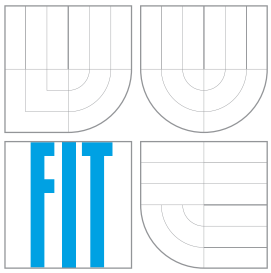
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROZPOZNÁVÁNÍ TEXTU V OBRAZE

OPTICAL CHARACTER RECOGNITION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VÁCLAV SUCHÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. MICHAL ŠPANĚL

BRNO 2007

Zadání

- Prostudujte základy zpracování obrazu. Zaměřte se zejména na problematiku detekce hran a elementárních objektů v obraze.
- Zorientujte se v současných metodách rozpoznávání textu v obraze (tzv. Optical Character Recognition, nebo-li OCR).
- Vyberte vhodnou metodu a navrhněte jednoduchý rozpoznávač textu v obraze.
- Experimentujte s vaší implementací a případně navrhněte vlastní modifikace metod.
- Porovnejte dosažené výsledky a diskutujte možnosti budoucího vývoje. Zvažte další pokračování v rámci diplomové práce.
- Vytvořte stručný plakát prezentující vaši bakalářskou práci, její cíle a výsledky.

Licenční smlouva

Abstrakt

Tato práce se zabývá problematikou rozpoznávání textu v obraze. Rozebírá metody rozpoznávání, jejich úspěchy, klady a zápory. Popisuje postup při návrhu a implementaci jednoduchého ukázkového programu pro rozpoznávání strojově tištěného textu s využitím neuronových sítí.

Klíčová slova

Rozpoznávání textu, OCR, segmentace textu, umělé neuronové sítě.

Abstract

This paper describes problems of text recognition in picture. Discuss successes, advatages and disadvantages several methods of recognition. In second part there is described design and implementation of a simple OCR software for typewritten text recognition by using artificial neural networks.

Keywords

Optical character recognition, OCR, text segmentation, ANN, Artificial Neural Networks

Citace

Václav Suchý: Rozpoznávání textu v obraze, bakalářská práce, Brno, FIT VUT v Brně, 2007

Rozpoznávání textu v obraze

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Michala Španěla

.....
Václav Suchý
14. května 2007

Poděkování

Rád bych poděkoval Ing. Michalovi Španělovi za jeho cenné rady během celého projektu.

Obsah

Kapitola 1

Úvod

S rozvojem moderních technologií posledních desetiletí 20. století došlo i k modernizaci přístupu k psanému a tištěnému textu. Psaný text byl postupně nahrazován textem elektronickým, který má oproti textu „na papíře“ řadu nesporných výhod. Mezi jeho zásadní přednosti lze zařadit jeho velmi jednoduchou upravitelnost, snadný převod do papírové podoby, schopnost provazovat text navzájem pomocí tzv. hypertextových odkazů, možnost elektronický text přečíst prostřednictvím vhodného software bez přítomnosti fyzického mluvčího apod. Moderní elektronickou komunikaci, posílání emailů, SMS, tvorbu elektronických knih (e-books), vývoj software i hardware si dnes již bez elektronického textu nelze představit. Přesto má oproti svému papírovému předchůdci jednu nespornou nevýhodu, existuje tu pouze několik desítek let.

OCR je zkratkou anglického slova Optical Character Recognition – Optické rozpoznávání znaku. Jedná se o *automatickou identifikaci grafických znaků snímaných opticky*, viz [11], případně o metodu *elektronického čtení tištěných znaků a jejich převádění do digitální formy, které lze dále zpracovat počítačem*, viz [8].

Toto téma bakalářské práce jsem si vybral „ze zvědavosti“, neboť jsem se chtěl o této oblasti dozvědět více.

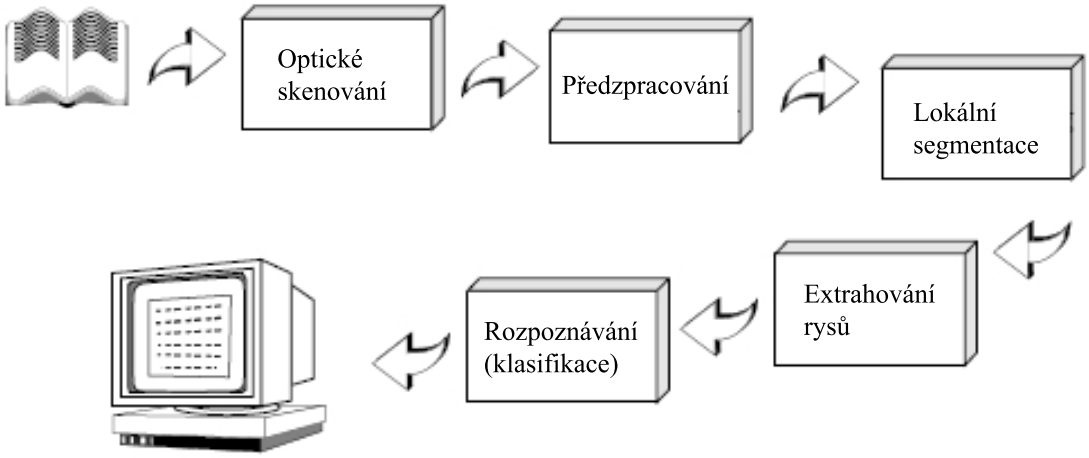
Druhá a třetí kapitola se zabývá teorií některých metod použitých v ukázkovém programu. Další kapitola stručně pojednává o jiných metodách používaných v OCR - systémech. Pátá a šestá kapitola potom popisuje návrh a implementaci jednoduchého programu pro rozpoznávání strojopisného textu v obraze. Dosažené výsledky jsou zmíněny v kapitole sedmé.

Bakalářská práce je inspirována několika různými pracemi podobného zaměření, avšak na žádnou přímo nenavazuje.

1.1 OCR systémy

Prvním krokem v automatickém rozpoznávání vzorů je „naučit“ stroj, jaké jsou třídy vzorů a jak vypadají. Vzory v OCR mohou být písmena, číslice a některé speciální symboly jako čárka, otazník, vykřičník atd. Učení probíhá formou ukázek příkladů vzorů všech tříd, přičemž si stroj vytvoří prototyp, popis každé třídy. Během rozpoznávání jsou neznámé znaky porovnány s natrénovanými popisy a podle nejlepší shody jsou přiřazeny do patřičné třídy.

Typické OCR systémy se většinou skládají z několika komponent (obr. 1.1). V prvním kroku jsou analogové dokumenty digitalizovány pomocí optických skenerů. Poté jsou lokalizovány oblasti textu a segmentovány jednotlivé znaky. V kroku předzpracování jsou jednotlivé znaky zbaveny šumů a jiných nežádoucích efektů, které mohly vzniknout během skenování. Dále přichází na řadu extrahování rysů jednotlivých znaků a jejich následné zařazování do tříd a rozpoznávání. Nakonec je ze získaných informací složen výsledný text.



Obrázek 1.1: Typické komponenty OCR systémů

Kapitola 2

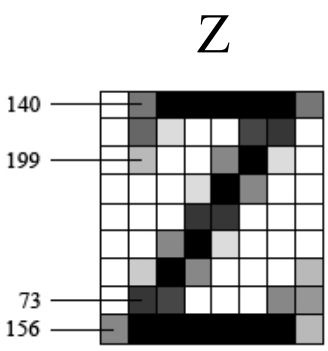
Zpracování obrazu

2.1 Co je to obraz

Oblast Zpracování obrazu se zabývá analyzováním tzv. digitálního obrazu. Ten lze získat například digitalizací analogového signálu z kamery, nebo přímo jako digitální signál z CCD kamery nebo ze skeneru.

Obraz je tvořen konečným počtem obrazových elementů zvaných pixely. Každý pixel nese informaci o své barevné hloubce, která je reprezentována jednou celočíselnou hodnotou nebo vektorem těchto hodnot. Pixely jsou pak většinou uloženy ve dvourozměrné mřížce zvané bitmapa nebo také rastr.

Rozpoznávat znaky můžeme například z 8 bit bitmapy, kde každý pixel nese diskrétní hodnotu 0-255 (2^8), tedy intenzitu bílé barvy (obr. 2.1).



