Tesseract, plik ten musi być nazwany zgodnie z wzorcem (3.1).

$$(3.1) \quad [\text{skrót języka}].[\text{nazwa czcionki}].\text{exp}[\text{numer pliku}].[\text{rozszerzenie pliku}]$$

Zbiór przygotowanych przeze mnie danych był zebrany na podstawie 11 paragonów. Podsumowując liczbę stron w każdym z nich, wynika, że do trenowania zostanie przekazane 349 linii tekstu w postaci obrazów.

Dodatkowo w celu uczenia, niezbędnie jest przekazanie do sieci LSTM oczekiwany rezultatu. Tesseract ma wbudowaną funkcjonalność, przy pomocy której dla wybranego obrazu, można wygenerować plik z informacją o znakach występujących na nim. W tym celu należy wywołać program tesseract, przekazując jako parametr operację lstmbox.:

```
tesseract -l pol --dpi 300 --oem 1 --psm 7 pol.auchan.exp4.tif pol.auchan.exp4 lstmbox
```



Rysunek 3.1. Zeskanowany paragon. Źródło: opracowanie własne

Do programu zostały przekazane takie parametry:

- 1 – wybór języka (ang. language), który należy rozpoznać. W tym przypadku jest to język polski,

- dpi – podanie rozdzielczości przekazywanego obrazu. Wartość DPI (ang. dots per inch), jest przekazywana, aby Tesseract zdefiniował rozmiar liter. Zeskanowane przez mnie obrazy posiadają wartość 300,
- oem – wybór silnika OCR którego należy użyć do rozpoznawania. Wartość jeden określa silnik oparty o sieci neuronowe LSTM,
- psm – wybór typu segmentacji obrazu. Wybrana wartość powoduje to, że obraz będzie traktowany jako jeden wiersz tekstu,
- pol.auchan.exp4.tif – ścieżka do obrazu, na podstawie którego wygeneruje się informacja,
- pol.auchan.exp4 – nazwa wyjściowego pliku. Musi być taka sama jak i nazwa obrazu,
- lstmbox – operacja użycie której wygeneruje plik z informacją o znakach na obrazie.

Wywołanie powyższego polecenia spowoduje, że Tesseract spróbuje zlokalizować i rozpoznać znaki znajdujące się na pliku wejściowym, a następnie odnotuje te dane w pliku tekstowym. Każdy znak wraz z koordynatami linii tekstu i numerem strony jest zapisywany w kolejnej linii pliku, według wzorca (3.2). Punkty X i Y są podawane według miejsca, w którym zaczyna się powiązana linia tekstu.

$$(3.2) \quad [znak] [punkt\ X] [punkt\ Y] [szerokość\ linii] [wysokość\ linii] [numer\ strony]$$

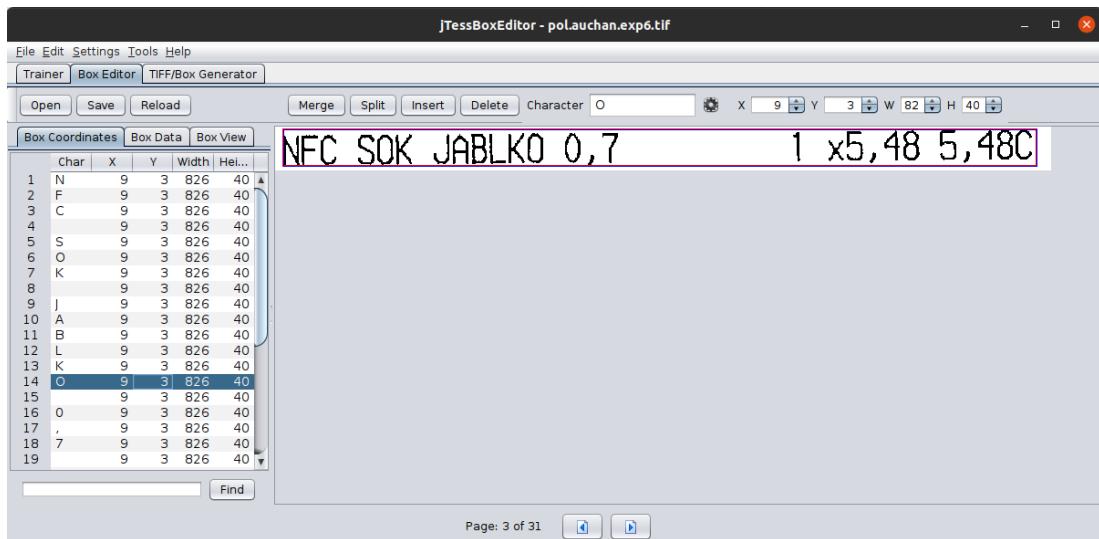
Z racji tego, że sieć jeszcze nie umie dobrze rozpoznawać danej czcionki, wygenerowany plik może zawierać błędy, które należy poprawić ręcznie. Wspomniany wcześniej program „jTessBoxEditor”, był stworzony do tego celu, aby umożliwić poprawianie współrzędnych za pośrednictwem graficznego interfejsu użytkownika (zob. Rysunek 3.2). Sprawdzić należy każdy znak, tak aby sieć neuronowa nie została nauczona fałszywymi danymi.

Ostatnim krokiem przygotowywania danych jest połączenie plików graficznych wraz z ich tekstowymi odpowiednikami. Ta kombinacja musi być przekazana jako zbiór uczący do Tesseract w postaci pliku o rozszerzeniu *lstmf*. Żeby wygenerować taki plik, przekazałem do programu tesseract podobne jak w poprzednim poleceniu parametry. Jedyną różnicą jest wywoływana operacja, którą w tym wypadku jest *lstm.train*:

```
tesseract -l pol --dpi 300 --oem 1 --psm 7 pol.auchan.exp4.tif pol.auchan.exp4 lstm.train
```

Trenowanie Tesseract odbywa się za pośrednictwem programu *lstmtraining*. Jest to wielofunkcyjne narzędzie, za pomocą którego można używać różnych metod uczenia. Tworzony przez mnie program jest przeznaczony do rozpoznawania języka polskiego. W tym celu używam pliku „pol.traineddata”. W przypadku, gdyby zawarty w nim model nie

umiał rozpoznawać jakiegoś symbolu tekstowego lub jak w przypadku rozwiązywanego przeze mnie problemu, nie radziłby z nową czcionką, to wtedy należy zastosować podejście nazywane *Fine Tuning*, które jest typem uczenia transferowego (ang. transfer learning). Jest to proces douczania nowymi danymi już istniejącej sieci, nie zmieniając jej struktury i warstw głębokich. Aby poprawić rozpoznawalność pobranego modelu wywoałem polecenie zamieszczone na Listingu 3.1.



Rysunek 3.2. Rzut ekranu aplikacji JTessBoxEditor. Źródło: opracowanie własne

Listing 3.1. Polecenie używane do trenowania Tesseract.

```

1 lstmtraining
2   --model_output checks/auchan
3   --continue_from polModel.lstm
4   --traineddata pol.traineddata
5   --train_listfile training_files.txt

```

Używane w nim parametry:

- model\_output – miejsce, do którego będzie zapisywany stan sieci podczas trenowania. Jako wartość podałem ścieżkę do stworzonego folderu o nazwie “checks” oraz nazwę modeli “auchan”,
- continue\_from – model sieci LSTM. Należy ją wyeksportować z pobranego wcześniej pliku pol.traineddata. Zrobiłem to przy pomocy kolejnego programu udostępnionego oprogramowaniem Tesseract, o nazwie “combine\_tessdata”. Program ten pozwala na różne działania związane z danymi zawartymi w pliku trainneddata. W celu eksportu,

przekazałem do niego parametr “-e”, ścieżkę do pliku i żądany element, którym jest model sieci o rozszerzeniu “lstm”:

```
combine_tessdata -e pol.traineddata pol.lstm
```

- traineddata – jest to wcześniej pobrany plik, zawierający model sieci lstm, listę liter i listę liczb, które rozpozna Tesseract, słownik słów oraz inne niezbędne do rozpoznawania dane,
- train\_listfile – plik, zawierający listę przygotowanych do uczenia plików, o rozszerzeniu “lstmf”.

Podczas uczenia, Tesseract zapisuje stan sieci w postaci plików, które w nazwie zawierają napis „checkpoint”. Są dwa typy takich plików:

- 1) *<nazwa\_modeli>\_checkpoint* – końcowy stan modeli sieci neuronowej,
- 2) *<nazwa\_modeli><zakowy\_wspolczynnik\_bledu>\_<iteracja>.checkpoint* – jest zapisywany każdorazowy, gdy wynik uczenia się polepsza.

Każdy z nich można przetworzyć do postaci pliku traineddata, używanego podczas rozpoznawania, za pomocą polecenia przedstawionego na Listingu 3.3.

Listing 3.2. Generowanie pliku traineddata.

```
1 lstmtraining  
2      --stop_training  
3      --continue_from checks/auchan_checkpoint  
4      --traineddata pol.traineddata  
5      --model_output auchan.traineddata
```

Przekazywanymi parametrami są:

- stop\_training – oznacza to, że sieć została wystarczająco nauczona i program musi wygenerować plik końcowy,
- continue\_from – stan sieci, na podstawie którego wygeneruje się plik końcowy. Przekazałem tu końcowy stan modeli w postaci pliku “auchan\_checkpoint”,
- traineddata – plik “pol.traineddata”, używany w procesie uczenia,
- model\_output – nazwa i ścieżka do pliku, który należy wygenerować. Z tego względu, że model ten jest przeznaczony do rozpoznawania czcionki używanej w sklepie Auchan, to podałem nazwę “auchan.traineddata”.

### 3.3 Implementacja aplikacji

Program do rozpoznawania paragonów jest aplikacją webową. Implementacja została napisana w języku Java w wersji 12, edycji ‘Enterprise edition’. W projekcie został użyty szkielet (ang. framework) do budowy aplikacji webowych Spring MVC. Szkielet używa wzorca projektowego ‘Model Widok Kontroler’ (ang. Model View Controller), w którym:

- widok, jest stroną przekazywaną do klienta,
- model, to obiekt, dane które zostaną zaprezentowane w widoku,
- kontroler, kontroluje jaki model, w którym widoku pokazać.

Do użycia Tesseract, do projektu została dodana biblioteka Tess4j. Jest to opakowanie (ang. Wrapper) JNA (Java Native Access), dzięki któremu można wywoływać oryginalne API Tesseract, używając kodu Java [15]. Do przetwarzania obrazów używa się biblioteki OpenCV. Wynik działania aplikacji, jest zapisywany do bazy danych MySql. Aplikacja działa na porcie 8080.

#### 3.3.1 OpenCVService

Implementacja klasy „OpenCVService” zawiera metody służące do przetwarzania obrazów. Wszystkie przetwarzania są wykonywane za pomocą algorytmów biblioteki OpenCV. Klasa zawiera jedną publiczną metodę o nazwie „getReceiptLines”, która przyjmuje na wejście obraz, a na wyjściu zwraca listę wyciętych, przetworzonych obrazów reprezentujących pojedyncze linii tekstu (zob. Listing 3.3).

Listing 3.3. Implementacja metody getReceiptLines

```
0 public List<BufferedImage> getReceiptLines (image) {  
1     preProcessedImage = preprocessImage (image);  
2     croppedLinesList = findLines(preProcessedImage);  
3     return croppedLinesList;  
4 }
```

Proces wycinania linii jest podzielony na dwa etapy. W pierwszym obraz jest przekazywany do metody „preProcessImage”, która jest przeznaczona do poprawienia jego jakości. W drugim, przetworzony obraz paragonu jest przekazywany na wejście metody o nazwie „findLines” która, dzieli go na pojedyncze wiersze. Idea tego algorytmu polega na tym, aby zwiększyć szerokość każdego występującego elementu na obrazie. To spowoduje,

że wszystkie znaki, pikseli, których znajdują się na tej samej osi y, zostaną połączone w jeden obiekt reprezentujący linie. W wyniku zwracana jest lista wyciętych fragmentów obrazu wejściowego.

Przetwarzanie obrazu (zob. Listing 3.4) zaczyna się od jego przekształcenia do skali szarości. Następnie do obrazu stosuje się bilateralny filtr, dzięki któremu ilość występujących szumów na nim, zmniejsza się. W linii nr. 3 odbywa się binaryzacja obrazu. W tym celu został użyty algorytm adaptacyjnego progowania (ang. Adaptive Threshold), który wylicza próg każdemu pikselu, według pikseli sąsiednich. W kolejnych dwóch liniach listingu jest wykonywane morfologiczne przekształcenie obiektów znajdujących się na obrazie. Najpierw operacja otwarcia (ang. Morphology Opening), usuwa zakłócenia obok liter. Następnie, aby poprawić kształty znaków, jest wykonywana operacja zamknięcia (ang. Morphology Closing). Wynik każdego kroku przetwarzania jest przedstawiony na Rysunku 3.3 i Rysunku 3.4.

