Univerzita Komenského, Bratislava
Fakulta Matematiky, Fyziky a Informatiky

OCR pomocou Holland-style adaptívnych klasifikátorov

Bakalárska práca

2013

Miloš Olleríny

Univerzita Komenského, Bratislava
Fakulta Matematiky, Fyziky a Informatiky

OCR pomocou Holland-style adaptívnych klasifikátorov

Bakalárska práca

Študijný program:	Informatika
Študijný odbor:	2508 Informatika
Školiace pracovisko:	Katedra Informatiky
Vedúci záverečnej práce:	RNDr. Tomáš Kulich, PhD.

Bratislava, 2013
Miloš Olleríny

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Miloš Olleríny
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský

Názov: OCR pomocou Holland-style adaptívnych klasifikátorov

Cieľ: Nadviazať na prácu Frey, Slate: "Letter recognition using Holland-style adaptive classifiers". Preskúmať kvalitu OCR v závislosti od počtu features a spôsobu trénovania klasifikátorov.

Vedúci: RNDr. Tomáš Kulich, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: doc. RNDr. Daniel Olejár, PhD.

Dátum zadania: 18.10.2012

Dátum schválenia: 24.10.2012 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Pod'akovanie

Ďakujem RNDr. Tomáš Kulich, PhD. za cenné rady, pripomienky a odborné vedenie pri vypracovávaní bakalárskej práce.

Abstrakt

Technológia, ktorá sa zaoberá strojovým rozpoznávaním znakov, sa nazýva OCR. Táto technológia má vo svete pomerne veľké využitie, no napriek existencii viacerých metód riešiacich daný problém, ani jedna z nich nie je dostatočne univerzálna. Existuje výskum, na ktorom sa podieľali P. Frey a D. Slate [FS91], ktorí sa snažili využiť klasifikátorský systém na strojové rozpoznávanie znakov. Podstatou tohoto systému bolo vyjadrovanie tzv. features jednotlivých písmen pomocou využitia rôznych funkcií. Týmto spôsobom vyjadrili každé písmenko ako vektor čísiel a následne ich použili v strojovom učení. V našej práci sme sa rozhodli na tento výskum nadviazať a zistiť, či nedokážeme modifikáciami jednotlivých postupov zvýšiť úroveň rozpoznávania písmen. Zamerali sme sa hlavne na zaznamenávanie viac vlastností písmen pomocou väčšieho množstva parametrizovaných funkcií a obrázkových modulácií. Ako vzorku písmen sme si pripravili rozsiahle množstvo znakov v rôznych fontoch, ktoré sme ľubovolne zmodifikovali. Hlavným cieľom práce je zistiť, či je takýto prístup vhodný, všeobecne použiteľný a vedie k uspokojivým výsledkom.

Kľúčové slová: OCR, rozpoznávanie písmena, strojové učenie, klasifikátory

Abstract

The technology, which studies letter recognition using machine, is called OCR. This technology is spread all over the world and is used quite often. In spite of existing many methods which try to solve this problem, none of them is usefull enough in general cases. There is a research formed by P. Frey and D. Slate [FS91], who tried to create a classifier system used for machine learning in order to recognize letters. The main substance of this research was to express features of each letter using different functions. This way they were able to formulate each letter into a vector containing numbers, which led to use those vectors in a machine learning process. In our thesis, we decided to examine this research and try to determine, whether it is possible to increase the quality of letter recognition by modificating individual processes. Our focus was to detect many features of letters using parametric functions and image modulations. As a sample, we used many different fonts to generate a big amount of letters, which were randomly modified. The main aim of this research is to find out, whether this approach is accurate, widely usable and leads us to satisfying results.

Keywords: OCR, letter recognition, machine learning, classifiers

Obsah

Zoznam obrázkov

Zoznam tabuliek

Úvod

Technológia, ktorá sa zaoberá rozpoznávaním znakov v texte, je známa ako OCR (Optical character recognition). Ide o tzv. konverziu naskenovaného textu do digitalizovanej podoby. Metód, akými sa tieto konverzie vykonávajú, je viacej, medzi najznámejšie a najviac používané patria Matrix matching (maticová zhoda) a Feature extraction (zhoda na vlastnostiach) [MSY92]. Zatiaľ čo v prvej metóde sa znaky porovnávajú "prikladaním"písmen z databázy na testovací znak a písmeno jemu najviac podobné je výsledok, druhá metóda využíva počítanie tzv. features (vlastnosti) písmen a na základe nich porovnáva vzorky. Z popisu týchto metód sa dá spozorovať, že Feature extraction je univerzálnejšia, kedže rovnaké písmená v rôznych fontoch majú aj rôzny tvar, no aj napriek tomu je väčšina ich features rovnaká (napr. krivky, symetria a pod.).

Hlavnou témou tejto práce je nadviazať na prácu P. Frey a D. Slate [FS91], ktorá využívala tzv. Holland-style adaptívne klasifikátory na zatrieďovanie písmien podľa ich rôznych features. Využívali sa pri tom rozmanité funkcie, pričom každá vyjadrovala určitú feature písmena. Podstatou tejto práce je vytvoriť klasifikátorský systém fungujúci na podobnom princípe, a na ňom skúmať a analyzovať vlastnosti jednotlivých funkcií a zároveň ich aj porovnať s tými, ktoré boli použité v práci [FS91]. Naším cieľom je taktiež zistiť, či kvalita rozpoznávania písmen závisí od počtu skúmaných features jednotlivých písmen (teda od počtu funkcií), od mechanizmu strojového učenia alebo aj od veľkej rozmanitosti vzoriek, ktoré si vytvoríme a použijeme, či už na testovanie alebo strojové učenie.