Obrázek 2.1: Ukázka uložení znaku Z do 8 bit bitmapy

2.2 Předzpracování

Zdrojový obrázek může obsahovat různé nežádoucí efekty jako je např. šum, nebo rozpadlá písmenka. Některé tyto chyby mohou být odstraněny aplikováním filtrů – např. vyhlazením. Proces vyhlazení obsahuje vyplňování i zužování. Vyplňování eliminuje malé mezery a díry ve znacích, zatímco zužování celkově ztenčuje rozpoznávané znaky.

Předzpracování obvykle obsahuje i proces normalizace (obr. 2.2), po jehož aplikování dostáváme znak v jednotné velikosti, sklonu a rotaci. K získání úhlu rotace se většinou

používají některé varianty Houghovi transformace, viz [4].

Obrázek 2.2: Normalizace a vyhlazování znaků

Rotace ve 2D

Otočení se provádí pro každý pixel zvlášť podle počátku souřadnicového systému o uhel α (obr. 2.3). Postup:

1. Souřadnice bodu $B = [x, y]$ převedeme do tzv. homogenního tvaru na $B = [x, y, w]$, kde w je tzv. váha (většinou dáváme 1) a touto váhou se vynásobí zbylé souřadnice x a y.

2. Použijeme transformační matici

$$\mathbf{R} = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3. Nové souřadnice bodu $B' = [x', y', w']$ vypočítáme vynásobením $B' = B \cdot \mathbf{R}$

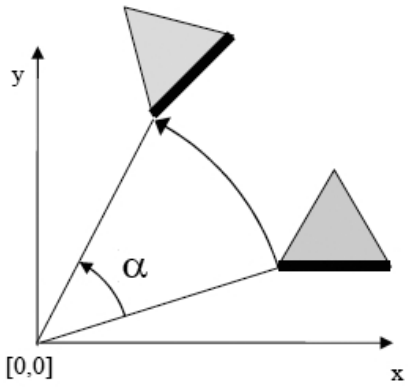
Pokud chceme rotovat okolo libovolně zvoleného středu $S[x, y]$, musíme použít metodu *Posun – Rotace – Posun* pomocí transformační matice pro posunutí:

$$\mathbf{P} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \delta x & \delta y & 1 \end{bmatrix}$$

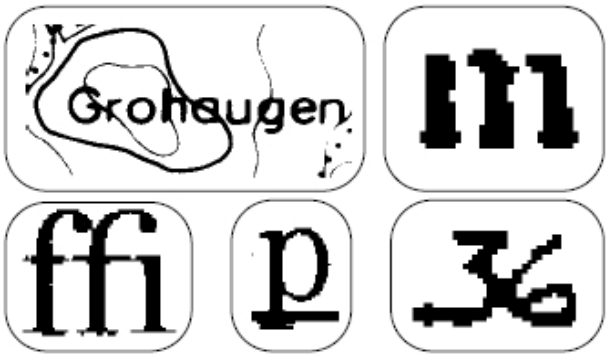
2.3 Lokalizace a segmentace

Lokalizace je proces, při kterém je zkoumáno rozložení textu na stránce. Je nezbytné lokalizovat oblasti s textem a oddělit je od obrázku a grafiky. Například u systému třídění pošty musí být adresa oddělena od ostatní grafiky jako jsou známky, loga společností apod.

Po lokalizaci přichází na řadu segmentace, při které jsou izolovány jednotlivé znaky z textu. Obvykle je segmentace založena na principu sledování spojitých komponent, tj. každá spojená tmavá oblast. Tato technika je snadno implementovatelná, avšak problémy nastávají, pokud se znaky navzájem dotýkají nebo naopak jsou rozpadlé na několik částí (obr. 2.4).



Obrázek 2.3: Otočení obrázku podle počátku souřadnic o úhel α



Obrázek 2.4: Příklady znehodnocených znaků

2.4 Binární obraz – prahování

Pro následnou extrakci příznaků pomocí momentů (viz 3.1) potřebujeme obraz v tzv. binární podobě, kde každý pixel nese jednobitovou informaci – 1/0 (bílá/černá barva). Binární obraz můžeme získat například pomocí prahování. Prahování je metoda, kdy se intenzita jednotlivých pixelů převede na maximální resp. minimální hodnotu. Algoritmus pro prahování 8 bit bitmapy (alg. 1):

Algorithm 1 Prahování

Require: práh $t \in \langle 0, 255 \rangle$
  for all pixely p v bitmapě **do**
    if intenzita pixelu $p > t$ **then**
      intenzita pixelu $p \leftarrow 255$
    else
      intenzita pixelu $p \leftarrow 0$
    end if
  end for

Kapitola 3

Klasifikace vzorů

3.1 Extrakce příznaků

Obvykle nejproblematičtější část OCR systémů. Úkolem je získání základních charakteristik každého znaku. Většina metod popisuje znak přímo z rastru obrázku. Jiné metody získávají určité rysy charakterizující znak, ale nevšímají si nedůležitých atributů.

Jedna z používaných technik při extrakci příznaků je použití *momentů* – za rys znaku je brán moment tmavých bodů, např. vzhledem ke středu gravitace nebo k vybranému počátku souřadnic.

HU momenty

Hu momenty jsou sada absolutně ortogonálních (včetně rotace) momentových invariantů, které mohou být použity pro rozpoznávání nezávislém na měřítku, pozici a rotaci rozpoznávaného vzoru. Hu, viz [10], je definoval pomocí normalizovaných centrálních momentů do třetího řádu, viz [12]. Sedm Hu momentů:

$$
\begin{aligned}
I_1 &= \eta_{20} + \eta_{02} \\
I_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
I_3 &= (\eta_{30} + 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
I_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
I_5 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad +(3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
I_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}))] \\
I_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + \\
&\quad +(\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned}
$$

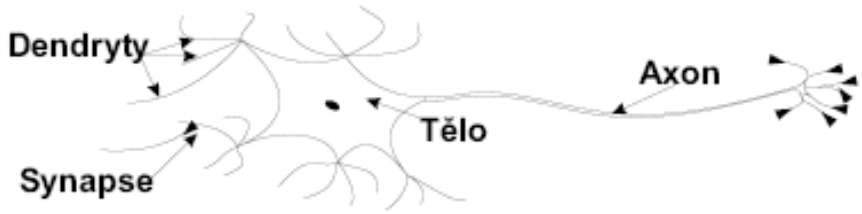
3.2 Neuronové sítě

Umělé neuronové sítě si ve své funkci kladou za vzor neuronové sítě biologické.

Biologická neuronová síť

Lidský mozek se skládá z asi 10^{11} výpočetních elementů zvaných neurony, které spolu komunikují prostřednictvím sítě vazeb. Vstup do sítě je umožněn smyslovými receptory, podle výsledného zpracování jsou řízeny efektory.

Neuron (obr. 3.1) se skládá ze tří hlavních částí: Tělo, z něhož vychází jeden poměrně dlouhý výstup – axon, a množství dendritů, které tvoří vstup neuronu. Dendrity se s axony sousedních neuronů stýkají prostřednictvím rozhraní – tzv. synapse. Přenášené signály jsou elektrické impulsy, jejichž přenos je ovlivněn uvolňováním chemických látek v synapsích. Tyto chemické látky působí jako tzv. excitátory, pokud umožňují následujícímu neuronu generovat impuls, nebo jako tzv. inhibitory, pokud naopak schopnost generovat snižují. Aktivace neuronu nastane tehdy, překročí-li hodnota budících vstupních signálů hodnotu tlumících signálů o určitou hodnotu.



Obrázek 3.1: Stavba neuronu biologické sítě

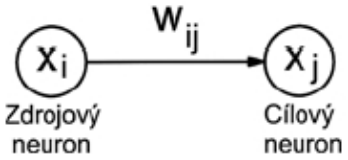
Modelování neuronů

Podobně jako u biologických neuronových sítí, tak i u umělých, je základní stavební jednotkou neuron. Modely umělých neuronů jsou určitým způsobem seskupeny a mezi sebou vzájemně propojeny, čímž tvoří umělou neuronovou síť. Každá neuronová síť jako celek realizuje určitou, pro ni typickou transformační funkci - převádí (transformuje) hodnoty vstupních veličin na hodnoty veličin výstupních. Často jsme postaveni před problém, že daný proces není možné s uspokojivou přesností matematicky popsat, nebo exaktní matematický model procesu je tak složitý, že jeho případná algoritmizace je buď časově a programově velmi náročná nebo dokonce nemožná.