Listing 3.4. Przetwarzanie obrazu

```
0 private void preProcessImage (image) {  
1     removeColors(image);  
2     bilateralFilter(image);  
3     adaptiveThreshold(image);  
4     morphologyOpen(image);  
5     morphologyClose(image);  
6 }
```

W procesie działania algorytmu wyszukiwania linii (zob. Listing 3.5), na obrazie zostaną same bloki reprezentujące linie. Aby zachować przekazany obraz, w pierwszej linii listingu tworzona jest kopia obrazu o nazwie „linesImage”, według której będą wyszukiwane wiersze. Tesseract traktuje białe pikseli jako tło, a czarne jako tekst [23]. W tym celu obraz wejściowy przekazuje się do operacji negacji bitowej (ang. Bitwise), która zamienia pikseli białe na czarne, a czarne na białe. W następnych liniach listingu przetwarza się kopia obrazu. W linii nr. 3 odbywa się przekształcenie morfologiczne, zwane erozją (ang. Erosion). Działanie algorytmu erozji polega na obcinaniu brzegów każdego obiektu, znajdującego się na obrazie. Jest on stosowany, aby zostawić na paragonie same znaki, a obiekty będące plamami lub brudem zmniejszyć lub całkowicie usunąć. Kolejnym użyтыm morfologicznym przekształceniem jest dylacja (ang. Dilation), które w przeciwieństwie do erozji – zwiększa brzegi obiektów [23]. Oprócz obrazu, do algorytmu dylacji zostały przekazane dwa parametry: wysokość o wartości 0 i szerokość która jest równa 40. To spowoduje, że

wysokość każdego obiektu na obrazie zostanie niezmienna, natomiast szerokość zwiększy się o 40 pikseli (o 20px z lewej strony obiektu i 20px z prawej). Metoda o nazwie „drawRectsOfObjects” rysuje na całą szerokość obrazu białe prostokąty. Ich wysokość i położenie są odpowiedni do każdego obiektu na obrazie. W ostatniej linii listingu jest wywoływana metoda „cropLinesAndSaveToList”, która na podstawie koordynat obiektów znajdujących się na obrazie „linesImage” wycina fragmenty z obrazu wejściowego „image”, a następnie zapisuje ich do zwracanej listy. Wyniki poszczególnych kroków zostały zaprezentowane na Rysunku 3.5.

Listing 3.5. Wyszukiwanie linii na obrazie.

```
0 private List<BufferedImage> findLines (image) {  
1     linesImage = image.clone();  
2     bitwise(image);  
3     erosion(linesImage);  
4     dilation(linesImage, 0, 40);  
5     drawRectsOfObjects(linesImage);  
6     return cropLinesAndSaveToList (linesImage, image);  
7 }
```

Tworzone metodą „drawRectsOfObjects” elementy (zob. Listing 3.6), są rysowane nad każdym obiektem. W pierwszej linii listingu przy pomocy algorytmu biblioteki OpenCV, odbywa się wyszukiwanie na obrazie konturów każdego występującego na nim obiektu. Następnie w linii nr. 4, dla każdego konturu tworzony jest niejawny otaczający go prostokąt. Zawiera on takie dane jak: koordynaty swojego górnego lewego punktu, szerokość i wysokość. W linii nr 9 została użyta metoda z biblioteki OpenCV za pomocą, której na wejściowym obrazie są rysowane białe prostokąty. Do niej zostały przekazane takie atrybuty:

- obraz wejściowy,
- startowy punkt na obrazie, aby określić położenie prostokąta. Podawane w nim koordynaty górnego lewego punktu rysowanego prostokąta. Według osi „x” została przekazana wartość „0” – jako początek obrazu, natomiast według osi „y”, przekazuje się koordynata „y” górnego punktu prostokąta konturu (linia nr. 5),
- punkt końcowy na obrazie, również do określenia pozycji prostokąta. Podawane w nim koordynaty dolnego prawego punktu rysowanego prostokąta. Według osi „x” została przekazana szerokość wejściowego obrazu, a według osi „y”, przekazuje się położenie dolnej krawędzi prostokąta konturu (linia nr. 6),

- kolor rysowanego prostokąta. Przekazany wektor odpowiada białemu koloru,
- grubość linii prostokąta. Wartość „-1”, oznacza, że cały obszar prostokąta zostanie pokolorowany.

Listing 3.6. Tworzenie elementów reprezentujących linii na paragonie

```

0 private void drawRectsOfObjects (image) {
1     contours = Imgproc.findContours(image, Imgproc.RETR_EXTERNAL,
2                                         Imgproc.CHAIN_APPROX_SIMPLE);
3     foreach (object in contours) {
4         rect = Imgproc.boundingRect(object);
5         startPoint = (0, rect.y);
6         endPoint = (image.width(), rect.y + rect.height );
7         color = (255, 255, 255);
8         thickness = -1;
9         Imgproc.rectangle(image, startPoint, endPoint, color, thickness);
10    }
11 }
```

Metoda do wycinania linii tekstu z obrazu (zob. Listing 3.7) przyjmuje na wejściu dwa atrybuty. Pierwszy przedstawia położenie linii, które należy wyciąć, a drugi przedstawia obraz paragonu, z którego te linii zostaną wycięty. W linii pierwszej odbywa się wyszukiwanie konturów każdego bloku reprezentującego linie. Następnie, otrzymana lista konturów jest sortowana według położenia każdego z nich na osi „y” (linii nr. 3-7). W linii nr 8 tworzona jest lista, która będzie przechowywać obrazy wyciętych linii paragonu. Według każdego obiektu w liście, z obrazu paragonu są wycinane prostokąty, a następnie zapisywane jako macierz pikseli (tzn. obiekt obrazu w bibliotece OpenCV) do zmiennej „croppedLine” (linia nr. 11). Aby wyśrodkować tekst na wyciętej linii, wokół niej tworzone jest białe obramowanie (linia nr. 16). Przekazanymi parametrami są:

- macierz wyciętej linii,
- szerokość obramowania z każdej strony. Dla górnego i dolnego obramowania podana wartość 10 piksel, a dla lewego i prawego 5 piksel,
- jako rodzaj obramowania została podana wartość „BORDER\_ISOLATED”. Oznacza to, że obramowanie zostanie stworzone nie wewnątrz, a zewnętrz obrazu i doda do niego nowe pikseli,
- kolor obramowania jest biały, reprezentujący tło.

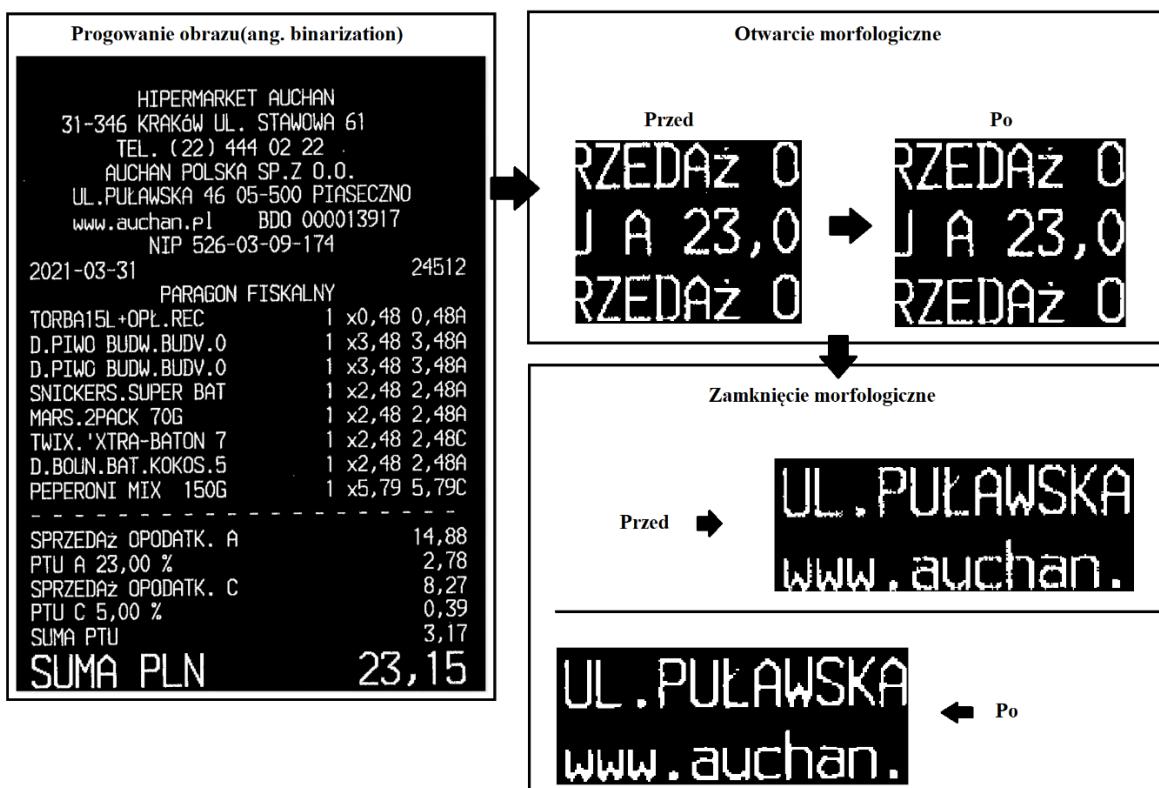
W linii nr. 18 gotowa macierz przekształca się do obiektu „BufferedImage”, a następnie dodaje się do zwracanej listy. Instancja tego obiektu używa się do rozpoznawania tekstu biblioteką Tesseract.

Listing 3.7. Wycinanie linii z obrazu

```
0 private List<BufferedImage> cropLinesAndSaveToList (linesImage, image) {  
1     contours = Imgproc.findContours(linesImage, Imgproc.EXTERNAL,  
2                                     Imgproc.CHAIN_APPROX_SIMPLE);  
3     contours.sort((object1, object2) -> {  
4         rect1 = Imgproc.boundingRect(object1);  
5         rect2 = Imgproc.boundingRect(object2);  
6         return Double.compare(rect1.y, rect2.y);  
7     });  
8     List<BufferedImage> lines = new ArrayList<>();  
9     foreach (object in contours) {  
10        rect = Imgproc.boundingRect(object);  
11        croppedLine = new Mat(image, rect);  
12        top = 10; bottom = top;  
13        left = 5; right = left;  
14        borderType = Core.BORDER_ISOLATED;  
15        color = (255, 255, 255);  
16        lineVsBorder = Core.copyMakeBorder(croppedLine, top, bottom, left,  
17                                             right, borderType, color);  
18        BufferedImage bufferedImage = mat2BufferedImage(lineVsBorder);  
19        lines.add(bufferedImage);  
20    }  
21    return lines;  
22 }
```



Rysunek 3.3. Przetwarzanie obrazu-1. Źródło: opracowanie własne



Rysunek 3.4. Przetwarzanie obrazu-2. Źródło: opracowanie własne

Obraz wejściowy "image"		Erozja(ang. Erosion)	
<b>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO <a href="http://www.auchan.pl">www.auchan.pl</a> BDO 000013917 NIP 526-03-09-174</b>		<b>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO <a href="http://www.auchan.pl">www.auchan.pl</a> BDO 000013917 NIP 526-03-09-174</b>	
2021-03-31	24512	2021-03-31	24512
PARAGON FISKALNY		PARAGON FISKALNY	
TORBA15L+OPŁ.REC	1 x0,48 0,48A	TORBA15L+OPŁ.REC	1 x0,48 0,48A
D.PIWO BUDW.BUDV.0	1 x3,48 3,48A	D.PIWO BUDW.BUDV.0	1 x3,48 3,48A
D.PIWC BUDW.BUDV.0	1 x3,48 3,48A	D.PIWC BUDW.BUDV.0	1 x3,48 3,48A
SNICKERS.SUPER BAT	1 x2,48 2,48A	SNICKERS.SUPER BAT	1 x2,48 2,48A
MARS.2PACK 70G	1 x2,48 2,48A	MARS.2PACK 70G	1 x2,48 2,48A
TWIX.'XTRA-BATON 7	1 x2,48 2,48C	TWIX.'XTRA BATON 7	1 x2,48 2,48C
D.BOUN.BAT.KOKOS.5	1 x2,48 2,48A	D.BOUN.BAT.KOKOS.5	1 x2,48 2,48A
PEPERONI MIX 150G	1 x5,79 5,79C	PEPERONI MIX 150G	1 x5,79 5,79C
-----	-----	-----	-----
SPRZEDAŻ OPODATK. A	14,88	SPRZEDAŻ OPODATK. A	14,88
PTU A 23,00 %	2,78	PTU A 23,00 %	2,78
SPRZEDAŻ OPODATK. C	8,27	SPRZEDAŻ OPODATK. C	8,27
PTU C 5,00 %	0,39	PTU C 5,00 %	0,39
SUMA PTU	3,17	SUMA PTU	3,17
<b>SUMA PLN</b>	<b>23,15</b>	<b>SUMA PLN</b>	<b>23,15</b>