V prvej časti tejto práce popíšem výskum [FS91], na ktorom P. Frey a D. Slate pracovali. Zhrniem ideu používania adaptívnych klasifikátorov a rôzne formy strojového učenia kde sa dané klasifikátory používali. Druhá kapitola už bude venovaná nášmu výskumu, v ktorom sa budem zaoberať tvorbou modifikovaných písmen, implementáciou systému funkcií a klasifikátorov a taktiež popíšem techniky, ktoré sme používali na efektívne spracovanie veľkého množstva údajov databázach. Následne sa budem venovať aj strojovému učeniu a posledná kapitola bude venovaná štatistikám, kde popíšem ich tvorbu a typy a zhrniem výsledky testovaní a analýz. Dôležitou súčasťou výskumu je porovnávanie features použitých v práci [FS91] s features, ktoré sme implementovali my. Za programovací jazyk, v ktorom

sme sa rozhodli daný výskum realizovať, sme si zvolili Python [Fou], využívajúc najmä jeho knižnicu Numpy [Num].

Kapitola 1

Holland-Style adaptívne klasifikátory

V prvej kapitole by som sa chcel venovať článku Letter Recognition Using Holland-Style Adaptive Classifiers [FS91] popisujúci výskum ohľadom rozpoznávania znakov metódou klasifikovania, na ktorý sme v tejto práci nadviazali a je pre nás smerodajný. Tento výskum, na ktorom pracovali P. Frey a D. Slate, pozostával z nasledujúcich častí:

1.1 Príprava znakov a funkcií

Na to, aby sa výskum mohol realizovať, museli sa najprv pripraviť potrebné atribúty v podobe testovacích sád znakov a funkcií popisujúce ich rôzne features.

1.1.1 Znaky

Ako testovacia a učiaca vzorka bola zvolená sada 20 000 jedinečných znakov, ktoré boli zoskupené do 26 skupín (pre každé písmenko jedna skupina). Tieto písmená boli vyberané z 20 rôznych fontov a následne mierne skreslované, aby sada neobsahovala žiadne dva rovnaké znaky. Všetky písmená boli vygenerované čiernou farbou na bielom pozadí.

1.1.2 Funkcie

Účelom funkcií bolo vyjadriť nejakú jednu konkrétnu feature písmenka pomocou vektora čísiel - atribútov. Jednotlivé atribúty sa počítali na základe pixelov nasnímaného znaku (či je pixel biely alebo čierny). Konkrétnym vlastnostiam sa budem venovať neskôr pri ich porovnávaní s našimi funkciami.

1.2 Učiaca fáza

Po implementovaní funkcií sa každé jedno písmenko preiterovalo a vznikol tak vektor čísiel vyjadrujúci vlastnosti daného písmenka. Následne sa vybralo 16000 písmien (už v podobe vektorov), ktoré slúžili ako učiaca sada a teda boli použité v učiacej fáze. Zvyšných 4 000 vektorov bolo použitých ako testovacia sada.
Učiaca fáza prebiehala nasledovne:

1. Porovnanie aktuálneho vektora z testovacej sady s každým vektorom z učiacej sady.
2. Výber takej množiny klasifikátorov M, ktoré sa najviac podobajú danému testovaciemu vektoru v závislosti od použitého kódovania atribútov.
3. Pre všetky klasifikátory z M vypočítať na základe rôznych techník silu (účinnosť), ako veľmi sa testovaciemu vektoru podobajú a na základe nej priradiť danému vektoru klasifikátor s najvyššou ponukou.
4. Na základe techniky použitej pri vypočítaní účinnosti nasleduje modifikovanie jednotlivých klasifikátorov, napr. ich vyradenie alebo modifikovanie niektorých atribútov
5. Tento proces sa opakuje pre všetky vzorky z testovacej sady.

Atribúty sa vo vektoroch písmien v klasifikátoroch kódovali nasledovnými tromi spôsobmi:

- Binárne kódovanie - každý atribút je zapísaný ako 4-bitové binárne číslo, porovnávanie klasifikátorov je úspešné vtedy, keď sa atribúty na všetkých pozíciách rovnajú
- Grayovo kódovanie - podobné, ako binárne, no porovnávanie nám zaručuje zhodu aj na vektoroch, ktoré sa líšia iba v jednom atribúte
- Celo-číselné kódovanie - atribúty sú preškálované na celé čísla od 0 do 15, porovnávanie klasifikátorov sa tu rozdeľuje ešte na ďalšie tri podprípady.

1.3 Generovanie nových pravidiel

Definícia 1 Strojové učenie je disciplína umelej inteligencie smerujúca k technologickému rozvoju napodobňujúci ľudské myslenie. Strojové učenie dovoľuje počítačom spracovávať nové stavy pomocou analýz, seba-trénovania, pozorovania a skúseností [Rou].

V opisovanom výskume boli použité 3 rôzne techniky generovania nových pravidiel založených na báze strojového učenia, ktoré sa používali v učiacej fáze. Ich podstatou bolo oslabovať alebo zosilňovať jednotlivé klasifikátory, poprípade ich úplne vyradiť a nahradiť inými. Sila daného klasifikátora sa hodnotila podľa 2 techník, najprv však stručne popíšem algoritmy na generovanie pravidiel. Všetky z veľkej časti využívali náhodnosť a tzv. divoké karty, ktoré slúžia ako náhrada za ľubovolnú hodnotu atribútu.