Propojení dvou na sebe navazujících neuronů je uvedeno na obr. 3.2. Podle umístění těchto neuronů dělíme neurony na zdrojové (presynaptické - před synapsí) a na cílové (postsynaptické - po synapsi). Váhy spojení mezi jednotlivými neurony jsou označeny symbolem w_{ij}. Příslušné indexy charakterizují spojení od i-tého zdrojového neuronu k j-tému cílovému neuronu.

Topologie (stavba) sítí

Společným rysem všech topologií umělých neuronových sítí je většinou jejich vrstevnatá struktura:



Obrázek 3.2: Propojení neuronů

- *vstupní vrstva* – je tvořena tzv. zdrojovými uzly, slouží ke vstupu určitého signálu z okolí a jeho rozdělení do neuronů následující vrstvy. Lze ji chápat jako pasivní prvky s jednotkovým přenosem, které nemají vlastnosti klasického neuronu.
- jedna nebo více *skrytých vrstev* – neurony transformují vstupy z předchozí vrstvy do následující. Úkolem skrytých vrstev je zvýšení aproximačních vlastností neuronové sítě jako celku.
- *výstupní vrstva* – jejíž neurony předávají výstupní signály ze sítě do okolí. Tyto signály jsou pak odezvou neuronové sítě na signály vstupní.

Podle toku signálu neuronovou sítí je dělíme na:

- *dopředné* – signál se v nich šíří po orientovaných spojeních jen jedním směrem, tzn. od vstupní vrstvy k výstupní.
- *rekurentní (zpětnovazební)* – mezi neurony nebo vrstvami existují navíc zpětné vazby. V těchto sítích se někdy těžko definuje vstupní a výstupní vrstva.

Proces učení umělé neuronové sítě

Základní a velmi podstatnou vlastností umělé neuronové sítě je její schopnost učení. Proces učení představuje dynamický proces, při kterém dochází k modifikaci vhodných, nastavitelných parametrů příslušné neuronové sítě za účelem dosažení požadované shody mezi výstupy z modelované soustavy a výstupy z neuronové sítě. V drtivé většině případů se proces učení soustřeďuje na adaptaci vah spojení mezi neurony. Někdy mohou být nastavitelnými parametry také strmosti aktivačních funkcí nebo postupná změna struktury neuronové sítě. Proces učení bývá charakterizován určitým algoritmem učení. Algoritmus učení obecně určuje, jakým způsobem dochází ke změně nastavitelných parametrů neuronové sítě. Algoritmus tedy představuje určitou strategii, která je kombinována určitými konkrétními matematickými operacemi.

Topologie perceptronové BackPropagation neuronové sítě

Perceptronové neuronové sítě patří mezi nejznámější a v praxi nejpoužívanější neuronové sítě. Topologie může být tvořena buď jedním výkonným prvkem – neuronem (jednoduchý perceptron), nebo více neurony (vícevrstvý perceptron). Základním výkonným prvkem perceptronových sítí je model neuronu s lineárně váženou agregační funkcí a skokovou aktivační

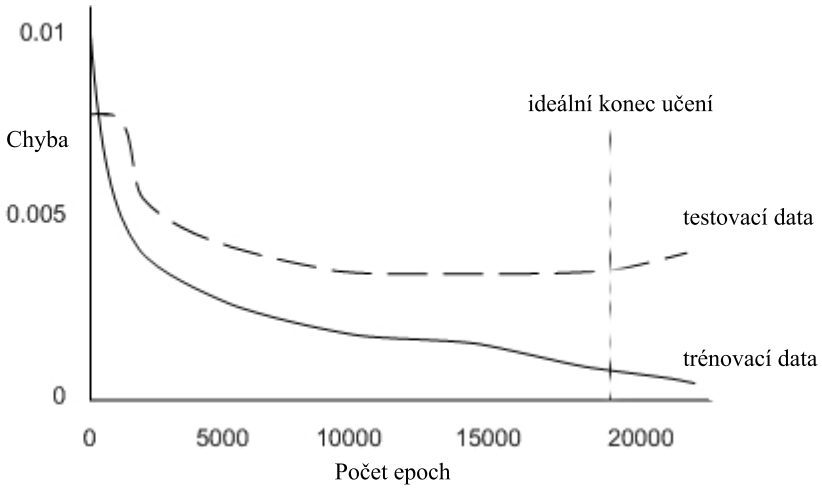
funkcí. Požadovaná funkce neuronové sítě je zadána trénovací množinou ve tvaru dvojic vektorů – vektoru vstupních vzorů a vektoru požadovaných (cílových) hodnot. Výstup z perceptronu je počítán podle následující rovnice:

$$y = \begin{cases} 0 \quad pro \quad \sum_{i=0}^{n} w_i x_i < 0 \\ 1 \quad pro \quad \sum_{i=0}^{n} w_i x_i \geq 0 \end{cases}$$

Neuronová síť BackPropagation je vícevrstvá perceptronová dopředná síť, jejíž váhy při učení, viz [7], jsou nastavovány na základě zpětného šíření chyby.

MSE – Mean square error

Chyba udává míru rozdílu mezi hodnotami výstupního vektoru sítě a požadovanými hodnotami pro všechna trénovací data. Počítá se po každé epoše (cyklu) během trénování. Čím je nižší, tím je síť více naučena na daný vzorek trénovacích dat. Avšak příliš intenzivní učení může vést až k tzv. přeučení, kdy síť perfektně rozpoznává trénovací data, ale ztrácí schopnost generalizace. Proto se většinou při trénování počítá i MSE dat, které síť ještě „neviděla“. Podle průběhu lze určit, kdy MSE neznámých dat začíná růst. V tomto místě by se mělo přestat s učením sítě, aby nedošlo k přetrénování (obr. 3.3)



Obrázek 3.3: Průběh chyby při učení

Kapitola 4

Současný stav

4.1 Historie

Už v roce 1929 obdržel Gustav Tauschek patent na OCR v Německu. Jeho stroj byl mechanickým zařízením využívající vzory. Na každý znak byly postupně přikládány vzory se znaky opačného výřezu, na ně bylo posvíceno, a pokud se vzor s písmenem dokonale kryli (na fotodetektor nedopadalo žádné světlo), znak byl rozpoznán. V 50. letech se v USA objevily první systémy založené na obrazové analýze, které byly schopny akceptovat i několik variant fontů. OCR se začíná komerčně rozšiřovat. Od 60. let se OCR začalo hojně využívat například v poštovních službách na třídění pošty.

4.2 Současnost

V dnešní době můžeme rozlišit několik oblastí rozpoznávání textu, viz [1]:

- *Strojopisný text* a problém s jeho rozpoznáváním se v nešní době považuje za téměř vyřešený. Některé systémy jsou schopny rozpoznávat znaky až s 99% úspěšností. I přes velkou přesnost rozpoznávání je v některých oborech potřeba osobní kontrola případných chyb.

- *Ručně psaný text* je objektem dnešních výzkumů. Celkem úspěšně se daří tzv. „rozpoznávání za letu“, kdy se znaky rozpoznávají přímo při psaní. Algoritmy v těchto systémech využívají výhody, že dopředu znají pořadí, směr a rychlost psaní jednotlivých tvarů znaků. Tento princip se bohužel nedá využít při rozpoznávání naskenovaného textu, kde výše zmíněné výhody nelze využít. Přestože můžeme dosáhnout kolem 80% úspěšnosti při rozpoznávání čistě napsaného textu, stále vzniká mnoho chyb a proto se tato technologie dá použít jen zřídka. Tato oblast OCR se také někdy označuje ICR – Intelligent Character Recognition.

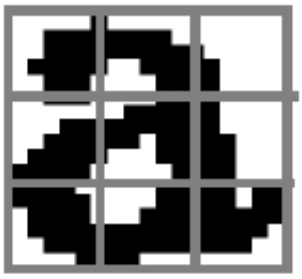
- *Kurzíva*, taktéž objekt dnešních výzkumů. Úspěšnost rozpoznávání je dokonce menší než při ručně psaném textu. Vyšší úspěšnosti se dá dosáhnout pouze za využití kontextových a gramatických informací. Například rozpoznání celých slov je jednodušší než zkoušení segmentovat jednotlivá písmenka z textu.