Dylacja(ang. Dilation)		Negacja bitowa obrazu "image"	
<b>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO <a href="http://www.auchan.pl">www.auchan.pl</a> BDO 000013917 NIP 526-03-09-174</b>		<b>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO <a href="http://www.auchan.pl">www.auchan.pl</a> BDO 000013917 NIP 526-03-09-174</b>	
2021-03-31	24512	2021-03-31	24512
PARAGON FISKALNY		PARAGON FISKALNY	
TORBA15L+OPŁ.REC	1 x0,48 0,48A	TORBA15L+OPŁ.REC	1 x0,48 0,48A
D.PIWO BUDW.BUDV.0	1 x3,48 3,48A	D.PIWO BUDW.BUDV.0	1 x3,48 3,48A
D.PIWC BUDW.BUDV.0	1 x3,48 3,48A	D.PIWC BUDW.BUDV.0	1 x3,48 3,48A
SNICKERS.SUPER BAT	1 x2,48 2,48A	SNICKERS.SUPER BAT	1 x2,48 2,48A
MARS.2PACK 70G	1 x2,48 2,48A	MARS.2PACK 70G	1 x2,48 2,48A
TWIX.'XTRA-BATON 7	1 x2,48 2,48C	TWIX.'XTRA BATON 7	1 x2,48 2,48C
D.BOUN.BAT.KOKOS.5	1 x2,48 2,48A	D.BOUN.BAT.KOKOS.5	1 x2,48 2,48A
PEPERONI MIX 150G	1 x5,79 5,79C	PEPERONI MIX 150G	1 x5,79 5,79C
-----	-----	-----	-----
SPRZEDAŻ OPODATK. A	14,88	SPRZEDAŻ OPODATK. A	14,88
PTU A 23,00 %	2,78	PTU A 23,00 %	2,78
SPRZEDAŻ OPODATK. C	8,27	SPRZEDAŻ OPODATK. C	8,27
PTU C 5,00 %	0,39	PTU C 5,00 %	0,39
SUMA PTU	3,17	SUMA PTU	3,17
<b>SUMA PLN</b>	<b>23,15</b>	<b>SUMA PLN</b>	<b>23,15</b>

**Wynik metody "drawRectsOfObjects"**

Rysunek 3.5 Wyszukiwanie linii na obrazie. Źródło: opracowanie własne

### 3.3.2 ReceiptReaderService

Klasa o nazwie „ReceiptReaderService” jest przeznaczona do przechowywania metod, za pomocą których odbywa się rozpoznawanie tekstu na paragonach. Do celów OCR używa się biblioteka Tesseract oraz wcześniej przygotowany model sieci neuronowej. Tworzenie instancji Tesseract i konfiguracja silnika odbywa się za pośrednictwem metody „createTesseractInstanse”, która jest zaprezentowana na Listingu 3.8.

Listing 3.8. Tworzenie instancji Tesseract

```
0 private Tesseract createTesseractInstanse(language) {  
1     tesseract = new Tesseract();  
2     tesseract.setDatapath("src/main/resources/tessdata");  
3     tesseract.setLanguage(language);  
4     tesseract.setPageSegMode(7);  
5     tesseract.setOcrEngineMode(1);  
6     tesseract.setTessVariable("user_defined_dpi", "300");  
7     return tesseract;  
8 }
```

Metoda przyjmuje na wejściu nazwę języka, który stworzona instancja będzie rozpoznawała. Tesseract traktuje go jako nazwę modeli sieci neuronowej. Przygotowany wcześniej model o nazwie „auchan.traineddata” został umieszczona w danym projekcie pod ścieżką „src/main/resources/tessdata”. W liniach nr 2-3 dla instancji Tesseract ustawia się model, który będzie używany do rozpoznawania. Następnie ustawia się wartość „7” dla parametru segmentacji, co oznacza, że obraz będzie traktowany jako pojedyncza linia. W kolejnej linii jako wartość „1”, ustawia się silnik OCR oparty o sieci neuronowe. W linii nr 6 jest podawany DPI obrazu o wartości „300”.

Na Listingu 3.9, jest zaprezentowana publiczna metoda klasy „ReceiptReaderService”, dzięki której odbywa się proces rozpoznawania danych na paragonie. W niej odczytywane są najpierw dane sklepu i spółki, a następnie informacja o zakupionych produktach. Rezultat jej działania jest zapisywany do obiektu klasy „ReceiptForm”. Zawiera on mapę „receiptItems” oraz obiekt klasy „Receipt”, które zostaną wyświetcone na stronie głównej aplikacji. Kluczem mapy jest obraz linii tekstu, a wartością jest instancja klasy „ReceiptItem”. Przekazanymi parametrami są lista wyciętych z paragonu linii oraz nazwa modeli sieci neuronowej. Najpierw jest tworzona instancja Tesseract, a następnie są ustawiane wartości niezbędnych zmiennych. Zmienna „market” zawiera

informację o nazwie sklepu i spółki oraz ich adresy. W „lineImage” znajduje się obraz konkretnej linii, a rozpoznany na nim tekst jest zapisywany do zmiennej „lineText”. Na przekazywanych do aplikacji paragonach informacja dotycząca sklepu, jest podawana w pierwszych wierszach aż do linii, w której jest zamieszczona nazwa dokumentu. Na podstawie tego w liniach nr 7-13 odbywa się iteracja od pierwszego wiersza paragonu do momentu aż rozpoznany przez Tesseract tekst nie będzie zawierał napisu „paragon fiskalny”. W przypadku gdy się nie uda znaleźć takiej linijki, to metoda wyrzuci wyjątek z informacją o tym, że na obrazie nie ma paragonu fiskalnego. W linii nr 10 jest pobierany kolejny obraz z listy wejściowej, a w następnej odbywa się rozpoznawanie tekstu przy pomocy Tesseract. Aby zdefiniować jakie dane dotyczące sklepu zawiera tekst, jest on przekazywany do metody „addMarketData”. Linie na paragonie znajdują się pomiędzy nazwą dokumentu, a informacją o podatkach, zawierającą informację o zakupionych produktach. Czasami, po nich występuje linia z kilkoma myślnikami, która oznacza koniec listy produktów. Dla tego w liniach nr 14-21 proces iteracji kontynuuje się aż do końca wejściowej listy lub jeżeli rozpoznany na obrazie tekst zawiera minimum 5 myślników albo skrót od słowa „opodatkowanie”. Rozpoznany z obrazu tekst wraz z obrazem linii przekazuje się do metody „addReceiptItem”, aby dodać do listy produktów informację o każdym produkcie.

Listing 3.9. Odczytywanie danych z paragonu

```
0 public ReceiptForm readReceiptData (imageLineList, language) {  
1     tesseract = createTesseractInstanse(language);  
2     result = new ReceiptForm();  
3     market = result.getReceipt().getMarket();  
4     lineText = "";  
5     lineImage = null;  
6     imageLinesIterator = imageLineList.iterator();  
7     while (!lineText.toLowerCase().contains("paragon fiskalny")) {  
8         if (!imageLinesIterator.hasNext())  
9             throw new TesseractException(NO_RECEIPT_MESSAGE);  
10        lineImage = imageLinesIterator.next();  
11        lineText = tesseract.doOCR(lineImage);  
12        addMarketData(market, lineText);  
13    }  
14    while (imageLinesIterator.hasNext() &&  
15        !lineText.toLowerCase().contains("- - - -") ||
```

```

16     lineText.toLowerCase().contains("opodatk")
17   )) {
18     lineImage = imageLinesIterator.next();
19     lineText = tesseract.doOCR(lineImage);
20     addReceiptItem(result.getReceiptItems(), lineImage, lineText);
21   }
22   return result;
23 }

```

Metoda „addMarketData” zapisuje dane uzyskane z przekazanego tekstu do obiektu „market” (zob. Listing 3.10) . Informacja o sklepie może występować na paragonach w różnej kolejności. Aby zdefiniować to, jaką informację reprezentuje linia, tekst się sprawdza na obecność w nim konkretnego ciągu znaków. Jeżeli tekst zawiera napis „sp.z o.o.”, lub „s.a.”, to znaczy, że jest w nim zamieszczona nazwa spółki kapitałowej. W obiekcie „market” ustawia się nazwa spółki w sytuacji, gdy parametr ten jeszcze nie jest ustawiony, a tekst reprezentuje te dane. W innym przypadku, jeżeli tekst zawiera skrót od słów „ulica” lub „aleja”, to jest on traktowany jako dane adresowe. Aby rozróżnić adres spółki od adresu sklepu wzięto pod uwagę to, że każdy z nich następuje po nazwie jednostki. Dlatego tekst wejściowy ustawia się jako adres spółki w przypadku, gdy w obiekcie „market” jest już podana ją nazwa. To oznacza, że poprzednia linia tekstu zawierała nazwę spółki, więc obecna jest ją adresem. W innym przypadku, gdy adres spółki jest już ustawiony lub jeżeli ją nazwa jeszcze nie została zapisana, tekst wejściowy ustawia się jako adres sklepu. Jeżeli linia nie reprezentuje ani nazwy spółki, ani danych adresowych, to wtedy sprawdza się czy w obiekcie „market”, jest podana nazwa sklepu. Jeżeli nie, to jest ona ustawiana w przypadku, gdy tekst zawiera jedno ze słów: „sklep”, „market” lub „auchan”.

Listing 3.10. Przetwarzanie tekstu, do obiektu klasy Market

```

0 private void addMarketData(market, lineText) {
1   text = lineText.toLowerCase();
2   if (isPartnershipData(text, market.getPartnership()) {
3     market.setPartnership(lineText);
4   } else if (isStreetData(text)) {
5     if (market.getPartnershipAddress() == null
6       && market.getPartnership() != null ) {
7       market.setPartnershipAddress(lineText);
8     } else if (market.getAddress() == null) {
9       market.setAddress(lineText); }

```

```

10     } else if (isMarketNameData(text, market.getName()) )
11         market.setName(lineText);
12 }
13 private boolean isPartnershipData (data, partnership) {
14     return partnership == null &&
15             ( data.contains("sp.z o.o.") || data.contains("s.a."));
16 }
17 private boolean isStreetData (data) {
18     return data.contains("ul.") || data.contains("al.");
19 }
20 private boolean isMarketNameData (data, marketName) {
21     return marketName == null && (data.contains("sklep") ||
22             data.contains("market") || data.contains("auchan"));
23 }

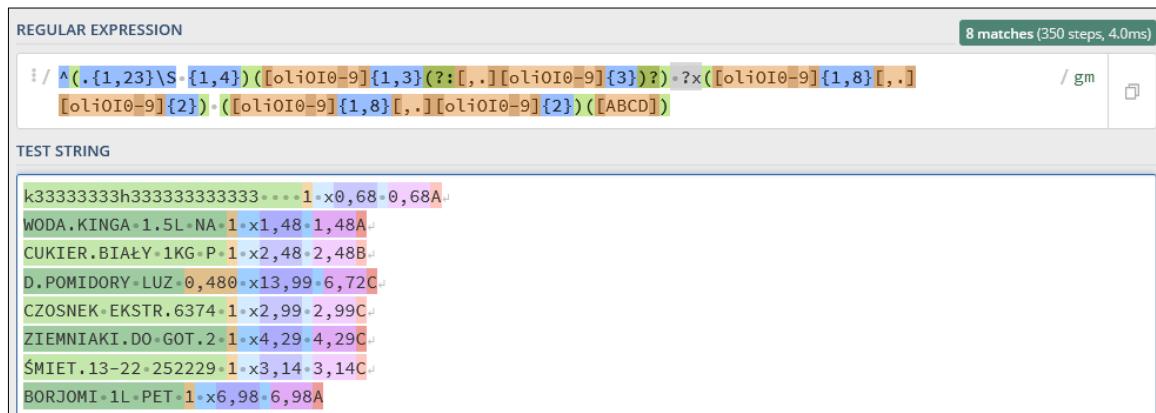
```

W metodzie „addReceiptItem” na podstawie przekazanego tekstu, tworzony jest obiekt klasy „ReceiptItem”, który razem z przekazanym obrazem linii jest dodawany do listy produktów (zob. Listing 3.11). W linii nr 6 tekst dopasowuje się z wyrażeniem regularnym, który jest zainicjowany w liniach nr 0-2. Według tego wzoru tekst zostanie rozbity na 5 grup:

- 1) ^(.{1,23}\S {1,4}) - nazwa produktu. Grupa ta musi być na początku. Najpierw idzie nazwa produktu składająca się z jakichkolwiek znaków, ilość których musi być pomiędzy 1 a 23 znaki. Za tym ciągiem następuje ostatni znak nazwy produktu, który musi być innym niż białe znaki. Ostatnie znaki tej grupy są znakami spacji, minimalnie jedna, a maksymalnie 4.
- 2) ([oliOI0-9]{1,3}(?:[.,][oliOI0-9]{3}))? - zakupiona ilość produktów. Wartość ta może występować na paragonie jako liczba całkowita lub jako zmienoprzecinkowa. Ilość znaków reprezentujących wartość całkowitą może być w przedziale od 1 do 3 znaków. Wartości dziesiętne mogą występować lub nie. Jeżeli tak, to muszą one być podane po przecinku lub kropki. Ilość znaków reprezentujących tę wartość jest w przedziale od 1 do 3 znaków. Tesseract podczas rozpoznawania może pomylić liczby „1” i „0” z literami „o”, „l” oraz „i”, dlatego są one też akceptowalne jako wartości numeryczne.
- 3) ?x([oliOI0-9]{1,8}[.][oliOI0-9]{2}) - cena za jednostkę. Grupa ta może być poprzedzona jedną spacją. Następnie występuje w niej znak „x” i wartość zmienoprzecinkowa. Ilość znaków przed przecinkiem jest w przedziale od 1 do 8 znaków, a po przecinku – 2 znaki.