1.3.1 Techniky generovania

Náhodné generovanie

Každému klasifikátoru bola priradená divoká karta alebo sa náhodným atribútom v jednotlivých klasifikátoroch priradili náhodné hodnoty. Keď takto upravené pravidlo nespĺňalo preddefinované levely (porovnávala sa sila jednotlivých klasifikátorov vypočítaná v závislosti od použitej techniky), bolo okamžite vyradené a nahradené novo vytvoreným pravidlom.

Hybridná procedúra

Klasifikátori sa upravili tak ako v predošlej metóde, keď však nesplnili definované kritériá, náhodne sa vyvolala jedna z 3 rôznych metód vytvorenia nového pravidla.

Vzorovo-založené generovanie

Táto metóda vytvárala pri každej nesprávnej identifikácii klasifikátora 2 nové pravidlá, a to tak, že s pravdepodobnosťou 50% bola každému atribútu pôvodného pravidla priradená divoká karta.

1.3.2 Techniky prerozdeľovania sil

Winner-take-all

Výslednú silu si pripíše len pravidlo, ktoré bolo najúspešnejšie.

Reward-sharing

Výsledná sila sa prerozdelí medzi viacero klasifikátorov.

1.3.3 Techniky hodnotenia klasifikátorov

priamo-úmerne k sile

Výsledná sila klasifikátora sa počíta len na základe jeho účinnosti.

priamo-úmerne k súčinu sila · presnosť

Pri tomto výpočte sa berie do úvahy aj presnosť, t.j. počet divokých kariet medzi atribútmi klasifikátora.

1.4 Výsledky

Frey a Slate na základe mnohých štatistík a pozorovaní dosiahli rôzne výsledky. Najlepší výsledok správne rozpoznaných písmen, aký sa im podarilo dosiahnuť, bolo 82,7%. Tento výsledok vznikol kombináciou nasledovných procedúr.

Z troch rôznych kódovaní atribútov v klasifikátoroch bolo najviac prosperujúce celočíselné s úspešnosťou približne 80%, zatiaľ čo zvyšné dve kódovania dosahovali úspešnosť približne 60%. Takisto dokázali, že najúspešnejšou metódou pri generovaní nových pravidiel je Vzorovo-založené generovanie. Táto procedúra dokázala v rozumnom čase vygenerovať efektívnejšie klasifikátory, ako Hybridná alebo Náhodná metóda. Čo sa týka techniky prerozdeľovania síl jednotlivým klasifikátorom, jednoznačne lepšou procedúrou je Reward-sharing.

V tomto výskume sa P. Frey a D. Slate zameriavali na rozpoznávanie písmen využívaním rôznych variant strojového učenia a použitím klasifikátorov na kategorizovanie jednotlivých prípadov. Ich výsledky síce nie sú neúspešné, ale ako sami uviedli, existuje ešte mnoho iných prístupov, ktoré sa dajú skúmať a môžu byť prosperujúce.

1.5 Technická správa

Existuje výskum [Fog92], na ktorom pracoval Terence C. Fogarty, ktorý tiež nadväzuje na prácu [FS91], využívajúc rovnakú sadu znakov a rovnaké funkcie na meranie features. Na to, aby dosiahol lepšie výsledky, sa rozhodol zameniť komplikované strojové učenie využívajúce Holland-style adaptívne klasifikátori za jednoduchší algoritmus klasifikovania, nazvaný First Nearest Neighbor (Prvý najbližší sused). Tento algoritmus spočíva vo vypočítaní Euklidovskej vzdialenosti testovacieho znaku s každým klasifikátorom z učiacej sady. Výsledné

písmeno určuje klasifikátor, s ktorým ma najmenšiu vzdialenosť (čiže je jeho najbližší susedd).

Podľa štatistík [Fog92] sa pomocou tejto metódy dosahovala až 95,4% presnosť, čo je v porovnaní s výsledkami [FS91], ktoré dosahovali maximálne 82,7% pomerne veľký rozdiel. Negatívum takéhoto prístupu je v porovnaní s výskum Fray-Slate väčšia časová aj priestorová zložitosť.

Kapitola 2

Modifikácia výskumu

V tejto kapitole budem popisovať výskum, ktorým sa v tejto práci zaoberáme. Využívame klasifikátory fungujúce na podobnom princípe, ako vo výskume [FS91] popísanom v predošlej kapitole. Našou snahou bolo nájsť také modifikácie jednotlivých procedúr vykonávaných počas výskumu, s ktorými by sme dosiahli lepšie výsledky a tým pádom zvýšili kvalitu rozpoznávania písmen. Oblasti, o ktorých si myslíme, že do značnej miery ovplyvňujú kvalitu rozpoznávania a je vhodné ich kompletne modifikovať, sú:

- hlbšia modifikácia znakovej sady
- použitie väčšieho množstva funkcií na vyjadrenie rôznych features
- použitie jednoduchšej formy strojového učenia

2.1 Znaky

Naša sada písmen je vyskladaná zo 160 fontov, pričom z každého fontu vyberáme a modifikujeme 26 písmen abecedy. Číslam a diakritike sa v našom výskume nevenujeme. Všetky písmená sú pôvodne vygenerované bielou farbou na čierny podklad. Ešte predtým, ako sme si z týchto písmen vygenerovali klasifikátori, museli sme ich modifikovať.

2.1.1 Čierno-bielo-sivé písmená

Vo výskume Frey-a a Slate-a sa znaky do učiacej aj testovacej sady vyrábali a modifikovali tak, aby ich jednotlivé pixely boli buď iba čierne alebo biele. Myslíme si, že takýto prístup nieje vhodný, a je lepšie ponechať, resp. vyrobiť na písmene aj rôzne odtiene sivej farby.