4.3 Další používané metody OCR

Extrahování rysů

Extrahovací techniky jsou většinou rozděleny do dvou hlavních skupin:

- *Rozložení bodů* – tato technika je založena na statistickém rozložení bodů v mřížce. Většinou je tolerantní k deformaci a variaci stylu znaku. Příklady technik:
  - *Rozdělení do pásem (Zoning)* – obrázek se znakem je rozdělen na několik překrývajících se nebo nepřekrývajících oblastí. Rys znaku je pak dán hustotou tmavých míst v jednotlivých regionech (obr. 4.1).
  - *Průsečíky (Crossings)* – tato technika, často využívána v komerčních systémech, je založena na počtu průsečíku znaku s předem zvolenými vektory (4.2).



Obrázek 4.1: Rozdělení do pásem (Zoning)

Obrázek 4.2: Metoda průsečíků

- *Strukturální analýza* – během analýzy je popisována geometrická a topologická struktura znaku. Hledají se základní prvky (úsečky, oblouky, smyčky, koncové body) a vzdálenosti mezi nimi. V porovnání s ostatními technikami je strukturální analýza mnohem více tolerantní k šumu a stylu znaku, méně však už k rotaci. Bohužel tato metoda nepatří mezi triviální, a některé části stále zůstávají v oblasti výzkumu.

Klasifikace

Příklad dalších používaných technik:

- *Shoda (Matching)* – skupina metod založená na stupni podobnosti, kdy se počítá vzdálenost mezi vektorem rysu rozpoznávaného znaku a vektorem popisující třídu. Základem je Euklidovská vzdálenost, viz [5].
- *Adaboost*, viz [2], *SVM - Support Vector Machine*, viz [3] – další příklady používaných klasifikátorů.

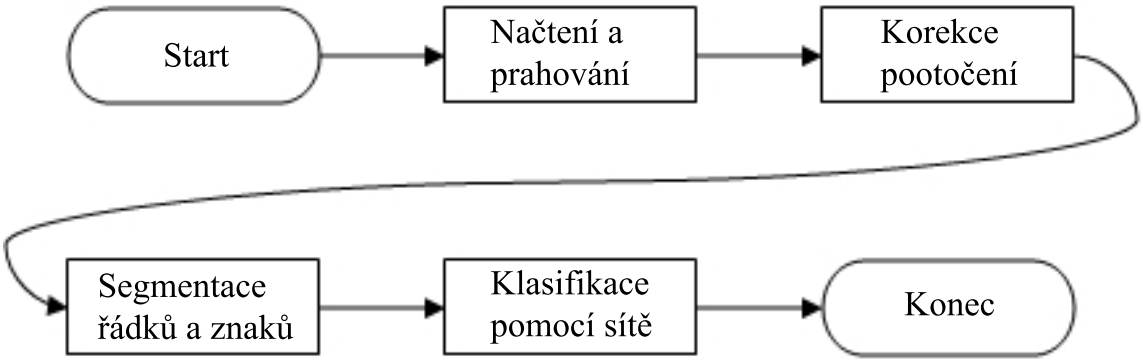
Více o obecné teorii OCR systémů, viz [9].

Kapitola 5

Návrh

Tato kapitola popisuje návrh jednoduchého programu na rozpoznávání textu z obrázku - velmi jednoduchý OCR systém.

Celý program jsem rozdělil na několik podproblémů, jak popisuje obrázek 5.1.

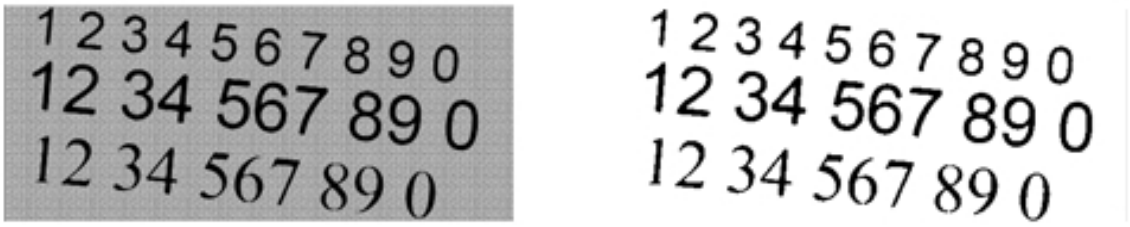


Obrázek 5.1: Jednotlivé části programu

5.1 Načtení a prahování

Obrázek s textem je načten jako 8 bit bitmapa, tedy obrázek ve stupních šedi, každý pixel obsahuje diskrétní hodnotu 0-255.

Pro další zpracování je však potřeba, aby byl obrázek v binarizovaném stavu, tedy aby obsahoval pouze minimální a maximální hodnoty (pouze černá a bílá barva), obr. 5.2. Za tímto účelem je provedeno jednoduché prahování (viz teorie, str. 6). Práh je standardně nastaven na hodnotu 160. Z toho plyne nevýhoda, že pro obrázky s menším kontrastem mezi textem a pozadím by tato hodnota nemusela stačit. Proto lze tuto hodnotu ručně měnit. Další možností by bylo použití adaptivního prahování, které bere v úvahu i okolí prahovaného bodu.



Obrázek 5.2: Načtený obrázek a výsledek následného prahování

5.2 Korekce pootočení

Protože při skenování může dojít k pootočení celého textu, je aplikována jednoduchá metoda pro zjištění úhlu pootočení.

Obrázek je protnut několika vektory určitého úhlu a je spočítán počet vektorů, které neprotínají žádné písmenko. Tento počet je uložen. Poté je zvolen jiný úhel vektorů a výpočet se opakuje. Po „vyzkoušení“ několika různých úhlů je vybrán ten, který měl nejvíce vektorů neprotínajících žádný text (obr. 5.3). Tato jednoduchá metoda dobře funguje jen pro obrázky, kde je text po celém obrázku, kde nejsou velké prázdné plochy mezi textem. Možné řešení, v obrázku nejprve detekovat oblast textu a poté metodu použít pouze v této oblasti.

Ukázka pseudokódu pro detekci úhlu v rozmezí -5 až 5 stupňů (alg. 2) použitého v programu.

Algorithm 2 Detekce úhlu pootočení

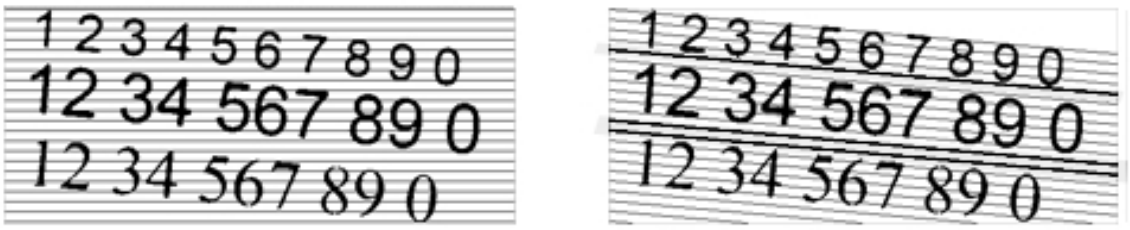
```
for uhel z −5 ≤ uhel ≤ 5 krok 0.5 stupňů do
  for y z 0 ≤ y ≤ vyskaObrazku krok 5 pixelů do
    projdi vektor z bodu[0,y] pod úhlem uhel a spočítej počet p tmavých míst
    if p ≤ tolerance then
      počet prázdných vektorů pv = pv + 1
    end if
  end for
  do asociativního pole pole ulož uhel a příslušný počet prázdných vektorů pv
end for
return  v pole najdi MAX(pv) a vrať příslušný uhel
```

Po zjištění úhlu je provedeno jednoduché otočení celého obrázku pomocí transformační matice, viz str. 5.