- 4) ([oliOI0-9]{1,8}{1,8}[.][oliOI0-9]{2}) - suma ilości i ceny. Reprezentacja wartości jest taka sama jak wzór ceny za jednostkę.
- 5) ([ABCD]) - typ podatku, reprezentowany jako jedna z liter: „A”, „B”, „C” lub „D”.

Przykład tego jak tekst zawierający dane produktu dopasowuję się do użytego wzoru jest zaprezentowany na Rysunku 3.6.



Rysunek 3.6. Wyrażenie regularne opisujące informację o produkcie. Źródło: rzut ekranu ze strony internetowej: <https://regex101.com/>.

W przypadku gdy rozpoznany przez Tesseract tekst odpowiada wyrażeniu regularnemu, to odpowiednie jego fragmenty zostaną zapisane jako dane produktu do obiektu „receiptItem”. W liniach nr 21-27 jest zaprezentowana metoda, która jest używana do konwertowania tekstu na liczby zmiennoprzecinkowe. W przypadku gdy Tesseract pomyli litery z liczbami, to metoda ta dodatkowo pozamienia ich na poprawne wartości.

Listing 3.11. Odczytywanie informacji o produkcie z tekstu

```

0 private static final String PATTERN_ITEM = ^(.{1,23}\$|[1,4])([oliOI0-9]{1,3}
1  (?:[.,][oliOI0-9]{3})?)?x([oliOI0-9]{1,8}[.,][oliOI0-9]{2})([oliOI0-9]{1,8}
2   [.,][oliOI0-9]{2})([ABCD])";
3
4 private void addReceiptItem(receiptItems, lineImage, lineText) {
5   receiptItem = new ReceiptItem();
6   matcher = Pattern.compile(PATTERN_ITEM).matcher(lineText);
7   if(matcher.find()) {
8     itemName = matcher.group(1).trim();
9     itemAmount = stringToDouble(matcher.group(2));
10    itemPrice = stringToDouble(matcher.group(3));
11    itemPriceSum = stringToDouble(matcher.group(4));
12    itemTaxSign = matcher.group(5).charAt(0);

```

```

13     itemTax = taxRepository.findBySign(itemTaxSign);
14     receiptItem.setItem(new Item(itemName, itemTax));
15     receiptItem.setAmount(itemAmount);
16     receiptItem.setItemPrice(itemPrice);
17     receiptItem.setPriceSum(itemPriceSum);
18 }
19 receiptItems.put(lineImage, receiptItem);
20 }
21 private double stringToDouble (string) {
22     return Double.parseDouble (string.toLowerCase())
23         .replace(",",".")
24         .replace("o", "0")
25         .replace("l", "1")
26         .replace("i", "1"));
27 }
```

### 3.3.3 HomeController

Klasa „HomeController” jest jedynym kontrolerem w aplikacji. W wyniku działania każdej jego funkcji jest zwracany widok o nazwie “home”, implementacja którego znajduje się w pliku “home.html”. W zależności od żądania, kontroler przekazuje do tego widoku odpowiednie atrybuty, które są w nim wyświetlane lub nie. Przykładem takiego atrybutu jest “receiptForm”, który przekazuje się do widoku w każdym żądaniu. Jest to obiekt klasy ReceiptForm, która zawiera w sobie dane paragonu. Kierowanie do ścieżki “\home”, spowoduje, że kontroler przekaże do przeglądarki widok z pustymi znaczeniami atrybutu “receiptForm”. Na stronie zostaną wyświetcone dwa przyciski. Pierwszy z nich jest przeznaczony do wybierania paragonu, tekst, którego należy rozpoznać, a drugi do przekazywania wybranego obrazu do serwera. Podczas wysyłania pliku, przeglądarka kieruje się do ścieżki “\upload”, która w kontrolerze jest zmapowana z metodą o nazwie “readText”. Jej logika została przedstawiona w postaci pseudokodu na listingu 3.12. Proces rozpoznawania tekstu zaczyna się od linii nr. 3, w której odbywa się przetwarzanie wstępne. W tym kroku za pomocą klasy “OpenCvService” przekazany do serwera obraz opracowuje się w celu polepszenia jego jakości, a następnie zawarty w tej klasie algorytm wyodrębnia z paragonu wiersze i zapisuje je do zwracanej listy. W kolejnej linii ta lista jest przekazywana się do metody klasy “ReceiptReaderService”, w której odbywa się rozpoznawanie tekstu na każdym jej elemencie. Informacja zawarta w paragonie jest zapisywana do obiektu

receiptForm, a następnie w linii 5, ten obiekt jest dodawany do listy atrybutów widoku. W linii 6, widok „home”, wraz z uzupełnionym atrybutem są przekazywane z powrotem do przeglądarki.

Listing 3.12. Metoda kierująca procesem rozpoznawania

```

1 @PostMapping("/upload")
2 String readText (receiptImage, attributes) {
3     receiptLines [] = openCvService.getReceiptLines(receiptImage)
4     receiptForm = receiptReaderService.readReceiptData(receiptLines,"auchan")
5     attributes.addFlashAttribute("receiptForm", receiptForm)
6     return "redirect:/home"
7 }

```

Końcowym rezultatem jest wyświetlanie danych zawartych w paragonie, na stronie głównej aplikacji. Jej wygląd jest umieszczony w postaci rzutu ekranu na Rysunku 3.7.

The screenshot shows a Java application interface. On the left, there's a form titled "Paste your receipt there" with fields for "Nazwa" (set to "HIPERMARKET AUCHAN"), "Adres" (set to "31-346 KRAKÓW UL. STAWOWA 61"), "Nazwa spółki" (set to "AUCHAN POLSKA SP.Z O.O."), and "Adres spółki" (set to "UL.PULAWSKA 46 05-500 PIASECZNO"). Below the form, there's a section for "Artykuły" (Articles) with two tables:

Nazwa	Ilość	Cena	Podsumowanie	Typ podatku
PD TWAR POLTL 350G .	1	5,47	5,47	C
TORBA20L+OPŁ.REC	1	0,60	0,60	

Below these tables is the text "PD TWAR POLTL 350G . 1 x5,47 5,47C" and "TORBA20L+OPŁ.REC 1 x0,60 0,60A". To the right, there's a detailed receipt summary table:

PARAGON FISKALNY				
PD TWAR POLTL 350G	1	x5,47	5,47C	
TORBA20L+OPŁ.REC	1	x0,60	0,60A	
ŚWIKLÀ.315ML.	1	x3,69	3,69C	
#SODA OCZYSZCZONA	1	x0,68	0,68A	
WODA.KINGA 1,5L NA	1	x1,48	1,48A	
CUKIER.BIAŁY 1KG P	1	x2,48	2,48B	
D.POMIDORY LUZ	0,480	x13,99	6,72C	
ČOSNEK EKSTR.6374	1	x2,99	2,99C	
ZIEMNIAKI.DO GOT.2	1	x4,29	4,29C	
ŚMIET.13-22 252229	1	x3,14	3,14C	
BORJOMI 1L PET	1	x6,98	6,98A	
-----	-----	-----	-----	-----
SPRZEDAŻ OPODATK. A				9,74
PTU A 23,00 %				1,82
SPRZEDAŻ OPODATK. B				2,48
PTU B 8,00 %				0,18
SPRZEDAŻ OPODATK. C				26,30
PTU C 5,00 %				1,25
SUMA PTU				3,25
<b>SUMA PLN</b>				<b>38,52</b>

Rysunek 3.7. Widok strony głównej aplikacji, po rozpoznaniu tekstu z paragonu. Źródło:

*opracowanie własne*

Po prawej stronie znajduje się początkowy obraz paragonu, który został przekazany do systemu, a po lewej są zgrupowane jego dane. W pierwszej kolejności podane są dane sklepu, takie, jak nazwa, adres, nazwa spółki oraz jej adres. W kolejnym kroku została wymieniona lista artykułów. Jej elementy składają się z obrazu, którym jest wyodrębniony wiersz i rozpoznanych na nim danych, którymi są nazwa artykułu, zakupiona ilość, cena za

jednostkę, suma ilości i ceny oraz typ podatku. W przypadku, gdyby wyświetlona lista zawierała błędy, użytkownik ma możliwość ich poprawienia. Na samym dole znajduje się przycisk “Akceptuj”, za pomocą którego dane paragonu, można zapisać w bazie danych.

## 3.4 Testy i rezultaty

### 3.4.1 Test silnika Tesseract

Aby sprawdzić na ile poprawi się jakość silnika OCR po treningu, wstępnie przeprowadziłem test na paragonach, których nie będę używał podczas uczenia. Jest to kolejny przygotowany zbiór składający się z 8 plików formatu “lstmf”. Do oceny działania modeli LSTM służy program będący częścią oprogramowania “Tesseract” o nazwie “lstmeval”. Podałem na jego wejście ścieżkę do modelu w postaci pliku “pol.traineddata” oraz listę przygotowanych do testów danych jako wartość parametru „eval\_listfile”:

```
lstmeval --model pol.traineddata --eval_listfile testing_files.txt
```

Wynik jest pokazywany w postaci znakowego współczynnika błędu, wartość którego w tym wypadku wyniosła 39.27%. W przypadku wytrenowanego pliku o nazwie “ auchan.traineddata”, wartość ta zmniejszyła się o 32.43% i wynosi 6.83%.

### 3.4.2 Test aplikacji

Podczas testowania aplikacji była sprawdzana poprawność rozpoznawania kwoty wydanej za towar. Dodatkowo brane były pod uwagę cena jednostkowa towaru i zakupiona ilość. Kwota sprzedaży, oznakowana na paragonie jako „SUMA PLN”, jest wyznacznikiem tego czy każdy towar na paragonie został poprawnie rozpoznany. Oto kryteria, według których było uznawane, że dane na paragonie zostały poprawnie rozpoznane:

- 1) ilość towarów na obrazie odpowiada ilości rozpoznanych towarów;
- 2) iloczyn ceny jednostkowej towaru i jego zakupionej ilości jest równy kwocie wydanej za towar. Dodatkowo musi on odpowiadać kwocie za ten towar na obrazie;
- 3) suma kwot wszystkich towarów, jest równa kwocie sprzedaży. Wartość ta musi odpowiadać liczbie znajdującej się na obrazie w linijce „SUMA PLN”.

Do testów został przekazany zbiór danych, składający się z 8 obrazów. Obrazem jest zeskanowany paragon, który nie był używany na etapie uczenia Tesseract. Rezultaty testu, uwzględniając osiągnięcie wymaganych kryteriów, zostały zaprezentowane w Tabeli 3.1.

Tabela 3.1. Rezultaty rozpoznawania paragonów.

Nazwa obrazu	Kryterium 1	Kryterium 2	Kryterium 3	Spełnia wszystkie kryteria
Paragon0.png	spełniony	spełniony	spełniony	TAK
Paragon1.png	spełniony	spełniony	spełniony	TAK
Paragon2.png	spełniony	spełniony	spełniony	TAK
Paragon3.png	spełniony	spełniony	spełniony	TAK
Paragon4.png	spełniony	spełniony	spełniony	TAK
Paragon5.png	spełniony	spełniony	spełniony	TAK
Paragon6.png	spełniony	nie spełniony	nie spełniony	NIE
Paragon7.png	spełniony	spełniony	spełniony	TAK

Obraz o nazwie Paragon6.png nie spełnił drugiego kryterium, co stało się powodem niezaliczenia trzeciego kryterium. Przyczyną tego, jest to, że Tesseract nie rozpoznał liczby jeden, która reprezentowała ilość zakupionego towaru o nazwie „LOYD ZI MATCHA 20P”. To spowodowało, że aplikacja nie uwzględniała tych danych, a różnica pomiędzy wyliczoną sumą kwot wszystkich towarów, a prawdziwą wyniosła 5.59zł.

## **Podsumowanie**

Jako cel swojej pracy podjąłem próbę opracowania i implementacji metod rozpoznawania tekstu na obrazach paragonów fiskalnych przy pomocy oprogramowania OCR. Użyta przeze mnie biblioteka Tesseract ma funkcjonalność, dzięki której można nauczyć jej silnik rozpoznawania dowolnej czcionki. Do tego jest potrzebny plik czcionki lub obrazy zawierające tekst o wybranej czcionce, na podstawie których jest generowany zestaw danych uczących. W otwartym dostępie nie znalazłem plików czcionek, które są używane do drukowania tekstu na paragonach fiskalnych, dlatego musiałem własnoręcznie zebrać zestaw składający się z obrazów paragonów. Proces uczenia wymaga dużej liczby danych uczących. W drukarkach fiskalnych, nie zawsze jest używana jedna czcionka, co w moim przypadku oznaczałoby przygotowanie ogromnej ilości obrazów, dla każdej możliwej czcionki. W swojej pracy zaprezentowałem proces uczenia na podstawie zrobionych obrazów paragonów hipermarketu „Auchan”. Jednak uważam, że aplikacja, która miałaby ułatwić robienie zakupów lub która miałaby pomóc planować własne finanse musi być bardziej uniwersalną w kontekście rozpoznawania paragonów fiskalnych. Do celów uczenia został użyty silnik oprogramowania Tesseract, który jest oparty na sieciach neuronowych LSTM. Każdy obraz w zbiorze danych uczących został wstępnie przetworzony i podzielony na pojedyncze linie tekstu własnymi zaimplementowanymi algorytmami, przy pomocy biblioteki OpenCV.