2.1.2 Distractions

Na každé písmeno v učiacej sade sme aplikovali tzv. distractions. Sú to funkcie, ktorých účelom je mierne zmodifikovať jednotlivé písmená. Dôvodom nebolo len zabezpečiť, aby žiadne dve písmená neboli totožné alebo veľmi sa na seba podobajúce. Chceli sme hlavne zabezpečiť, aby aj kamerou nasnímané písmená, na ktorých budeme náš výskum testovať, sa aj pri nepresnom zachytení podobali na našu znakovú sadu. Modifikácie, ktoré sme použili, sú:

- blur - rozmazanie písmena
- rand_add - preškálovanie pixelov v písmene podľa Gaussovej krivky
- deform_metric - náhodné natiahnutie/skrátenie niekoľkých častí písmena.

Všetky tieto modifikácie využívali generátor pre náhodné hodnoty, čo nám zaručuje unikátne písmená v celej sade. Na obrázku 2.1 si môžete pozrieť účinnosť modifikovania šachovnice pomocou funkcie deform_metric pre parametre rastúce od 0 po 9.

Obr. 2.1: Modifikácia šachovnice pomocou deform_metric pre rôzne parametre

Distractions nám taktiež zaručili, že písmená nebudú len čiernej a bielej farby, ale vyskytne sa tam aj mnoho odtieňov sivej farby. Obrázok 2.2 zobrazuje 10 náhodne vybratých písmeniek pred a po zmodifikovaní pomocou všetkých 3 distractions v poradí deform_metric, blur, rand_add.

Obr. 2.2: Modifikácia písmen pomocou distractions

2.2 Funkcie

Ďalšou, a hlavnou sekciou, ktorú sme oproti výskumu Frey-a a Slate-a zmenili, sú funkcie. V ich výskume bolo na vyjadrenie rôznych features použitých 16 funkcií. V našom výskume sme chceli zistiť, či by zvýšenie počtu funkcií prinieslo odlišné výsledky a ak áno, snažiť sa nájsť takú podmnožinu funkcií, ktoré nám dávajú najlepší prínos.

2.2.1 Systém funkcií

Na to, aby sme vygenerovali veľké množstvo funkcií, každá reprezentujúca inú vlastnosť, vytvorili sme si nasledovný systém:
Na celé vstupné písmeno postupne aplikujeme nasledovné 3 modulácie, ktorých využitie vysvetlím pri popise funkcií:

- identita - modulácia vráti identitu obrázka
- diff(axis = X) - modulácia vráti upravené písmeno tak, že sa odčítajú hodnoty 2 pixelov, ktoré sú pri sebe horizontálne
- diff(axis = Y) - modulácia vráti upravené písmeno tak, že sa odčítajú hodnoty 2 pixelov, ktoré sú pri sebe vertikálne

Následne každý z týchto modulačných obrázkov iterujeme a počítame jednotlivé Features, pričom každá je charakterizovaná kombináciou 7 parametrov *mod*, pow_X, k_X, ph_X, pow_Y, k_Y, ph_Y. Každú Feature teda vieme popísať vzťahom:

$$\text{Feature} = \frac{\sum_{i}\sum_{j} \text{mod(obr)[i,j]} \cdot \text{f}_{\text{pow}_X,\text{k}_X,\text{ph}_X}(\text{i}) \cdot \text{f}_{\text{pow}_X,\text{k}_Y,\text{ph}_Y}(\text{j})}{\sum_{i}\sum_{j} \text{mod(obr)[i,j]}},$$

pričom čísla i, j vyjadrujú aktuálnu pozíciu v danom obrázku (písmene). Funkcia f je definovaná nasledovne:

$$\text{f}_{\text{power,k,phase}}(\text{i}) = \text{i}^{\text{power}} \cdot \cos(\text{k} \cdot \pi \cdot \text{i} + \text{phase})$$

Power, k, *phase* sú parametre, i je poloha aktuálneho pixela v danom písmene (i nadobúda hodnoty z intervalu $\langle -0.5, 0.5 \rangle$ a vystihuje relatívnu pozíciu aktuálneho bodu k celkovej výške alebo šírke písmena). Parametre *power* a k sú z množiny {0, 1, 2}, *phase* nadobúda 4 rovnomerne vybrané hodnoty z intervalu $\langle 0,\pi)$. Výsledky každej iterácie sa postupne pripisujú do vektora daného písmenka, čím ho klasifikujeme.

Práve vďaka takýmto hodnotám parametrov sme schopní vypočítať okrem iného aj features použité vo výskume, na ktorý sme nadviazali, a zároveň nám tento systém funkcií poskytuje aj mnoho ďalších informácií o písmene.

V tejto fáze už vieme zistiť, že počet vygenerovaných funkcií bude 3888, čiže každé jedno písmeno vieme popísať vektorom dĺžky 3888 obsahujúci čísla reprezentujúce jednotlivé features. Takto popísané písmená sme si uložili do hlavnej databázy, vďaka ktorej sme neskôr mohli realizovať strojové učenie a hľadať tie najprínosnejšie funkcie.