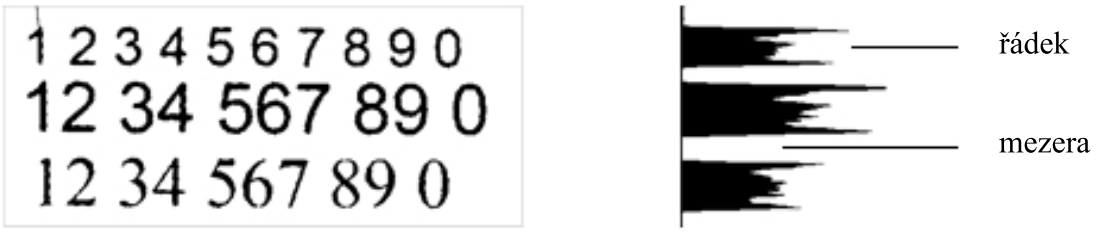
5.3 Segmentace řádků a znaků

Po prahování a případné rotaci je obrázek připraven na segmentaci jednotlivých řádků a znaků. Segmentace probíhá na základě histogramu.

Segmentace řádků využívá horizontální histogram, ten udává počet tmavých bodů na jednotlivých řádcích bitmapy. Oblasti s menším počtem tmavých míst jsou brány jako mezery, ostatní oblasti jako řádky (obr. 5.4).



Obrázek 5.3: Zjištění pootočení, testovány různé úhly



Obrázek 5.4: Segmentace řádků pomocí histogramu

Segmentace znaků na řádku probíhá stejným způsobem, tentokráte pomocí horizontálního histogramu pro každý řádek. Zároveň se počítá i průměrná šířka znaků, která je později využita pro rozpoznání mezery mezi slovy.

5.4 Extrakce příznaků (rysů)

Jednotlivé obrázky znaků jsou zmenšeny na jednotnou velikost a připraveny na extrakci. Pro popis znaků jsem zvolil Hu momenty. Těchto sedm momentů by mělo být invariantních vůči měřítku a rotaci. Způsob výpočtu viz str. 7. Momenty jsou přivedeny na vstup neuronové sítě.

5.5 Trénovací data

Jako trénovací data jsem chtěl použít sadu obrázků písmenek latinské abecedy. Bohužel, nepodařilo se mi na internetu najít žádnou volně dostupnou databázi obrázků. Proto jsem si musel vytvořit malý program na generování písmenek do obrázků. Obrázky jsou pak převedeny na trénovací soubor obsahující vektor Hu momentů každého písmenka a k němu odpovídající vektor identifikující znak.

5.6 Topologie sítě

První návrh

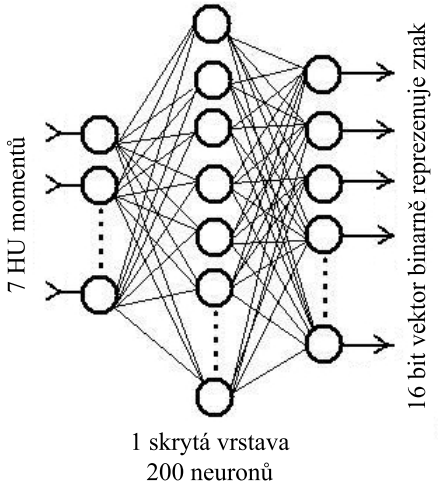
Původní představa byla taková, že by program měl rozpoznávat předem neurčený počet různých znaků, teoreticky všechny znaky sady UTF. První struktura sítě byla tvořena 7 neurony na vstupu, několika neurony ve skryté vrstvě (záleží na počtu trénovacích dat) a jen jedním neuronem ve výstupní vrstvě, který nabýval hodnoty 0 až teoretických 65 536. Hned po prvních pokusech s trénováním bylo jasné, že perceptronová neuronová síť BackPropagation (viz str. 9) si s touto topologií neporadí.

Souhrn:

vstupní vrstva	–	7 neuronů – Hu momenty
skryté vrstvy	–	jedna skrytá vrstva
výstupní vrstva	–	jeden neuron - hodnoty 0 - 65 536 (teoreticky) pro každý znak

Druhý návrh

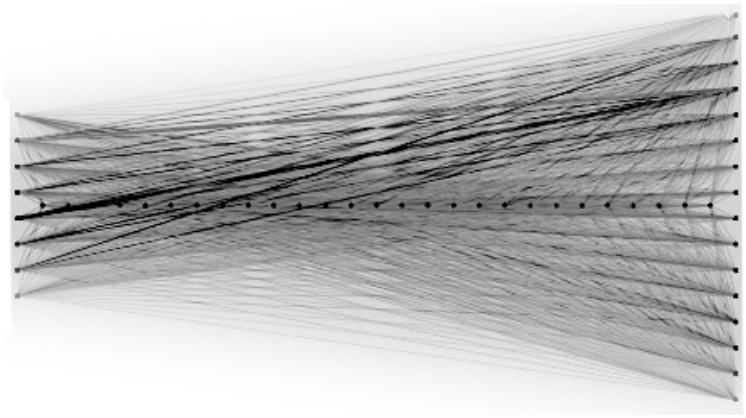
Při vytváření druhé sítě jsem se nechal inspirovat článkem Daniela Admassua, viz [6], který jako výstupní vrstvu použil 16 neuronů. Každý neuron nabývá hodnot 0 a 1 a výsledný vektor udává číslo znaku v binární hodnotě. Topografie druhé sítě je na obrázku 5.5.



Obrázek 5.5: Návrh topologie sítě číslo 2

Výsledky učení však nebyly o nic lepší, síť stále nereagovala na dané vstupy a výstupy. Zkoušel jsem nejrůznější kombinace aktivačních funkcí v neuronech, různé učící metody, rychlosti učení. První náznaky učení se objevily až při použití topologie evoluční (kaskádová) místo topologie pevné (obr. 5.6). Síť na začátku obsahuje pouze vstupní a výstupní vrstvu. Skryté vrstvy se dodávají až při učení, každá obsahuje pouze jeden neuron.

Síť se naučila trénovací data na požadovanou chybu a zdálo se, že jsem konečně na správné cestě. Avšak síť byla přetrénovaná, nebyla schopna generalizace, jinými slovy, uměla rozpoznat pouze obrázky znaků, na kterých byla trénována. Mírně odlišný rozměr či posunutí vedlo k neschopnosti znak rozpoznat. Proto jsem se vrátil k pevné topologii, jako na začátku. Věřil jsem, že úspěch bude záviset na správném nastavení sítě.



Obrázek 5.6: Topologie sítě po kaskádním učení

Souhrn:

vstupní vrstva	–	7 neuronů – Hu momenty
skryté vrstvy	–	27 skrytých vrstev, každá obsahuje jeden neuron
výstupní vrstva	–	16 neuronů, každý nese hodnotu 1 nebo 0, výsledný vektor udává binární číslo reprezentující každý znak

Třetí návrh

Tato struktura perceptronové sítě je stejná jako v předchozím pokusu, akorát jsem snížil výstupní vektor z 16 neuronů na 9. Tedy pevná topologie, vstupní vektor 7 Hu momentů, 1 skrytá vrstva 100 neuronů, výstupní vrstva 9 neuronů reprezentující binární hodnotu znaku. Trénovací množinu jsem zúžil jen na obrázky čísel 0-9 různých fontů a velikostí. Trénovací data jsem rozdělil na 2 části, na jedné se síť učila, druhou používala pro kontrolu chyby MSE (viz str.10), aby nedošlo k přeučení.

Výsledky rozpoznávání byly nedostačující, proto jsem se vrátil opět k návrhu struktury celé sítě.

Souhrn:

vstupní vrstva	–	7 neuronů – Hu momenty
skryté vrstvy	–	1 skrytá vrstva, 100 neuronů
výstupní vrstva	–	16 neuronů, každý nese hodnotu 1 nebo 0, výsledný vektor udává binární číslo reprezentující každý znak

Finální návrh

Ukázalo se, že návrh výstupního vektoru byl od počátku špatný. Přístup, kdy podle vstupního vektoru se snažíme určit výstupní vektor o více hodnotách, je spíše vhodný pro neuronové sítě typu asociativní paměti nebo typu ART. Perceptronová síť se s tímto problémem neumí vypořádat. Proto jsem opět poopravil výstupní vrstvu. Tentokrát obsahuje tolik neuronů, kolik je rozpoznávaných znaků, a každý znak je reprezentován právě jedním neuronem.

Vstupní vrstva obsahuje 7 neuronů, jediná skrytá 80 neuronů. Výstupní obsahuje 56 neuronů, kde každý neuron reprezentuje jedno písmeno. Nakonec jsem se rozhodl rozpoznávat alespoň velká a malá písmena anglické abecedy (je jich 56).