Do sprawdzenia skuteczności nauczonego modelu sieci neuronowej Tesseract, stworzyłem aplikację webową, celem której jest odzyskanie informacji o zakupionych artykułach z przekazanego do aplikacji obrazu paragonu. Po stronie serwera odbywa się przetwarzanie obrazu, rozpoznawanie na nim tekstu oraz identyfikacja tego, jakie dane ten tekst zawiera. Identyfikacja nazwy, ceny jednostki, ilości oraz sumy zakupionego artykułu jest wykonywana na podstawie stworzonego wyrażenia regularnego. To wyrażenie akceptuje również znaki, które Tesseract mógł błędnie rozpoznać. Na przykład, jeżeli w cenie za jednostkę znajdują się takie litery jak ‘l’ lub ‘o’, to w kolejnym kroku działania aplikacji, zostaną one zamienione na liczby ‘1’ i ‘0’. Do klienta (lub przeglądarki), serwer wysyła stronę webową, na której znajduje się lista rozpoznanych artykułów wraz ze szczegółowym opisem.

Analizując wyniki, widać wysoką skuteczność przedstawionego rozwiązania. W ośmiu testowych przypadkach sprawdzały się trzy kryterium: rozpoznanie informacji o ilości artykułów, cena każdego towaru oraz wartość całej sprzedaży. W kryterium pierwszym system osiągnął 100% skuteczność, w kryteriach drugim i trzecim,

system pomylił się w pojedynczym przypadku, osiągając skuteczność 87,5%. Tym samym cel pracy, którym było opracowanie i implementacji metod rozpoznawania tekstu na obrazach paragonów fiskalnych został osiągnięty.

Podczas działania systemu zauważylem wysokie zużycie pamięci, a czas identyfikacji danych z pojedynczego obrazu zajmuje około 6 sekund. Nakłada to ograniczenie w stosunku do używania systemu przez kilka osób naraz. W celu rozwiązania tego problemu należałyby przenieść proces przetwarzania obrazu i identyfikacji tekstu do oddzielnych aplikacji klienta.

## Bibliografia

- [1] Verma P., Tesseract OCR with Java with Examples,  
<https://www.geeksforgeeks.org/tesseract-ocr-with-java-with-examples/>, (dostęp 07.2021).
- [2] He, X., Zheng, L., Wu, Q., Jia, W., Samali, B. & Palaniswami, M. (2008). Segmentation of characters on car license plates, *MMSP* (pp. 399-402), : IEEE Signal Processing Society. ISBN: 978-1-4244-2295-1,
- [3] Kunte, R Sanjeev & Samuel, R. (2008). A simple and efficient optical character recognition system for basic symbols in printed Kannada text. *Sadhana*. 32. (pp. 521-533). 10.1007/s12046-007-0039-1.
- [4] Forough K., The Comprehensive Guide to Optical Character Recognition (OCR),  
<https://moov.ai/en/blog/optical-character-recognition-ocr/>, (dostęp 07.2021).
- [5] Tesseract, <https://tesseract-ocr.github.io/>, (dostęp 07.2021).
- [6] Leptonica, <http://www.leptonica.org/>, (dostęp 07.2021).
- [7] Tadeusiewicz R. (1993), Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa, 2 wydanie.
- [8] Stęgowski Z. (2004), Sztuczne sieci neuronowe, Wydawnictwo Naukowo-Techniczne, Kraków.
- [9] Nielsen M. (2019), Neural Networks and Deep Learning,  
<http://neuralnetworksanddeeplearning.com/index.html>, (dostęp 07.2021).
- [10] Miernik, Tomasz. (2012). Wykonanie sztucznej sieci neuronowej do klasyfikacji depesz agencyjnych / Implementation of neural network to classify Reuters telegrams POLISH LANGUAGE. 10.13140/RG.2.2.23646.02887,
- [11] Łącki M. (2016), Intelligent prediction of ship maneuvering, *TransNav, International Journal on Marine Navigation and Safety of Sea Transportation* 10 (3), pp. 511-516.
- [12] Dobrosielski W. (2010), Zastosowanie sztucznych sieci neuronowych do rozpoznawania obrazów, *Studia i Materiały Informatyki Stosowanej* Tom 2, Nr 3, Uniwersytet Kazimierza Wielkiego, Bydgoszcz.
- [13] Stateczny A. (2002), Sztuczne sieci neuronowe w rozpoznawaniu obiektów morskich, GTN, Gdynia.
- [14] Dawida S. (2020), Wykorzystanie sieci neuronowych do rozpoznawania statycznych gestów dłońmi do sterowania robotem mobilnym z wykorzystaniem kamery / Using neural networks to recognize hand static gestures to control a mobile robot using a camera, Akademia Górnictwa-Hutnicza im. Stanisława Staszica w Krakowie, Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej,

- [15] Srivastala P. (10.12.2017), Essentials of Deep Learning : Introduction to Long Short Term Memory, <https://www.analyticsvidhya.com/blog/2017/12/fundamentals-of-deep-learning-introduction-to-lstm/>, (dostęp 07.2021).
- [16] Dorobek M. (2020), Generating Jazz Chords Progressions Using Word Embeddings and Recurrent Neural Networks, Politechnika Warszawska, Wydział Matematyki i Informatyki.
- [17] Olah C. (2015), Understanding LSTM Networks, <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>, (dostęp 07.2021).
- [18] Ogan. L (2018), Sieci neuronowe generują muzykę – część 3 cyklu, <https://blog.lukaszogan.com/informatyka/rekurencyjne-sieci-neuronowe-generuja-muzyke/>, (dostęp 07.2021).
- [19] Tesseract trained models with support for legacy and LSTM OCR engine, <https://github.com/tesseract-ocr/tessdata>, (dostęp 07.2021).
- [20] Instrukcja dotycząca uczenia Tesseract 4.00, <https://tesseract-ocr.github.io/tessdoc/tess4/TrainingTesseract-4.00.html#creating-training-data>, (dostęp 07.2021).
- [21] Skrypt używany do uczenia Tesseract, <https://github.com/svn2github/tesseract-ocr/blob/master/training/tesstrain.sh>, (dostęp 07.2021).
- [22] jTessBoxEditor, <http://vietocr.sourceforge.net/training.html>, (dostęp 07.2021).
- [23] Tesseract documentation, Improving the quality of the output, <https://tesseract-ocr.github.io/tessdoc/ImproveQuality>, (dostęp 07.2021).
- [24] Radzieński M. (2007), Typowe przekształcenia morfologiczne, <http://atol.am.gdynia.pl/tc/Radzienski/morfologiczne.htm>, (dostęp 07.2021).

## Dodatek. Paragony użyte do testów

<p>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174</p> <p>2021-03-31 24512</p> <p>PARAGON FISKALNY</p>															
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>TORBA15L+OPL.REC</td> <td>1</td> <td>0,48</td> <td>0,48A</td> </tr> <tr> <td>D.PIWO BUDW.BUDV.O</td> <td>1</td> <td>3,48</td> <td>3,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : TORBA15L+OPL.REC 1 x0,48 00,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	TORBA15L+OPL.REC	1	0,48	0,48A	D.PIWO BUDW.BUDV.O	1	3,48	3,48A
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
TORBA15L+OPL.REC	1	0,48	0,48A												
D.PIWO BUDW.BUDV.O	1	3,48	3,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>D.PIWO BUDW.BUDV.O</td> <td>1</td> <td>3,48</td> <td>3,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : D.PIWO BUDW.BUDV.O 1 x3,48 3,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	D.PIWO BUDW.BUDV.O	1	3,48	3,48A				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
D.PIWO BUDW.BUDV.O	1	3,48	3,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>D.PIWO BUDW.BUDV.O</td> <td>1</td> <td>3,48</td> <td>3,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : D.PIWO BUDW.BUDV.O 1 x3,48 3,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	D.PIWO BUDW.BUDV.O	1	3,48	3,48A				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
D.PIWO BUDW.BUDV.O	1	3,48	3,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>SNICKERS.SUPER BAT</td> <td>1</td> <td>2,48</td> <td>2,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : SNICKERS.SUPER BAT 1 x2,48 2,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	SNICKERS.SUPER BAT	1	2,48	2,48A				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
SNICKERS.SUPER BAT	1	2,48	2,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>MARS.2PACK 70G</td> <td>1</td> <td>2,48</td> <td>2,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : MARS.2PACK 70G 1 x2,48 2,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	MARS.2PACK 70G	1	2,48	2,48A				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
MARS.2PACK 70G	1	2,48	2,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>TWIX.'XTRA-BATON 7</td> <td>1</td> <td>2,48</td> <td>2,48C</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : TWIX. XTRA-BATON 7 1 x2,48 2,48C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	TWIX.'XTRA-BATON 7	1	2,48	2,48C				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
TWIX.'XTRA-BATON 7	1	2,48	2,48C												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>D.BOLIN.BAT.KOKOS.5</td> <td>1</td> <td>2,48</td> <td>2,48A</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : D.BOLUN.BAT.KOKOS.5 1 x2,48 2,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	D.BOLIN.BAT.KOKOS.5	1	2,48	2,48A				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
D.BOLIN.BAT.KOKOS.5	1	2,48	2,48A												
<table border="1"> <thead> <tr> <th>Nazwa</th> <th>Ilość</th> <th>Cena</th> <th>Podsumowanie Typ podatku</th> </tr> </thead> <tbody> <tr> <td>PEPERONI MIX 150G</td> <td>1</td> <td>5,79</td> <td>5,79C</td> </tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : PEPERONI MIX 150G 1 x5,79 5,79C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	PEPERONI MIX 150G	1	5,79	5,79C				
Nazwa	Ilość	Cena	Podsumowanie Typ podatku												
PEPERONI MIX 150G	1	5,79	5,79C												

Rysunek 0.1. Paragon0.png

ORZECZY.LIGHT.380G 1 x9,48 9,48C			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
ORZECZY.LIGHT.380G	1	9,48	C
Rozpoznany przez Tesseract tekst : ORZECZY.LIGHT.380G 1 x9,48 9,48C			
CZEK.BIAŁA.KARM. O 1 x3,48 3,48A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
CZEK.BIAŁA.KARM. O	1	3,48	A
Rozpoznany przez Tesseract tekst : CZEK.BIAŁA.KARM. O 1 x3,48 3,48A			
ZŁOTE.LWY BUT.0,5L 1 x3,48 3,48A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
ZŁOTE.LWY BUT.0,5L	1	3,48	A
Rozpoznany przez Tesseract tekst : ZŁOTE.LWY BUT.0,5L 1 x3,48 3,48A			
PIWO.HEIN.0,5L 308 1 x3,59 3,59A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
PIWO.HEIN.0,5L 308	1	3,59	A
Rozpoznany przez Tesseract tekst : PIWO.HEIN.0,5L 308 1 x3,59 3,59A			
CAPTAIN.JACK ORAN 1 x3,19 3,19A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
CAPTAIN.JACK ORAN	1	3,19	A
Rozpoznany przez Tesseract tekst : CAPTAIN.JACK ORAN 1 x3,19 3,19A			
SOMERSBY.WATERM.4, 1 x4,09 4,09A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
SOMERSBY.WATERM.4,	1	4,09	A
Rozpoznany przez Tesseract tekst : SOMERSBY.WATERM.4, 1 x4,09 4,09A			
SOMERSBY.WATERM.4, 1 x4,09 4,09A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
SOMERSBY.WATERM.4,	1	4,09	A
Rozpoznany przez Tesseract tekst : SOMERSBY.WATERM.4, 1 x4,09 4,09A			
CYDR.JABL.O,33L 4, 1 x3,39 3,39A			
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>
CYDR.JABL.O,33L 4,	1	3,39	A
Rozpoznany przez Tesseract tekst : CYDR.JABL.O,33L 4, 1 x3,39 3,39A			