2.2.2 Vlastnosti písmen vyjadrené funkciami

Následne pomocou rôznych hodnôt parametrov pre funkciu a moduláciu vieme popísať všetkých 16 features použitých v [FS91]:

1. určenie horizontálneho ohraničenia písmenka
   - v práci [FS91] sa vychádzalo z predpokladu, že vstupné písmená neboli orezané na najmenšie možné ohraničenie. V našom výskume sme si písmená už pri vytváraní orezávali, takže táto feature pre nás nie je potrebná.
2. určenie vertikálneho ohraničenia písmenka
   - platí to, čo aj v predošlom bode, písmená sme mali orezané už pri ich generovaní.
3. určenie šírky znaku
   - šírka znaku je veľmi obmedzujúca vlastnosť a poskytuje nám informáciu len o malej množine písmen, preto si myslíme, že táto vlastnosť nie je vhodná na meranie.
4. určenie výšky znaku
   - o výške znaku platí to isté, čo o šírke, preto si myslíme, že meranie tejto feature nám nespokytne adekvátnu informáciu.
5. počet čiernych pixelov, ktoré tvoria daný znak
   - aj o táto vlastnosť je príliš obmedzujúca, nakoľko si myslíme, že pri rovnakých písmenách rôznej veľkosti nám táto vlastnosť klasifikovanie pravdepodobne viac pokazí, ako by mala pomôcť. Preto sme túto vlastnosť nezaradili ako potrebnú pre náš výskum.
6. priemerná horizontálna pozícia čiernych pixelov relatívne k stredu (napr. písmeno L má hodnotu tohto atribútu zápornú)

Modulácia = Identita			
	power	k	phase
X	1	0	0
Y	_	0	0

Tabuľka 2.1: Feature nr. 1

7. priemerná vertikálna pozícia čiernych pixelov relatívne k stredu

Modulácia = Identita			
	power	k	phase
X	_	0	0
Y	1	0	0

Tabuľka 2.2: Feature nr. 2

8. pomocou predošlého ohodnotenia vyjadriť priemernú početnosť čiernych pixelov po uhlopriečkach (napr. I má hodnotu tohto atribútu menšiu, ako Y)

Modulácia = Identita			
	power	k	phase
X	1	0	0
Y	1	0	0

Tabuľka 2.3: Feature nr. 3

9. ohodnotenie, ako veľmi je znak roztiahnutý v horizontálnom smere (napr. písmeno M má väčšie ohodnotenie ako I)

Modulácia = Identita			
	power	k	phase
X	2	1	0
Y	_	_	_

Tabuľka 2.4: Feature nr. 4

10. ohodnotenie, ako veľmi je znak roztiahnutý vo vertikálnom smere

Modulácia = Identita			
	power	k	phase
X	_	_	_
Y	2	1	0

Tabuľka 2.5: Feature nr. 5

11. korelácia medzi horizontálnou odchýlkou a vertikálnou pozíciou pixelov

Modulácia = Identita			
	power	k	phase
X	2	1	_
Y	1	1	_

Tabuľka 2.6: Feature nr. 6

12. korelácia medzi vertikálnou odchýlkou a horizontálnou pozíciou pixelov

Modulácia = Identita			
	power	k	phase
X	1	1	_
Y	2	1	_

Tabuľka 2.7: Feature nr. 7

13. priemerný počet okrajov - biely pixel vedľa čierneho, pri iterovaní písmenom zľava doprava (napr. M má väčšie ohodnotenie ako L)

    - v nasledujúcich funkciách už nepoužívame identitu ako moduláciu obrázka, ale diff(axis), nakoľko táto modulácia nám poskytuje prehľad o jednotlivých hranách písmeniek.

Modulácia = diff(axis = Y)			
	power	k	phase
X	2	1	_
Y	_	_	_

Tabuľka 2.8: Feature nr. 8

14. súčet vertikálnych okrajov nameraných v predošlom bode

Modulácia = diff(axis = Y)			
	power	k	phase
X	1	1	_
Y	_	_	_

Tabuľka 2.9: Feature nr. 9

15. priemerný počet okrajov - biely pixel vedľa čierneho, pri iterovaní písmenom zdola nahor (napr. K má tento atribút väčší ako I)

Modulácia = diff(axis = X)			
	power	k	phase
X	_	_	_
Y	2	1	_

Tabuľka 2.10: Feature nr. 10

16. súčet horizontálnych okrajov nameraných v predošlom bode

Modulácia = diff(axis = X)			
	power	k	phase
X	_	_	_
Y	1	1	_

Tabuľka 2.11: Feature nr. 11

2.3 Škálovanie features

Škálovaním features sme nazvali proces, vďaka ktorému vieme všetky features zapisovať do databázy v jednotnej efektívnej mierke. Tento postup bol zvolený kvôli efektívnosti a miernej optimalizácii databázy, v ktorej máme všetky klasifikátory uložené.

Jednotlivé features sú vypočítané a zobrazované ako reálne čísla, zaznamenávané s presnosťou aj viac ako 10^{-6}, a to prináša so sebou aj určité nevýhody, akými sú napr. pamäťová náročnosť (reálne čísla zaberajú v pamäti počítača viac priestoru), náročnejšie výpočty s desatinnými časťami, alebo nerovnomerné rozloženie. Totiž ľahko môže nastať situácia, že nejaké konkrétne features nadobúdajú vo všetkých klasifikátoroch iba hodnoty v malom intervale, zatiaľ čo iné môžu byť mnohonásobne väčšie, prípadne jednotlivé intervaly nemusia byť rovnomerne zastúpené, a tým pádom máme na dané features pomerne chaotický pohľad.

Na to, aby sme zabránili takému nerovnomernému rozloženiu si budeme jednotlivé features už zároveň počas generovania škálovať do podoby celých čísiel. Ako finálny interval sme si zvolili $\langle 0{,}255 \rangle$. Algoritmus je skonštruovaný tak, že na základe 512 vzoriek písmen si vypočíta veľkosť preškálovania pre každú feature osobitne, čím sa dosiahne, že jednotlivé atribúty, ktoré sú vo všetkých klasifikátoroch na rovnakých pozíciách, budú na danom intervale rovnomerne rozložené.