Souhrn:

vstupní vrstva	–	7 neuronů – Hu momenty
skryté vrstvy	–	1 skrytá vrstva, 80 neuronů
výstupní vrstva	–	56 neuronů, každý nese hodnotu 1 nebo 0, pořadí hodnoty 1 ve vektoru reprezentuje jeden z 56 znaků (velké a malé znaky anglické abecedy)

Průběhy učení a výsledky schopnosti rozpoznávání některých topologií jsou v popsány v kapitole Výsledky, viz str. 22.

Kapitola 6

Implementace

Ukázkový program pro rozpoznávání textu je implementován v jazyce C++. Od začátku byl navržen tak, aby byl pokud možno multiplatformní. Vyvíjen a úspěšně odzkoušen byl pod operačním systémem Windows XP Professional. Program využívá dvě externí knihovny:

- *OpenCV* – Open source knihovna, kterou začala vyvíjet společnost Intel. Knihovna se stará o veškerou práci okolo obrázků. Je rychlá a snadno použitelná.

  Url: http://www.intel.com/technology/computing/opencv/

- *FANN* – volně dostupná knihovna pro práci s neuronovými sítěmi. Implementuje vícevrstvou umělou neuronovou síť podporující plně nebo částečně propojené sítě. Je multiplatformní, umožňuje výpočty v pevné i pohyblivé čárce. Obsahuje prostředky pro snadnou manipulaci s trénovacími daty.

  Url: http://leenissen.dk/fann/index.php

Program je realizován jako konzolová aplikace, která se ovládá pomocí parametrů:

```
ocr.exe obrázek [-d -nr -p práh -t tolerance -s síť]
```

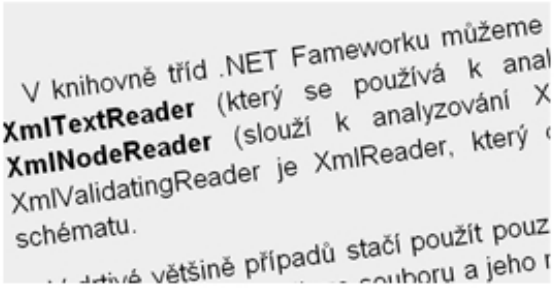
- `-d` – Zapnout zobrazování jednotlivých kroků zpracovávání obrázku
- `-nr` – Nepokoušet se opravovat pootočení textu
- `-p` – Nastavit práh pro prahování, rozmezí 0-255
- `-t` – Nastavit toleranci při segmentaci řádků a písmenek, rozsah 0-100
- `-s` – Zvolit jinou než defaultní síť pro rozpoznávání

Kapitola 7

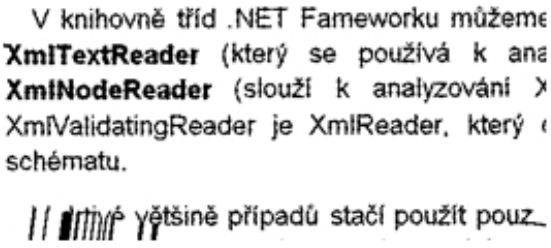
Výsledky

7.1 Korekce pootočení textu

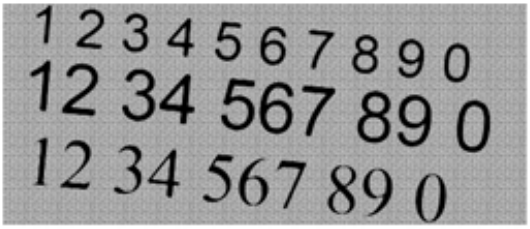
Provedl jsem několik testů s pootočenými obrázky v rozmezí -10 až 10 stupňů. Dostatečně kontrastní text, který zároveň pokrýval většinu plochy obrázku, byl ve většině případů správně vyrovnán.



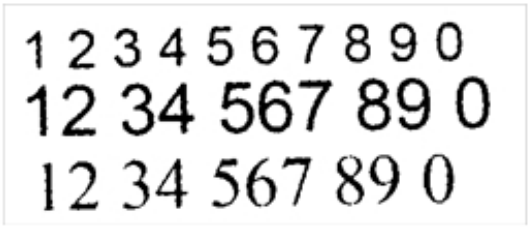
Rozpoznaný úhel: 7.5 stupňů



Úspěšné pootočení stránky



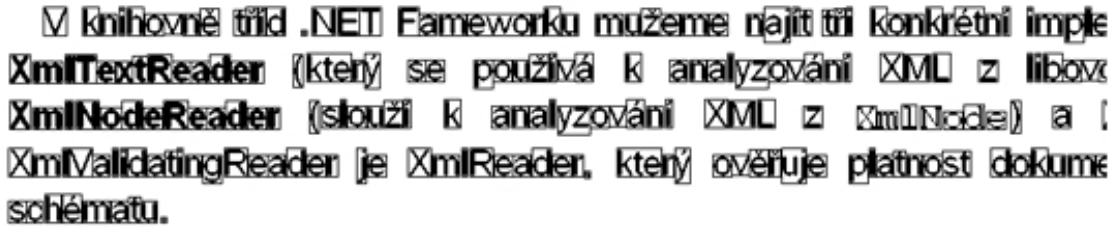
Rozpoznaný úhel: -6.5 stupňů



Úspěšné pootočení stránky

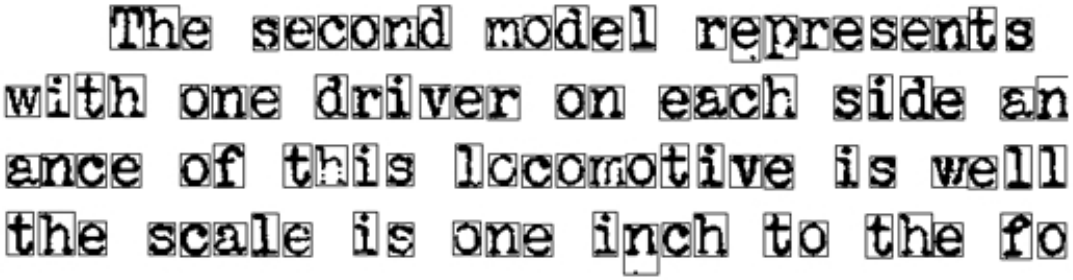
7.2 Lokalizace řádků a segmentace znaků

Segmentace řádků a znaků probíhá na základě počtu tmavých míst v oblasti, proto je hodně závislá na „čistotě“ obrázku. Další ovlivňující faktor je kontrast mezi textem a pozadím. Proto je potřeba zvolit vhodný práh pro prahování.



Obrázek 7.1: Dostatečně kontrastní text, žádný šum, přesto dochází ke splynutí několika písmenek v jedno (např. ve slově který)

Chyby nastávají v oblastech, kde se písmenka dotýkají. Řešením by mohla být jiná hodnota prahování, nebo na obrázek použít metodu zužování a tím písmenka rozpojit. Problematičtější jsou však písmenka, jejichž postavení neumožňuje najít mezeru použitým algoritmem. Například písmenka AV se příliš překrývají. Řešení – použít jiný algoritmus.