Rysunek 0.2. Paragon1.png

<p>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PULAWSKA 46 05-500 PIASECZNO <a href="http://www.auchan.pl">www.auchan.pl</a> BDO 000013917 NIP 526-03-09-174</p> <p>2021-04-03 32082</p> <p>PARAGON FISKALNY</p> <table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>PD TWAR POLTL 350G</td><td>1</td><td>5,47</td><td>5,47C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : PD TWAR POLTL 350G 1 x5,47 5,47C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	PD TWAR POLTL 350G	1	5,47	5,47C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
PD TWAR POLTL 350G	1	5,47	5,47C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>TORBA20L+OPŁ.REC</td><td>1</td><td>0,60</td><td>0,60A</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : TORBA20L+OPŁ.REC 1 x0,60 0,60A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	TORBA20L+OPŁ.REC	1	0,60	0,60A
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
TORBA20L+OPŁ.REC	1	0,60	0,60A								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>ĆWIĘKŁA.315ML.</td><td>1</td><td>3,69</td><td>3,69C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : ĆWIĘKŁA.315ML. 1x3,69 3,69C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	ĆWIĘKŁA.315ML.	1	3,69	3,69C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
ĆWIĘKŁA.315ML.	1	3,69	3,69C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>#SODA OCZYSZCZONA</td><td>1</td><td>0,68</td><td>0,68A</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : #SODA OCZYSZCZONA 1 x0,68 0,68A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	#SODA OCZYSZCZONA	1	0,68	0,68A
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
#SODA OCZYSZCZONA	1	0,68	0,68A								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>WODA.KINGA 1.5L NA</td><td>1</td><td>1,48</td><td>1,48A</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : WODA.KINGA 1.5L NA 1 x1,48 1,48A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	WODA.KINGA 1.5L NA	1	1,48	1,48A
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
WODA.KINGA 1.5L NA	1	1,48	1,48A								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>CUKIER.BIAŁY 1KG P</td><td>1</td><td>2,48</td><td>2,48B</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : CUKIER.BIAŁY 1KG P 1 x2,48 2,48B</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	CUKIER.BIAŁY 1KG P	1	2,48	2,48B
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
CUKIER.BIAŁY 1KG P	1	2,48	2,48B								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>D.POMIDORY LUZ</td><td>0,480</td><td>x13,99</td><td>6,72C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : D.POMIDORY LUZ 0,480 x13,99 6,72C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	D.POMIDORY LUZ	0,480	x13,99	6,72C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
D.POMIDORY LUZ	0,480	x13,99	6,72C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>CZOSNEK EKSTR.6374</td><td>1</td><td>2,99</td><td>2,99C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : CZOSNEK EKSTR.6374 1 x2,99 2,99C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	CZOSNEK EKSTR.6374	1	2,99	2,99C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
CZOSNEK EKSTR.6374	1	2,99	2,99C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>ZIEMNIAKI.DO GOT.2</td><td>1</td><td>4,29</td><td>4,29C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : ZIEMNIAKI.DO GOT.2 1 x4,29 4,29C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	ZIEMNIAKI.DO GOT.2	1	4,29	4,29C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
ZIEMNIAKI.DO GOT.2	1	4,29	4,29C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>ŚMIET.13-22 252229</td><td>1</td><td>3,14</td><td>3,14C</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : ŚMIET.13-22 252229 1 x3,14 3,14C</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	ŚMIET.13-22 252229	1	3,14	3,14C
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
ŚMIET.13-22 252229	1	3,14	3,14C								
<table border="1"> <thead> <tr> <th>Nazwa</th><th>Ilość</th><th>Cena</th><th>Podsumowanie Typ podatku</th></tr> </thead> <tbody> <tr> <td>BORJOMI 1L PET</td><td>1</td><td>6,98</td><td>6,98A</td></tr> </tbody> </table> <p>Rozpoznany przez Tesseract tekst : BORJOMI 1L PET 1 x6,98 6,98A</p>				Nazwa	Ilość	Cena	Podsumowanie Typ podatku	BORJOMI 1L PET	1	6,98	6,98A
Nazwa	Ilość	Cena	Podsumowanie Typ podatku								
BORJOMI 1L PET	1	6,98	6,98A								

Rysunek 0.3. Paragon2.png

<p>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PULAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174</p> <p>2021-03-31 29251</p> <p>PARAGON FISKALNY</p> <table border="1"> <tbody> <tr><td>PAPIER WC 40913</td><td>1</td><td>x6,99</td><td>6,99A</td></tr> <tr><td>OLEJ.BESKIDZKI 1L</td><td>1</td><td>x5,98</td><td>5,98C</td></tr> <tr><td>TORBA30X50+OPŁ.REC</td><td>1</td><td>x0,48</td><td>0,48A</td></tr> <tr><td>TEO PIES MINI 1,5K</td><td>1</td><td>x9,99</td><td>9,99B</td></tr> <tr><td>PIW.ZYWIEC.0,5L P.</td><td>1</td><td>x2,98</td><td>2,98A</td></tr> <tr><td>PIW.ZYWIEC.0,5L P.</td><td>1</td><td>x2,98</td><td>2,98A</td></tr> <tr><td>*SOK.TYMB.JABLKOWY</td><td>1</td><td>x3,78</td><td>3,78C</td></tr> <tr><td>LU.MAK SPAG CIE 50</td><td>1</td><td>x3,28</td><td>3,28C</td></tr> <tr><td>MASŁO EXTRA POL.20</td><td>1</td><td>x3,99</td><td>3,99C</td></tr> <tr><td>-----</td><td></td><td></td><td></td></tr> <tr><td>SPRZEDAŻ OPODATK. A</td><td></td><td>13,43</td><td></td></tr> <tr><td>PTU A 23,00 %</td><td></td><td>2,51</td><td></td></tr> <tr><td>SPRZEDAŻ OPODATK. B</td><td></td><td>9,99</td><td></td></tr> <tr><td>PTU B 8,00 %</td><td></td><td>0,74</td><td></td></tr> <tr><td>SPRZEDAŻ OPODATK. C</td><td></td><td>17,03</td><td></td></tr> <tr><td>PTU C 5,00 %</td><td></td><td>0,81</td><td></td></tr> <tr><td>SUMA PTU</td><td></td><td>4,06</td><td></td></tr> <tr><td><b>SUMA PLN</b></td><td></td><td><b>40,45</b></td><td></td></tr> </tbody> </table>				PAPIER WC 40913	1	x6,99	6,99A	OLEJ.BESKIDZKI 1L	1	x5,98	5,98C	TORBA30X50+OPŁ.REC	1	x0,48	0,48A	TEO PIES MINI 1,5K	1	x9,99	9,99B	PIW.ZYWIEC.0,5L P.	1	x2,98	2,98A	PIW.ZYWIEC.0,5L P.	1	x2,98	2,98A	*SOK.TYMB.JABLKOWY	1	x3,78	3,78C	LU.MAK SPAG CIE 50	1	x3,28	3,28C	MASŁO EXTRA POL.20	1	x3,99	3,99C	-----				SPRZEDAŻ OPODATK. A		13,43		PTU A 23,00 %		2,51		SPRZEDAŻ OPODATK. B		9,99		PTU B 8,00 %		0,74		SPRZEDAŻ OPODATK. C		17,03		PTU C 5,00 %		0,81		SUMA PTU		4,06		<b>SUMA PLN</b>		<b>40,45</b>	
PAPIER WC 40913	1	x6,99	6,99A																																																																								
OLEJ.BESKIDZKI 1L	1	x5,98	5,98C																																																																								
TORBA30X50+OPŁ.REC	1	x0,48	0,48A																																																																								
TEO PIES MINI 1,5K	1	x9,99	9,99B																																																																								
PIW.ZYWIEC.0,5L P.	1	x2,98	2,98A																																																																								
PIW.ZYWIEC.0,5L P.	1	x2,98	2,98A																																																																								
*SOK.TYMB.JABLKOWY	1	x3,78	3,78C																																																																								
LU.MAK SPAG CIE 50	1	x3,28	3,28C																																																																								
MASŁO EXTRA POL.20	1	x3,99	3,99C																																																																								
-----																																																																											
SPRZEDAŻ OPODATK. A		13,43																																																																									
PTU A 23,00 %		2,51																																																																									
SPRZEDAŻ OPODATK. B		9,99																																																																									
PTU B 8,00 %		0,74																																																																									
SPRZEDAŻ OPODATK. C		17,03																																																																									
PTU C 5,00 %		0,81																																																																									
SUMA PTU		4,06																																																																									
<b>SUMA PLN</b>		<b>40,45</b>																																																																									
Rozpoznany przez Tesseract tekst : PAPTER WC 40913 1 x6,99 6,99A																																																																											
<b>OLEJ.BESKIDZKI 1L</b>	1	x5,98	5,98C																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
OLEJ.BESKIDZKI 1L	1	5,98	5,98 C																																																																								
Rozpoznany przez Tesseract tekst : OLEJ.BESKIDZKI 1L 1 x5,98 5,98C																																																																											
<b>TORBA30X50+OPŁ.REC</b>	1	x0,48	0,48A																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
TORBA30X50+OPŁ.REC	1	0,48	0,48 A																																																																								
Rozpoznany przez Tesseract tekst : TORBA30X50+OPŁ.REC 1 x0,48 0,48A																																																																											
<b>TEO PIES MINI 1,5K</b>	1	x9,99	9,99B																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
TEO PIES MINI 1,5K	1	9,99	9,99 B																																																																								
Rozpoznany przez Tesseract tekst : TEO PIES MINI 1,5K 1x9,99 9,99B																																																																											
<b>PIW.ZYWIEC.0,5L P.</b>	1	x2,98	2,98A																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
PIW.ZYWIEC.0,5L P.	1	2,98	2,98 A																																																																								
Rozpoznany przez Tesseract tekst : PIW.ZYWIEC.0,5L P. 1 x2,98 2,98A																																																																											
<b>PIW.ZYWIEC.0,5L P.</b>	1	x2,98	2,98A																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
PIW.ZYWIEC.0,5L P.	1	2,98	2,98 A																																																																								
Rozpoznany przez Tesseract tekst : PIW.ZYWIEC.0,5L P. 1 x2,98 2,98A																																																																											
<b>*SOK.TYMB.JABLKOWY</b>	1	x3,78	3,78C																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
*SOK.TYMB.JABLKOWY	1	3,78	3,78 C																																																																								
Rozpoznany przez Tesseract tekst : *SOK.TYMB.JABLKOWY 1 x3,78 3,78C																																																																											
<b>LU.MAK SPAG CIE 50</b>	1	x3,28	3,28C																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
LU.MAK SPAG CIE 50	1	3,28	3,28 C																																																																								
Rozpoznany przez Tesseract tekst : LU.MAK SPAG CIE 50 1 x3,28 3,28C																																																																											
<b>MASŁO EXTRA POL.20</b>	1	x3,99	3,99C																																																																								
<b>Nazwa</b>	<b>Ilość</b>	<b>Cena</b>	<b>Podsumowanie Typ podatku</b>																																																																								
MASŁO EXTRA POL.20	1	3,99	3,99 C																																																																								
Rozpoznany przez Tesseract tekst : MASŁO EXTRA POL.20 1 x3,99 3,99C																																																																											

Rysunek 0.4. Paragon3.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL. PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174			
2021-04-10 50750			
PARAGON FISKALNY			
JAJA SZT.10 KL L 1 x5,98 5,98C			
REJKAW.UNIWER.WINYL 1 x5,79 5,79B			
PIECZARKI 10847 0,210 x7,98 1,68C			
D CEBULA LUZ 0,190 x1,99 0,38C			
SER ZŁOTY MAZUR ML 0,300 x19,49 5,85C			
TORTILLA.PSZENN.24 1 x4,19 4,19C			
D PIECZARKI 500G 1 x4,98 4,98C			
D.PROSZ.DO PIECZ.3 1 x0,58 0,58A			
SER TOP.412097 1 x1,98 1,98C			
D CEBULA LUZ 0,208 x1,99 0,41C			
SPRZEDAŻ OPODATK. A 11,16			
PTU A 23,00 % 2,09			
SPRZEDAŻ OPODATK. B 12,77			
PTU B 8,00 % 0,95			
SPRZEDAŻ OPODATK. C 47,64			
PTU C 5,00 % 2,27			
SUMA PTU 5,31			
<b>SUMA PLN</b> 71,57			

Rysunek 0.5. Paragon4\_1.png

<p>HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL. PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174</p>			
<p>2021-04-10 50750</p>			
PARAGON FISKALNY			
JAJA SZT.10 KL L	1	x5,98	5,98C
REKW. UNIWER.WINYL	1	x5,79	5,79B
PIECZARKI 10847	0,210	x7,98	1,68C
D CEBULA LUZ	0,190	x1,99	0,38C
SER ZŁOTY MAZUR ML	0,300	x19,49	5,85C
TORTILLA.PSZENN.24	1	x4,19	4,19C
D PIECZARKI 500G	1	x4,98	4,98C
D.PROSZ.DO PIECZ.3	1	x0,58	0,58A
SER TOP.412097	1	x1,98	1,98C
SER TOP.412097	1	x1,98	1,98C
D CEBULA LUZ	0,208	x1,99	0,41C
JABL-TYP.A UKLADAN	0,344	x3,69	1,27C
ACKOPEREK PECZEK	1	x1,69	1,69C
PAPIER WC 40913	1	x6,99	6,99A
MAKA.WROCŁAWSKA 1K	1	x2,19	2,19C
MLEKO ŁACIATE 2% 1	1	x2,68	2,68C
HELLM.MAJO ORY.420	1	x6,98	6,98B
MASCARPONE 250G	1	x5,98	5,98C
ŚMIET.13-22 252229	1	x3,32	3,32C
PALUSZKI.SURIMI 25	1	x2,99	2,99A
KEFIR LUKS.BUT.1L	1	x3,08	3,08C
TORBA20L+OPL.REC	1	x0,60	0,60A
SPRZEDĄZ OPODATK. A		11,16	
PTU A 23,00 %		2,09	
SPRZEDĄZ OPODATK. B		12,77	
PTU B 8,00 %		0,95	
SPRZEDĄZ OPODATK. C		47,64	
PTU C 5,00 %		2,27	
SUMA PTU		5,31	
<b>SUMA PLN</b>		<b>71,57</b>	