2.4 Učiaca fáza

Momentálne už máme pripravené všetky atribúty potrebné na strojové učenie a vytváranie databáz: vygenerované znaky, ktoré sme zmodifikovali pomocou distractions, systém funkcií, vďaka ktorým vieme jednotlivé písmená vyjadriť ako vektor čísel a vhodný algoritmus na škálovanie jednotlivých features, vďaka čomu ich vieme efektívne ukladať do databáz.

2.4.1 Príprava databázy

Naša finálna databáza, od ktorej sa neskôr bude odvíjať strojové učenie má nasledujúce parametre: obsahuje zoznam 3888 funkcií, zoznam veľkostí preškálovania pre každú feature a 4000 klasifikátorov (každý obsahujúci 3888 features) vyjadrujúcich 26 písmen abecedy. Naším cieľom je nájsť a vyradiť také funkcie, ktoré nám poskytujú slabú informáciu o písmene a následne vytvoriť databázu už len so zvyšnými "prospešnými"funkciami a novovypočítanými klasifikátormi. Počet, na ktorý sme sa snažili funkcie zredukovať bol po viacerých pokusoch a testovaní stanovený na 1024.

Naše strojové učenie v porovnaní s prístupom použitým v [FS91] funguje na jednoduchšom princípe. Zatiaľ čo P. Frey a D. Slate využívali mnoho techník a algoritmov na generovanie nových klasifikátorov, úpravu jednotlivých atribútov, prípadne rozoberanie viacerých podproblémov, my sme si vystačili s niekoľkými algoritmami, ktoré sa zameriavali na ohodnocovanie úspešností jednotlivých klasifikátorov a hľadanie najprínosnejších funkcií.

2.4.2 Strojové učenie

Ako testovacie vzorky sme si zvolili sadu 256 znakov, ktoré sme porovnávali s klasifikátormi uloženými v databáze. Z každej vzorky z testovacej sady sme po vypočítaní features hľadali v databáze také klasifikátori, ktoré sa na daný vektor najviac podobali. Využívali sme pri tom algoritmus na výpočet Euklidovskej vzdialenosti. Podobný algoritmus bol použitý aj v spomínanej Technickej správe [Fog92]. Kedže jednotlivé klasifikátory v databáze charakterizujú písmená, po každom porovnaní sme danému písmenu zvýšili jeho tzv. usefullness v závislosti od toho, ako veľmi sa na testovací vektor podobal. Usefullness v našom prípade reprezentuje užitočnosť daného klasifikátora.

Po preiterovaní všetkými klasifikátormi v databáze pre danú testovaciu vzorku sme spomedzi 26 písmen abecedy vybrali to, ktoré malo najväčšiu usefullness. Popritom sme si každou iteráciou pamätali aj úspešnosť jednotlivých funkcií. Tú sme si vyjadrovali ako rozdiel konkrétneho vypočítaného atribútu testovacieho písmena s tým atribútom klasifikátora z našej

databázy, s ktorým sme dané písmeno práve porovnávali. Vďaka tomu sme boli schopní vyjadriť úspešnosť každej funkcie, na základe ktorej už vieme vytriediť najprínosnejších 1024 features. Výsledkom a štatistikám jednotlivých funkcií sa bližšie budem venovať v ďalšej kapitole.

Kapitola 3

Štatistiky a výsledky

Táto kapitola je venovaná výsledkom a štatistikám nášho výskumu a zároveň aj ukážem, ako úspešné a prínosné boli jednotlivé features skúmané v [FS91], ktoré sme použili aj v našej práci.

3.1 Príprava

Počas učiacej fázy sa nám podarilo vytvoriť viacero databáz, nakoľko sme sa snažili hľadať najvhodnejšiu kombináciu parametrov pre distractions tak, aby písmená neboli až príliš zmodifikované, prípadne sme mierne upravovali algoritmy použité v učiacej fáze. Väčšinou sa nám podarilo jednotlivé procedúry upravovať tak, že nám novovytvorená databáza produkovala lepšie výsledky, ako tá predošlá.

Naša finálna databáza pozostáva zo 4000 klasifikátorov a z 1024 najprínosnejších funkcií na výpočet rôznych features, vďaka ktorým sme v pri testovacej fáze dosiahli úspešnosť približne 84%. Nás však zaujíma, ktoré parametre (alebo aj skupiny parametrov) použité vo funkciách sú tie prínosné. Na to, aby sme to zistili, sme využili väčšinu databáz, ktoré sme počas učiacej fázy vytvorili. Síce ich celkové úspešnosti boli o čosi menšie, ako v našej finálnej databáze, ale aj v nich sa vedeli prejaviť rozdiely medzi jednotlivými funkciami a to nám poskytovalo adekvátnu informáciu o jednotlivých parametroch.

Na to, aby sme boli schopní vyrábať takéto štatistiky, upravili sme si algoritmy v učiacej fáze tak, aby nám na záver učenia vytvorili zoznam použitých funkcií s jednotlivými parametrami a ich úspešnosťami do osobitného súboru, ktorý už vieme spracovávať rôznymi spôsobmi. Na meranie úspešností jednotlivých parametrov sme si zvolili dva druhy štatistík, nazvané Priemerové a Exponenciálne. Obidva typy majú svoje výhody aj nevýhody, pričom oba typy sú navrhnuté takým spôsobom, že vieme zmerať úspešnosti jedného parametra

alebo aj ľubovolnej kombinácie zložených z 2 a viac parametrov.