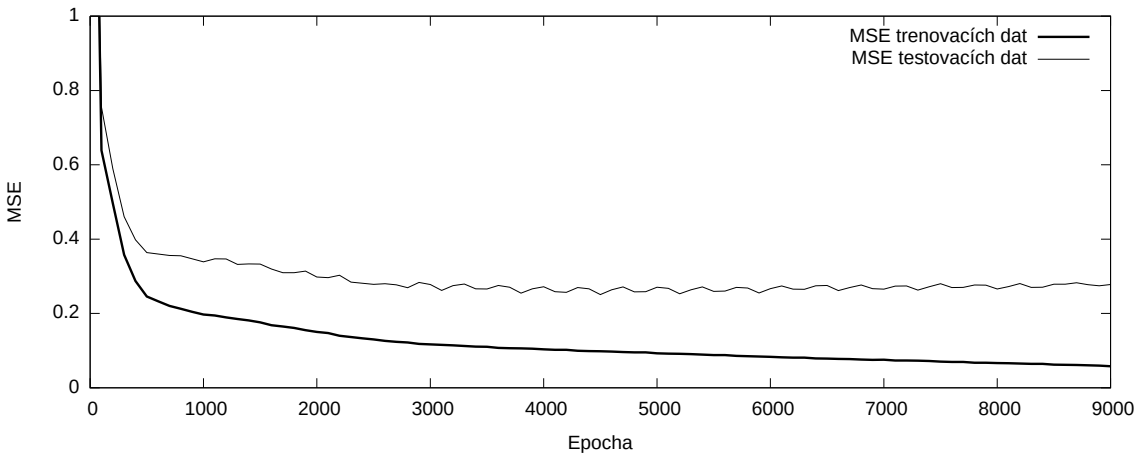


Obrázek 7.2: Výsledek segmentace naskenovaného textu, zde se neprojevilo žádné splynutí

7.3 Trénování

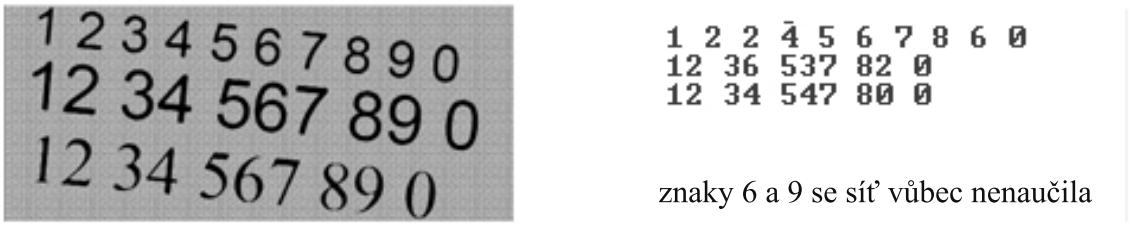
Zde uvádím jednotlivé průběhy učení pro různé topologie.

Návrh č. 3 - učení pouze znaků čísel



Obrázek 7.3: Průběh učení neuronové sítě

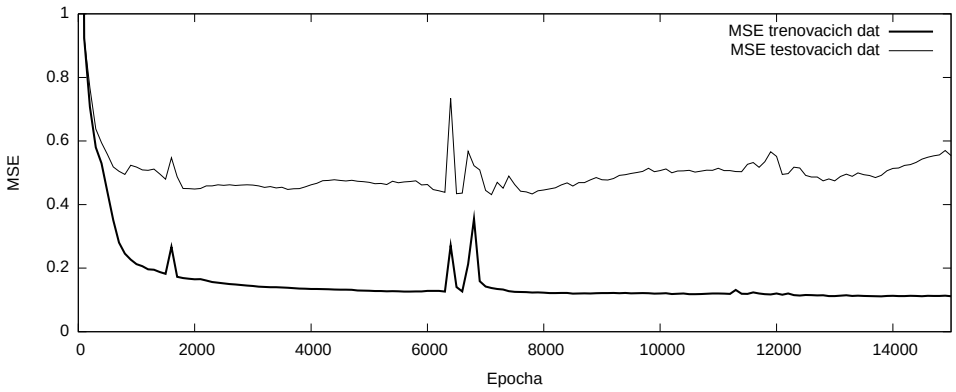
Učení bylo ukončeno po 8000 epochách, aby se zabránilo přeučení. Připomínám, že zde se síť učila jen 10 znaků (čísla 0-9) a topologie byla jiná, než u následujících příkladů. Pro zajímavost uvádím ukázku schopnosti rozpoznávání této sítě (obr. 7.4).



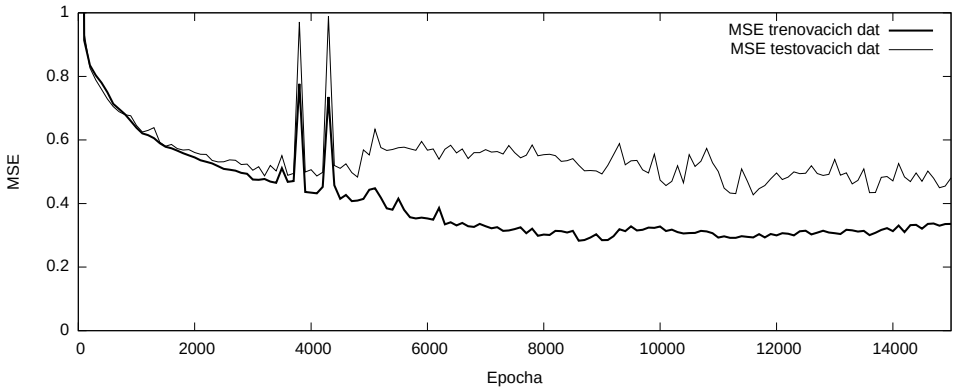
Obrázek 7.4: Výsledek rozpoznávání obrázku

Finální návrh - učení velkých a malých znaků anglické abecedy

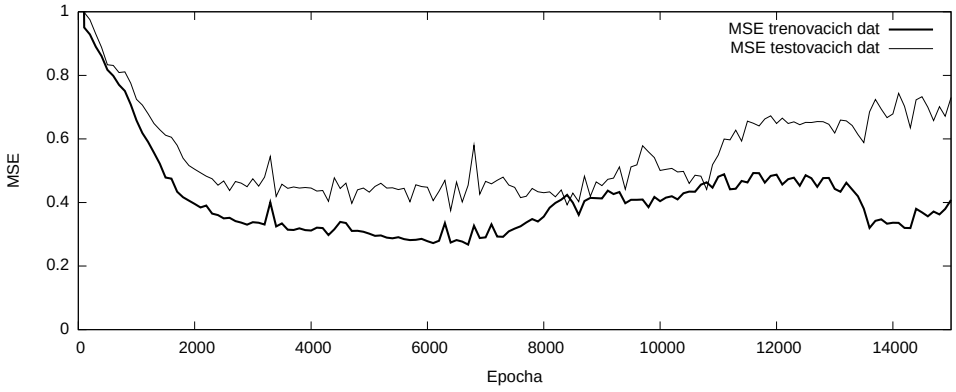
Výsledky učení finální sítě. Použita jsou stejná trénovací i testovací data, stejný učící algoritmus. Topologie sítí se liší pouze v počtu neuronů ve skryté vrstvě.



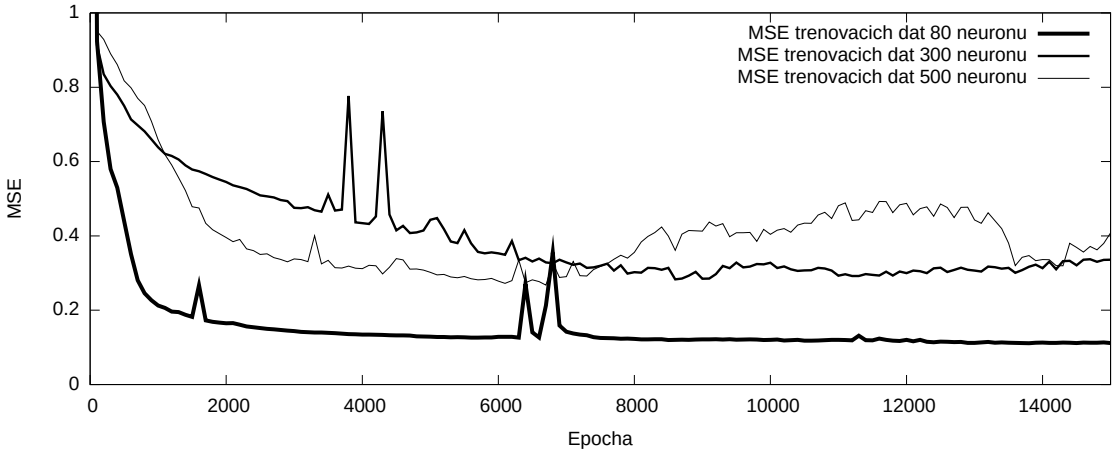
Obrázek 7.5: Průběh učení sítě s 80 neurony ve skryté vrstvě