Rysunek 0.6. Paragon4\_2.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL. PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	MLECZKO UHT 200ML 1 x1,48 1,48C Nazwa Ilość Cena Podsumowanie Typ podatku MLECZKO UHT 200ML  1  1,48  1,48   C Rozpoznany przez Tesseract tekst : MLECZKO UHT 200ML 1 x1,48 1,48C
2021-04-30 58094 PARAGON FISKALNY	D #LIPTON YL 25 EX 1 x5,27 5,27A Nazwa Ilość Cena Podsumowanie Typ podatku D #LIPTON YL 25 EX  1  5,27  5,27   A Rozpoznany przez Tesseract tekst : D #LIPTON YL 25 EX 1 x5,27 5,27A
MLECZKO UHT 200ML 1 x1,48 1,48C D #LIPTON YL 25 EX 1 x5,27 5,27A BUŁECZ PSZENN-5594 1 x2,99 2,99C ATAK.CHMIELU AMERI 1 x7,69 7,69A BZM.ZYTN.CICHY WIE 1 x7,69 7,69A D.PIWO BUDW.BUDV.O 1 x3,48 3,48A D.PIWO BUDW.BUDV.O 1 x3,48 3,48A TARTARE ŁOSOŚ.3570 1 x4,58 4,58C D.PIWO BUDW.BUDV.O 1 x3,48 3,48A JAJA SZT.10 KL L 1 x5,98 5,98C RUKOLA 75G PP 1 x2,99 2,99C BZM.ZYTN.CICHY WIE 1 x7,69 7,69A MLEKO ŁACIAT.3,2% 1 x2,68 2,68C FAVITA 18% 270G 1 x3,99 3,99C RISO WISNIA 200G 1 x1,59 1,59B FELIX.ORZESZKI.Z P 1 x5,08 5,08C CYTRYNY LUZ 33995. 0,190 x4,99 0,95C POLEDWICZ.WP 57150 0,575 x19,99 11,49C CZEK.CIEMN.DESER.2 1 x8,49 8,49A	BUŁECZ PSZENN-5594 1 x2,99 2,99C Nazwa Ilość Cena Podsumowanie Typ podatku BUŁECZ PSZENN-5594  1  2,99  2,99   C Rozpoznany przez Tesseract tekst : BUŁECZ PSZENN-5594 1 x2,99 2,99C
----- SPRZEDAŻ OPODATK. A 47,27 PTU A 23,00 % 8,84 SPRZEDAŻ OPODATK. B 1,59 PTU B 8,00 % 0,12 SPRZEDAŻ OPODATK. C 42,21 PTU C 5,00 % 2,01 SUMA PTU 10,97  <b>SUMA PLN 91,07</b>	ATAK.CHMIELU AMERI 1 x7,69 7,69A Nazwa Ilość Cena Podsumowanie Typ podatku ATAK.CHMIELU AMERI  1  7,69  7,69   A Rozpoznany przez Tesseract tekst : ATAK.CHMIELU AMERI 1 x7,69 7,69A
----- D.PIWO BUDW.BUDV.O 1 x3,48 3,48A Nazwa Ilość Cena Podsumowanie Typ podatku D.PIWO BUDW.BUDV.O  1  3,48  3,48   A Rozpoznany przez Tesseract tekst : D.PIWO BUDW.BUDV.O 1 x3,48 3,48A	BZM.ZYTN.CICHY WIE 1 x7,69 7,69A Nazwa Ilość Cena Podsumowanie Typ podatku BZM.ZYTN.CICHY WIE  1  7,69  7,69   A Rozpoznany przez Tesseract tekst : BZM.ZYTN.CICHY WIE 1 x7,69 7,69A
----- D.PIWO BUDW.BUDV.O 1 x3,48 3,48A Nazwa Ilość Cena Podsumowanie Typ podatku D.PIWO BUDW.BUDV.O  1  3,48  3,48   A Rozpoznany przez Tesseract tekst : D.PIWO BUDW.BUDV.O 1 x3,48 3,48A	TARTARE ŁOSOŚ.3570 1 x4,58 4,58C Nazwa Ilość Cena Podsumowanie Typ podatku TARTARE ŁOSOŚ.3570  1  4,58  4,58   C Rozpoznany przez Tesseract tekst : TARTARE ŁOSOŚ.3570 1 x4,58 4,58C
----- D.PIWO BUDW.BUDV.O 1 x3,48 3,48A Nazwa Ilość Cena Podsumowanie Typ podatku D.PIWO BUDW.BUDV.O  1  3,48  3,48   A Rozpoznany przez Tesseract tekst : D.PIWO BUDW.BUDV.O 1 x3,48 3,48A	JAJA SZT.10 KL L 1 x5,98 5,98C Nazwa Ilość Cena Podsumowanie Typ podatku JAJA SZT.10 KL L  1  5,98  5,98   C Rozpoznany przez Tesseract tekst : JAJA SZT.10 KL L 1 x5,98 5,98C

Rysunek 0.7. Paragon5\_1.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	RUKOLA 75G PP 1 x2,99 2,99C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> RUKOLA 75G PP   1   2,99   2,99   C Rozpoznany przez Tesseract tekst : RUKOLA 75G PP 1 x2,99 2,99C
2021-04-30 58094	BZM.ZYTN.CICHY WIE 1 x7,69 7,69A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> BZM.ZYTN.CICHY WIE   1   7,69   7,69   A Rozpoznany przez Tesseract tekst : BZM.ZYTN.CICHY WIE 1 x7,69 7,69A
PARAGON FISKALNY	MLEKO ŁACIAT.3,2% 1 x2,68 2,68C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> MLEKO ŁACIAT.3,2   1   2,68   2,68   C Rozpoznany przez Tesseract tekst : MLEKO ŁACIAT.3,2 1 x2,68 2,68C
MLECZKO UHT 200ML D #LIPTON YL 25 EX BUŁECZ PSZENN-5594 ĀTAK.CHMIELU AMERI BZM.ZYTN.CICHY WIE D.PIWO BUDW.BUDV.0 D.PIWO BUDW.BUDV.0 TARTARE ŁOSÓŚ.3570 D.PIWO BUDW.BUDV.0 JAJA SZT.10 KL L RUKOLA 75G PP BZM.ZYTN.CICHY WIE MLEKO ŁACIAT.3,2% FAVITA 18% 270G RISO WISNIA 200G FELIX.ORZESZKI.Z P CYTRYNY LUZ 33995.	FAVITA 18% 270G 1 x3,99 3,99C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> FAVITA 18% 270G   1   3,99   3,99   C Rozpoznany przez Tesseract tekst : FAVITA 18% 270G 1 x3,99 3,99C
1 x1,48 1,48C 1 x5,27 5,27A 1 x2,99 2,99C 1 x7,69 7,69A 1 x7,69 7,69A 1 x3,48 3,48A 1 x3,48 3,48A 1 x4,58 4,58C 1 x3,48 3,48A 1 x5,98 5,98C 1 x2,99 2,99C 1 x7,69 7,69A 1 x2,68 2,68C 1 x3,99 3,99C 1 x1,59 1,59B 1 x5,08 5,08C 0,190 x4,99 0,95C 0,575 x19,99 11,49C 1 x8,49 8,49A	RISO WISNIA 200G 1 x1,59 1,59B <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> RISO WISNIA 200G   1   1,59   1,59   B Rozpoznany przez Tesseract tekst : RISO WISNIA 200G 1x1,59 1,59B
SPRZEDAŻ OPODATK. A PTU A 23,00 % SPRZEDAŻ OPODATK. B PTU B 8,00 % SPRZEDAŻ OPODATK. C PTU C 5,00 % SUMA PTU	FELIX.ORZESZKI.Z P 1 x5,08 5,08C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> FELIX.ORZESZKI.Z P   1   5,08   5,08   C Rozpoznany przez Tesseract tekst : FELIX.ORZESZKI.Z P 1x5,08 5,08C
47,27 8,84 1,59 0,12 42,21 2,01 10,97	CYTRYNY LUZ 33995. 0,190 x4,99 0,95C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> CYTRYNY LUZ 33995.   0,19   4,99   0,95   C Rozpoznany przez Tesseract tekst : CYTRYNY LUZ 33995. 0,190 x4,99 0,95C
SUMA PLN 91,07	POLEDWICZ.WP 57150 0,575 x19,99 11,49C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> POLEDWICZ.WP 57150   0,575   19,99   11,49   C Rozpoznany przez Tesseract tekst : POLEDWICZ.WP 57150 0,575 x19,99 11,49C
	CZEK.CIEMN.DESER.2 1 x8,49 8,49A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> CZEK.CIEMN.DESER.2   1   8,49   8,49   A Rozpoznany przez Tesseract tekst : CZEK.CIEMN.DESER.2 1 x8,49 8,49A

Rysunek 0.8. Paragon5\_2.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	NACZYNIE żARO 5,8L 1 x24,99 24,99A RECZ.KUCH. 804352 1 x4,79 4,79A LOYD ZI MATCHA 20P 1 x5,59 5,59A POMID.KOK.GALAZ.50 1 x7,49 7,49C D MARCHEW LUZ 0,126 x3,49 0,44C MASŁO EXTRA 200G 1 x4,99 4,99C LIPTON EARL GR.LEM 1 x5,98 5,98A CENOS.KASZ. GRYCZ. 1 x5,58 5,58C D CEBULÄ LUZ 0,210 x2,49 0,52C PRZECIER-Z ZIOŁ ŚR 1 x3,38 3,38C KUR PODUDZIE LUZ. 0,784 x7,48 5,86C PAPIER WC 40913 1 x6,99 6,99A SZWAJCAR KRÓLEWIEC 0,366 x20,99 7,68C WODA.KINGA GAZOWAN 1 x1,58 1,58A COCA.COLA 1,5L PET 1 x5,54 5,54A  ----- SPRZEDAŻ OPODATK. A 55,46 PTU A 23,00 % 10,37 SPRZEDAŻ OPODATK. C 35,94 PTU C 5,00 % 1,71 SUMA PTU 12,08  <b>SUMA PLN 91,40</b>	NACZYNIE żARO 5,8L 1 x24,99 24,99A RECZ.KUCH. 804352 1 x4,79 4,79A LOYD ZI MATCHA 20P 1 x5,59 5,59A POMID.KOK.GALAZ.50 1 x7,49 7,49C D MARCHEW LUZ 0,126 x3,49 0,44C MASŁO EXTRA 200G 1 x4,99 4,99C LIPTON EARL GR.LEM 1 x5,98 5,98A CENOS.KASZ. GRYCZ. 1 x5,58 5,58C D CEBULÄ LUZ 0,210 x2,49 0,52C	
Rozpoznany przez Tesseract tekst : NACZYNIE żARO 5,8L 1 x24,99 24,99A			
Rozpoznany przez Tesseract tekst : RECZ.KUCH. 804352 1 x4,79 4,79A			
Rozpoznany przez Tesseract tekst : LOYD ZI MATCHA 20P x5,59 5,59A			
Rozpoznany przez Tesseract tekst : POMID.KOK.GALAZ.50 1 x7,49 7,49C			
Rozpoznany przez Tesseract tekst : D MARCHEW LUZ 0,126 x3,49 0,44C			
Rozpoznany przez Tesseract tekst : MASŁO EXTRA 200G 1 x4,99 4,99C			
Rozpoznany przez Tesseract tekst : LIPTON EARL GR.LEM 1x5,98 5,98A			
Rozpoznany przez Tesseract tekst : CENOS.KASZ. GRYCZ. 1 x5,58 5,58C			
Rozpoznany przez Tesseract tekst : D CEBULÄ LUZ 0,210 x2,49 0,52C			

Rysunek 0.9. Paragon6\_1.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO WWW.auchan.pl BDO 000013917 NIP 526-03-09-174	PRZECIER-Z ZIOŁ ŚR 1 x3,38 3,38C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> PRZECIER-Z ZIOŁ ŚR 1   3,38   3,38   C Rozpoznany przez Tesseract tekst : PRZECIER-Z ZIOŁ ŚR 1 x3,38 3,38C
2021-04-19	KUR PODUDZIE LUZ. 0,784 x7,48 5,86C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> KUR PODUDZIE LUZ. 0,784   7,48   5,86   C Rozpoznany przez Tesseract tekst : KUR PODUDZIE LUZ. 0,784 x7,48 5,86C
30761	PAPIER WC 40913 1 x6,99 6,99A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> PAPIER WC 40913 1   6,99   6,99   A Rozpoznany przez Tesseract tekst : PAPIER WC 40913 1 x6,99 6,99A
NACZYNIE ZARO 5,8L RĘCZ.KUCH. 804352 LOYD ZI MATCHA 20P POMID.KOK.GALAZ.50 D MARCHEW LUZ MASŁO EXTRA 200G LIPTON EARL GR.LEM CENOS.KASZ. GRYCZ. D CEBULA LUZ PRZECIER-Z ZIOŁ ŚR KUR PODUDZIE LUZ. PAPIER WC 40913 SZWAJCAR KRÓLEWIEC WODA.KINGA GAZOWAN COCA.COLA 1,5L PET	SZWAJCAR KRÓLEWIEC 0,366 x20,99 7,68C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> SZWAJCAR KRÓLEWIEC 0,366   20,99   7,68   C Rozpoznany przez Tesseract tekst : SZWAJCAR KRÓLEWIEC 0,366 x20,99 7,68C
1 x24,99 24,99A 1 x4,79 4,79A 1 x5,59 5,59A 1 x7,49 7,49C 0,126 x3,49 0,44C 1 x4,99 4,99C 1 x5,98 5,98A 1 x5,58 5,58C 0,210 x2,49 0,52C 1 x3,38 3,38C 0,784 x7,48 5,86C 1 x6,99 6,99A 0,366 x20,99 7,68C 1 x1,58 1,58A 1 x5,54 5,54A	WODA.KINGA GAZOWAN 1 x1,58 1,58A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> WODA.KINGA GAZOWAN 1   1,58   1,58   A Rozpoznany przez Tesseract tekst : WODA.KINGA GAZOWAN 1 x1,58 1,58A
----- SPRZEDAŻ OPODATK. A 55,46 PTU A 23,00 % 10,37 SPRZEDAŻ OPODATK. C 35,94 PTU C 5,00 % 1,71 SUMA PTU 12,08	COCA.COLA 1,5L PET 1 x5,54 5,54A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> COCA.COLA 1,5L PET 1   5,54   5,54   A Rozpoznany przez Tesseract tekst : COCA.COLA 1,5L PET 1 x5,54 5,54A
<b>SUMA PLN 91,40</b>	