3.1.1 Priemerová štatistika

V tomto type štatistiky sa zameriavam na priemernú úspešnosť parametrov. Výhodou je, že výsledky nám poskytnú všeobecnú predstavu o danej skupine skúmaných parametrov. Ak je nejaká kombinácia parametrov podpriemerná, máme istotu, že táto skupina má pomerne malý prínos oproti tým nadpriemerným. Nevýhodou je, že tento typ štatistík trochu utláča parametre, ktoré sú pre nás prínosné ale zároveň sa vyskytujú aj v podpriemerne ohodnotených funkciách (To, že sa objavujú aj v podpriemere nám v porovnaní s tým, že daná kombinácia je aj veľmi úspešná, neprekáža, lebo takéto skupiny parametrov považujeme za prínosné).

3.1.2 Exponenciálna štatistika

Exponenciálna štatistika nám nevýhodu popísanú v Priemerovej celkom dobre rieši. Je totižto navrhnutá tak, aby skupiny parametrov, ktoré sú úspešné a prínosné, zvýraznila podstatne viac, ako tie, ktoré nám ponúkajú menšiu účinnosť. Inak povedané, ak máme skupiny parametrov, ktoré sú prínosné, ale zjavujú sa aj v tých málo účinných funkciách, štatistika tejto skupiny bude pomerne veľmi vysoká. Respektívne, bude dosahovať lepšie výsledky, ako skupina parametrov, ktorá sa vyskytuje len v priemerných funkciách (dá sa vidieť, že v Priemerovej štatistike by sa tieto 2 skupiny parametrov ich účinnosťou veľmi podobali). V tomto type štatistiky využívame vlastnosti exponenciálnej funkcie.

3.2 Výsledky a štatistiky výskumu

V tejto sekcii ukážem a popíšem výsledky, ktoré sme spracovali počas učiacej fázy. Tú sme vykonávali na 4 rôznych databázach, ktoré pri strojovom učení dosahovali rôznu úspešnosť. Výsledky štatistík sa však od seba líšili len minimálne, čo sme v podstate aj predpokladali. Tabuľky 3.1 a 3.2 zobrazujú štatistiky len pre jednu konkrétnu databázu, aby čitateľ dostal približnú predstavu o úspešnostiach a prínose jednotlivých parametrov. My sme preskúmali štatistiky všetkých 4 databáz pre rôzne skupiny parametrov, vypočítané pomocou Priemerovej a Exponenciálnej štatistiky, pričom tieto výsledky zhrniem v nasledujúcich sekciách. Výsledky pre obidva typy štatistík sú preškálované na interval $\langle 0,1 \rangle$, pričom platí, že čím je ohodnotenie väčšie, tým prínosnejšia je funkcia s danou kombináciou parametrov. Pripomeniem, že dané parametre sme používali vo funkcii

$$f_{power,k,phase}(i) = i^{power} \cdot \cos(k \cdot \pi \cdot i + phase)$$

pri iterovaní písmena vo vodorovnom aj horizontálnom smere, $i \in \langle -0.5, 0.5 \rangle$

3.2.1 Štatistika meraní s jedným parametrom

Pre každú zo 4 databáz, na ktorých sme skúmali štatistiky účinností parametrov meraných samostatne, nám vyšli podobné ohodnotenia pre Priemerový aj Exponenciálny typ. Tabuľka 3.1 nám zobrazuje štatistiku len z jednej databázy. Z výsledkov oboch typov štatistík sme zistili, že medzi naše naprínosnejšie funkcie patria tie, ktorých parametre powerX alebo powerY majú hodnotu 2. Zaujímavé je, že tieto isté parametre s hodnotami 0 dosahujú veľmi podpriemerné úspešnosti v porovnaní s ostatnými.

Typ štatistiky = Priemerová	
Úspešnosť	Kombinácie parametrov
1.0000000	powerY = 2
0.8834833	powerX = 2
0.8084604	powerX = 1
0.6937272	powerY = 1
0.6861963	kY = 0
0.6753768	kX = 0
0.6396732	phaseX = 1.5707
0.6174937	phaseX = 2.3561
0.6119659	phaseY = 1.5707
0.5879039	phaseX = 0.7853
0.5764816	phaseY = 0.7853
0.5597495	kY = 2
0.5486986	phaseY = 2.3561
0.5277702	kX = 2
0.5211568	phaseY = 0
0.4905802	kX = 1
0.4477814	kY = 1
0.4132321	phaseX = 0
0.0017834	powerX = 0
0.0000000	powerY = 0

Typ štatistiky = Exponenciálna	
Úspešnosť	Kombinácie parametrov
1.0000000	powerX = 2
0.8899565	powerY = 2
0.7366202	powerX = 1
0.7103466	kY = 0
0.6459274	powerY = 1
0.6349762	kX = 2
0.6319760	phaseY = 1.5707
0.6279037	phaseX = 2.3561
0.5964675	phaseY = 0.7853
0.5926944	kX = 0
0.5809444	phaseX = 0.7853
0.5752722	phaseX = 1.5707
0.5697388	phaseY = 0
0.5313732	phaseX = 0
0.5239079	kY = 2
0.5173112	phaseY = 2.3561
0.5089496	kX = 1
0.5023657	kY = 1
0.2007363	powerY = 0
0.0000000	powerX = 0

Tabuľka 3.1: Štatistiky - kombinácia s jedným parametrom

Čo sa ešte týka parametrov powerX a powerY s hodnotou 1, Priemerové a hlavne Exponenciálne štatistiky na všetkých skúmaných databázach nám ako výsledok poskytli ich pomerne vysokú účinnosť (viď aj v tabuľke 3.1). Parametre phaseX a phaseY sú pre všetky ich rôzne hodnoty zástancami priemernej úspešnosti. V praxi má tento parameter význam v miernom "vychýlení pixelov", čo sa v podstate javí ako parameter, ktorý pri výpočte veľké zmeny robiť nebude.