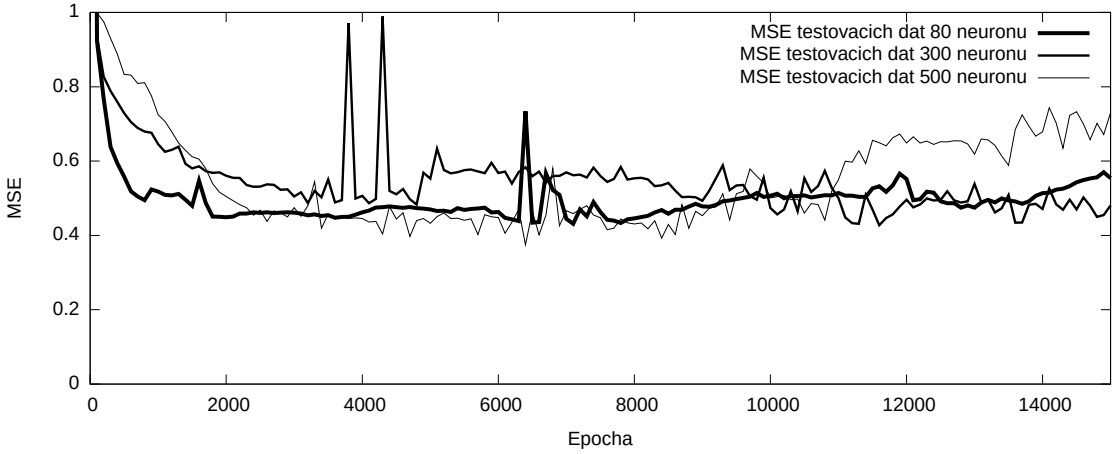
Obrázek 7.6: Průběh učení sítě s 300 neurony ve skryté vrstvě



Obrázek 7.7: Průběh učení sítě s 500 neurony ve skryté vrstvě



Obrázek 7.8: Průběh MSE na trénovacích datech vypadá nejlépe pro síť 80



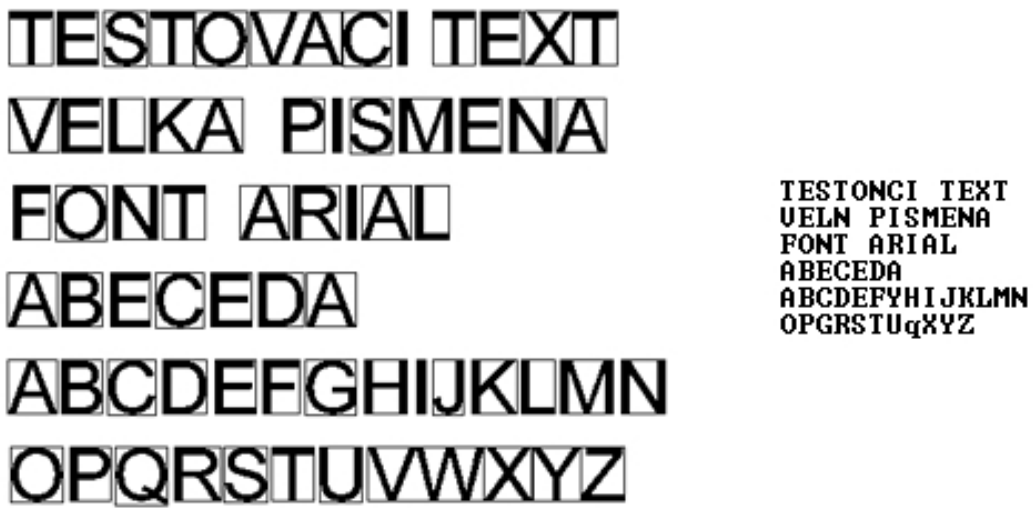
Obrázek 7.9: Průběh MSE na testovacích datech je u všech sítí podobný

Výsledky jsou celkem překvapivé. Přestože 80 neuronů ve skryté vrstvě se na první pohled může zdát málo, má tato síť v porovnání s ostatními nejlepší poměr průběhů MSE trénovacích a testovacích dat.

7.4 Rozpoznávání

Výsledná síť umí některá písmenka rozpoznat. Lépe rozpoznává velká písmenka (obr. 7.10) než malá (obr. 7.11).

Záměrně neuvádím žádné procentuální výsledky, protože jak je vidět z obrázků, program má problém rozpoznat i ideální obrázky. Jakýkoli mírně degradovaný znak už nerozpozná.



Obrázek 7.10: Výsledek rozpoznávání velkých písmen

Obrázek 7.11: Výsledek rozpoznávání malých písmen

Kapitola 8

Závěr

Cílem této práce bylo seznámit se s dnešními metodami rozpoznávání textu v obraze a na jejich základě vytvořit jednoduchý program pro rozpoznávání strojopisného textu.

Výsledná aplikace demonstruje použití několika zvolených metod:

- Korekce rotace textu hledáním prázdných vektorů.
- Segmentace textu za použití vertikálních a horizontálních histogramů.
- Extrahování rysů s využitím Hu momentů .
- Klasifikace znaků pomocí neuronových sítí.

Z výsledků je patrné, že zvolené metody zpracování obrazu je možné úspěšně použít na většinu strojopisného textu. Naopak kombinace Hu momentů a neuronových sítí při klasifikaci není příliš ideální. Toto spojení je použitelné pouze pro menší počet rozpoznávaných znaků (například pro číslice). Úspěšnost rozpoznání také hodně závisí na kvalitě zdrojového obrázku. Příčinou nejspíše budou zvolené Hu momenty, které nedokážou dostatečně rozlišit větší počet znaků a jsou velmi citlivé na sebemenší změnu vzhledu znaku. Lepších výsledků by mohlo být dosaženo použitím jiné metody extrakce příznaků, například použitím tzv. skeletonizace. Tato metoda vytvoří z každého znaku kostru (spojitá oblast tmavých bodů o šířce 1 pixel), ze které se dají určit směry sousedních bodů, a pomocí počtu jednotlivých směrů charakterizovat znaky. Metoda by mohla být méně náchylná na degradaci znaků.

V budoucích fázích projektu je možné se zaměřit na další metody extrakce a klasifikace, například implemenetovat výše zmíněnou skeletonizaci nebo vyzkoušet další topologie neuronových sítí.

Literatura

[1] Optical character recognition - Wikipedia, the free encyclopedia. [online], This page was last modified 07:38, 25 April 2007, [cit 1. 5. 2007].
URL http://en.wikipedia.org/wiki/Optical_character_recognition

[2] AdaBoost - Wikipedia, the free encyclopedia. [online], This page was last modified 07:57, 19 April 2007, [cit 1. 5. 2007].
URL http://en.wikipedia.org/wiki/AdaBoost

[3] Support vector machine - Wikipedia, the free encyclopedia. [online], This page was last modified 13:48, 1 May 2007, [cit 5. 5. 2007].
URL http://en.wikipedia.org/wiki/Support_vector_machine

[4] Hough transform - Wikipedia, the free encyclopedia. [online], This page was last modified 18:36, 2 May 2007, [cit 5. 5. 2007].
URL http://en.wikipedia.org/wiki/Hough_transform

[5] Euclidean distance - Wikipedia, the free encyclopedia. [online], This page was last modified 20:58, 18 March 2007 [cit 5. 5. 2007].
URL http://en.wikipedia.org/wiki/Euclidean_distance

[6] ADMASSU, D.: *Unicode Optical Character Recognition*. [online], Updated: 23 Aug 2006, [cit 5. 5. 2007].
URL http://www.codeproject.com/cs/algorithms/UnicodeOCR.asp

[7] BERNACKI, M.; WLODARCZYK, P.: *Principles of training multi-layer neural network using backpropagation algorithm*. [online], 2005, [cit 5. 5. 2007].
URL http://galaxy.agh.edu.pl/~vlsi/AI/backp_t_en/backprop.html

[8] CLAUS, V.; SCHWILL, A.: *Encyclopaedia of Information Technology*. Chichester : Ellis Horwood Books In Information Technology, 1992, ISBN 0-13-275728-1.

[9] EIKVIL, L.: *OCR - Optical Character Recognition*. [online], 1993.
URL http://citeseer.ist.psu.edu/eikvil93ocr.html

[10] HU, M.-K.: *Visual pattern recognition by moment invariants*. Information Theory, IEEE Transactions on, ročník 8, 1962: s. 179 – 187, ISSN 0018-9448.

[11] KŘÍŽ, J.: *Velký frekvenční slovník počítačů*. Ostrava: Montanex, 1995, ISBN 80-85780-47-X.

[12] SHUTLER, J.: *Centralised moments*. [online], 2002-08-15, [cit 5. 5. 2007].
URL http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/SHUTLER3/node5.html