Rysunek 0.10. Paragon6\_2.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	D CEBULA LUZ 0,690 x2,49 1,72C RĘCZ.KUCH. 804352 1 x4,79 4,79A FILET LUZ 0,892 x21,48 19,16C PRINCES.TUN KAW 17 1 x4,99 4,99C PRINCES.TUN KAW 17 1 x4,99 4,99C PAPIER WC 40913 1 x6,99 6,99A WODA.KINGA GAZOWAN 1 x1,58 1,58A PRZECIER-POMIDOR-P 1 x2,39 2,39C D BURAKI CZERW.910 0,922 x1,28 1,18C POMID.KOK.GALAZ.50 1 x7,49 7,49C KAPUSTA BIALA LUZ 1,744 x1,38 2,41C MAJON.LEKKI.300ML 1 x4,78 4,78B B KAWA PALON WANIL 0,126 x29,90 3,77A SER GRAN.BIR.PŁA.8 1 x5,99 5,99C ŚMIETANA 18% 2522 1 x2,68 2,68C KREMOWY ZE SŁON.10 1 x1,98 1,98C KREMOWY ZE SŁON.10 1 x1,98 1,98C KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A D MARCHEW LUZ 0,402 x1,99 0,80C SOK.Z CYTRYN 125ML 1 x3,39 3,39C FORMAGIA 180G 1 x10,48 10,48C D MARCHEW LUZ 0,164 x1,99 0,33C //RUKOLA 150G 1 x4,49 4,49C NOW-D CEBULA DYMKA 1 x2,49 2,49C ACTKA PIETRUSZ.PEC 1 x1,39 1,39C  ----- SPRZEDAŻ OPODATK. A 22,32 PTU A 23,00 % 4,17 SPRZEDAŻ OPODATK. B 4,78 PTU B 8,00 % 0,35 SPRZEDAŻ OPODATK. C 80,33 PTU C 5,00 % 3,83 SUMA PTU 8,35  <b>SUMA PLN 107,43</b>	D CEBULA LUZ 0,690 x2,49 1,72C Nazwa Ilość Cena Podsumowanie Typ podatku D CEBULA LUZ 0,69 2,49 1,72 C Rozpoznany przez Tesseract tekst : D CEBULA LUZ 0,690 x2,49 1,72C
RĘCZ.KUCH. 804352 1 x4,79 4,79A Nazwa Ilość Cena Podsumowanie Typ podatku RĘCZ.KUCH. 804352 1 4,79 4,79 A Rozpoznany przez Tesseract tekst : RĘCZ.KUCH. 804352 1x4,79 4,79A	FILET LUZ 0,892 x21,48 19,16C Nazwa Ilość Cena Podsumowanie Typ podatku FILET LUZ 0,892 21,48 19,16 C Rozpoznany przez Tesseract tekst : FILET LUZ 0,892 x21,48 19,16C	
PRINCES.TUN KAW 17 1 x4,99 4,99C Nazwa Ilość Cena Podsumowanie Typ podatku PRINCES.TUN KAW 17 1 4,99 4,99 C Rozpoznany przez Tesseract tekst : PRINCES.TUN KAW 17 1 x4,99 4,99C	PRINCES.TUN KAW 17 1 x4,99 4,99C Nazwa Ilość Cena Podsumowanie Typ podatku PRINCES.TUN KAW 17 1 4,99 4,99 C Rozpoznany przez Tesseract tekst : PRINCES.TUN KAW 17 1 x4,99 4,99C	
PAPIER WC 40913 1 x6,99 6,99A Nazwa Ilość Cena Podsumowanie Typ podatku PAPIER WC 40913 1 6,99 6,99 A Rozpoznany przez Tesseract tekst : PAPIER WC 40913 1 x6,99 6,99A	WODA.KINGA GAZOWAN 1 x1,58 1,58A Nazwa Ilość Cena Podsumowanie Typ podatku WODA.KINGA GAZOWAN 1 1,58 1,58 A Rozpoznany przez Tesseract tekst : WODA.KINGA GAZOWAN 1x1,58 1,58A	
PRZECIER-POMIDOR-P 1 x2,39 2,39C Nazwa Ilość Cena Podsumowanie Typ podatku PRZECIER-POMIDOR-P 1 2,39 2,39 C Rozpoznany przez Tesseract tekst : PRZECIER-POMIDOR-P 1 x2,39 2,39C	D BURAKI CZERW.910 0,922 x1,28 1,18C Nazwa Ilość Cena Podsumowanie Typ podatku D BURAKI CZERW.910 0,922 1,28 1,18 C Rozpoznany przez Tesseract tekst : D BURAKI CZERW.910 0,922 x1,28 1,18C	
POMID.KOK.GALAZ.50 1 x7,49 7,49C Nazwa Ilość Cena Podsumowanie Typ podatku POMID.KOK.GALAZ.50 1 7,49 7,49 C Rozpoznany przez Tesseract tekst : POMID.KOK.GALAZ.50 1 x7,49 7,49C	KAPUSTA BIALA LUZ 1,744 x1,38 2,41C Nazwa Ilość Cena Podsumowanie Typ podatku KAPUSTA BIALA LUZ 1,744 1,38 2,41 C Rozpoznany przez Tesseract tekst : KAPUSTA BIALA LUZ 1,744 x1,38 2,41C	
MAJON.LEKKI.300ML 1 x4,78 4,78B Nazwa Ilość Cena Podsumowanie Typ podatku		

Rysunek 0.11. Paragon7\_1.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	MAJON.LEKKI.300ML 1 x4,78 4,78B <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> MAJON.LEKKI.300ML   1   4,78   4,78   B Rozpoznany przez Tesseract tekst : MAJON.LEKKI.300ML 1 x4,78 4,78B
2021-03-27 38042	B KAWA PALON WANIL 0,126 x29,90 3,77A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> B KAWA PALON WANIL   0,126   29,90   3,77   A Rozpoznany przez Tesseract tekst : B KAWA PALON WANIL 0,126 x29,90 3,77A
PARAGON FISKALNY	SER GRAN.BIR.PŁA.8 1 x5,99 5,99C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> SER GRAN.BIR.PŁA.8   1   5,99   5,99   C Rozpoznany przez Tesseract tekst : SER GRAN.BIR.PŁA.8 1 x5,99 5,99C
D CEBULA LUZ 0,690 x2,49 1,72C RECZ.KUCH. 804352 1 x4,79 4,79A FILET LUZ 0,892 x21,48 19,16C PRINCES.TUN KAW 17 1 x4,99 4,99C PRINCES.TUN KAW 17 1 x4,99 4,99C PAPIER WC 40913 1 x6,99 6,99A WODA.KINGA GAZOWAN 1 x1,58 1,58A PRZECIER-POMIDOR-P 1 x2,39 2,39C D BURAKI CZERW.910 0,922 x1,28 1,18C POMID.KOK.GALAZ.50 1 x7,49 7,49C KAPUSTA BIALA LUZ 1,744 x1,38 2,41C MAJON.LEKKI.300ML 1 x4,78 4,78B B KAWA PALON WANIL 0,126 x29,90 3,77A SER GRAN.BIR.PŁA.8 1 x5,99 5,99C ŚMIETANA 18% 2522 1 x2,68 2,68C KREMOWY ZE SLON.10 1 x1,98 1,98C KREMOWY ZE SLON.10 1 x1,98 1,98C KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A D MARCHEW LUZ 0,402 x1,99 0,80C SOK.Z CYTRYN 125ML 1 x3,39 3,39C FORMAGIA 180G 1 x10,48 10,48C D MARCHEW LUZ 0,164 x1,99 0,33C //RUKOLA 150G 1 x4,49 4,49C NOW-D CEBULA DYMKA 1 x2,49 2,49C ACTKA PIETRUSZ.PEC 1 x1,39 1,39C	
-----	KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> KAWA Z ORZ LAS LUZ   0,13   39,90   5,19   A Rozpoznany przez Tesseract tekst : KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A
D MARCHEW LUZ 0,402 x1,99 0,80C SOK.Z CYTRYN 125ML 1 x3,39 3,39C FORMAGIA 180G 1 x10,48 10,48C D MARCHEW LUZ 0,164 x1,99 0,33C //RUKOLA 150G 1 x4,49 4,49C	KREMOWY ZE SLON.10 1 x1,98 1,98C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> KREMOWY ZE SLON.10   1   1,98   1,98   C Rozpoznany przez Tesseract tekst : KREMOWY ZE SLON.10 1x1,98 1,98C
SPRZEDAŻ OPODATK. A 22,32 PTU A 23,00 % 4,17 SPRZEDAŻ OPODATK. B 4,78 PTU B 8,00 % 0,35 SPRZEDAŻ OPODATK. C 80,33 PTU C 5,00 % 3,83 SUMA PTU 8,35	KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> KAWA Z ORZ LAS LUZ   0,13   39,90   5,19   A Rozpoznany przez Tesseract tekst : KAWA Z ORZ LAS LUZ 0,130 x39,90 5,19A
<b>SUMA PLN 107,43</b>	FORMAGIA 180G 1 x10,48 10,48C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> FORMAGIA 180G   1   10,48   10,48   C Rozpoznany przez Tesseract tekst : FORMAGIA 180G 1 x10,48 10,48C
	D MARCHEW LUZ 0,164 x1,99 0,33C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b> D MARCHEW LUZ   0,164   1,99   0,33   C Rozpoznany przez Tesseract tekst : D MARCHEW LUZ 0,164 x1,99 0,33C
	//RUKOLA 150G 1 x4,49 4,49C <b>Nazwa Ilość Cena Podsumowanie Typ podatku</b>

Rysunek 0.12. Paragon7\_2.png

HIPERMARKET AUCHAN 31-346 KRAKÓW UL. STAWOWA 61 TEL. (22) 444 02 22 AUCHAN POLSKA SP.Z O.O. UL.PUŁAWSKA 46 05-500 PIASECZNO www.auchan.pl BDO 000013917 NIP 526-03-09-174	
2021-03-27	38042
PARAGON FISKALNY	
D CEBULA LUZ	0,690 x2,49 1,72C
RĘCZ.KUCH. 804352	1 x4,79 4,79A
FILET LUZ	0,892 x21,48 19,16C
PRINCES.TUN KAW 17	1 x4,99 4,99C
PRINCES.TUN KAW 17	1 x4,99 4,99C
PAPIER WC 40913	1 x6,99 6,99A
WODA.KINGA GAZOWAN	1 x1,58 1,58A
PRZECIER-POMIDOR-P	1 x2,39 2,39C
D BURAKI CZERW.910	0,922 x1,28 1,18C
POMID.KOK.GALAZ.50	1 x7,49 7,49C
KAPUSTA BIALA LUZ	1,744 x1,38 2,41C
MAJON.LEKKI.300ML	1 x4,78 4,78B
B KAWA PALON WANIL	0,126 x29,90 3,77A
SER GRAN.BIR.PŁA.8	1 x5,99 5,99C
ŚMIETANA 18% 2522	1 x2,68 2,68C
KREMOWY ZE SŁON.10	1 x1,98 1,98C
KREMOWY ZE SŁON.10	1 x1,98 1,98C
KAWA Z ORZ LAS LUZ	0,130 x39,90 5,19A
D MARCHEW LUZ	0,402 x1,99 0,80C
SOK.Z CYTRYN 125ML	1 x3,39 3,39C
FORMAGIA 180G	1 x10,48 10,48C
D MARCHEW LUZ	0,164 x1,99 0,33C
//RUKOLA 150G	1 x4,49 4,49C
NOW-D CEBULA DYMKA	1 x2,49 2,49C
ACTKA PIETRUSZ.PEC	1 x1,39 1,39C
-----	-----
SPRZEDAŻ OPODATK. A	22,32
PTU A 23,00 %	4,17
SPRZEDAŻ OPODATK. B	4,78
PTU B 8,00 %	0,35
SPRZEDAŻ OPODATK. C	80,33
PTU C 5,00 %	3,83
SUMA PTU	8,35
<b>SUMA PLN</b>	<b>107,43</b>

Rysunek 0.13. Paragon7\_3.png