Výsledky, ktoré sme vyčítali ohľadom parametrov kY a kX sú nasledovné: Možno nie v tomto konkrétnom príklade, ako je tabuľka 3.1, ale podľa štatistík ostatných databáz sa aj tieto parametre s hodnotou 0 radia do kategórie prospešných, nakoľko všeobecne dosahujú pomerne vysoké hodnotenie, hlavne pre Exponenciálny typ štatistiky. Zvyšné hodnoty pre tieto parametre sa javia ako mierne podpriemerné, no napriek tomu sa o dosť lepšie uchytili v štatistikách pri kombinácii s dvomi parametrami.

3.2.2 Štatistika meraní s dvomi parametrami

Pri týchto štatistikách sme sa snažili nájsť tie najprínosnejšie kombinácie dvoch parametrov. Opäť, tabuľka 3.2 reprezentuje len jednu štatistiku, my sme ich preskúmali viacej a na základe nich odvodím nasledovné výsledky.

Medzi najprínosnejšie parametre opäť patria powerX a powerY s hodnotou 2. Tieto parametre sú pomerne dosť silné, nakoľko každá kombinácia jedného z nich a ľubovolného iného parametra s akoukoľvek hodnotou patrí medzi najviac ohodnotenú. Pri hlbšom preskúmaní všetkých štatistík sme spozorovali, že powerX a powerY s hodnotami 1 sú tiež pomerne úspešné, v Exponenciálnych typoch štatistík sa kombinácie s nimi vyskytujú väčšinou nad celkovým priemerom. Podobnými vlastnosťami vieme popísať aj parametre kX a kY s hodnotami 0.

Tabuľka 3.2 spolu s ďalšími výsledkami nám len potvrdzujú, že parametre phaseX a phaseY nám pre akékoľvek hodnoty s ľubovoľnou kombináciou iných parametrov poskytujú iba priemernú úspešnosť (v Exponenciálnom type sú dokonca menej úspešné, ako v Priemerovom). Na konci rebríčka úspešností sa aj naďalej držia powerX a powerY s hodnotami 0 s akýmikoľvek kombináciami. Ostatné parametre sú v štatistikách úspešností prerozdelené pomerne narovnako.

Typ štatistiky = Priemerová	
Úspešnosť	Kombinácie parametrov
1.0000000	powerY = 2, kY = 0
0.9642740	powerX = 2, powerY = 2
0.9260993	powerX = 2, kX = 1
0.9256448	powerX = 1, kX = 2
0.9030201	powerY = 2, phaseY = 0
0.9018355	powerX = 1, powerY = 2
0.8201060	powerY = 2, phaseY = 0.7853
0.8175208	phaseX = 1.5707, powerY = 2
0.8159406	phaseX = 2.3561, powerY = 2
0.8145322	powerY = 2, kX = 0
0.8092776	powerX = 2, phaseX = 2.3561
0.8045453	powerX = 2, phaseX = 1.5707
0.7960456	powerX = 2, kY = 0
0.7843309	powerY = 2, kX = 2
0.7805610	powerX = 1, kX = 0
0.7743953	powerX = 2, powerY = 1
0.7684927	phaseX = 0.7853, powerY = 2
0.7606093	powerX = 1, powerY = 1
0.7556481	powerX = 2, phaseY = 0.7853
0.7555131	powerX = 1, phaseX = 0.7853
0.7441159	powerX = 1, kY = 0
0.7439196	phaseX = 1.5707 kX = 1
0.7417632	powerY = 2, kX = 1
0.7415380	powerY = 1, kY = 2
0.7303883	powerY = 2, phaseY = 2.3561
0.7247650	powerX = 1, phaseY = 1.5707
0.7241943	powerX = 2, phaseY = 1.5707
0.7221697	powerY = 1, phaseY = 1.5707
:	:
0.2496944	powerY = 0, phaseY = 0.7853
0.2483893	powerX = 0, phaseX = 0.7853
0.2438435	powerY = 0, phaseY = 2.3561
0.2357978	powerY = 0, kY = 2
0.2352495	powerY = 0, kX = 2
0.2273752	powerY = 0, phaseY = 0
0.2148902	powerY = 0, kX = 1
0.2069014	powerX = 0, kY = 1
0.1961257	phaseX = 0, kX = 1
0.1830633	phaseX = 0, powerY = 0
0.1588906	powerX = 0, phaseX = 0
0.1217127	powerY = 0, kY = 1
0.0042694	powerX = 0, kX = 2
0.0000000	powerX = 0, powerY = 0

Typ štatistiky = Exponenciálna	
Úspešnosť	Kombinácie parametrov
1.0000000	powerX = 2, powerY = 2
0.9198445	powerX = 2, kX = 1
0.9196445	powerY = 2, kY = 0
0.8845946	powerX = 1, kX = 2
0.8645635	powerX = 2, kY = 0
0.8409519	powerX = 2, phaseX = 0.7853
0.8247408	powerX = 2, powerY = 1
0.8059717	powerX = 2, phaseY = 1.5707
0.8045931	powerX = 1, powerY = 2
0.8044698	powerY = 2, phaseY = 0
0.7902558	powerX = 2, phaseX = 2.3561
0.7895261	phaseX = 0, kX = 2
0.7802767	powerX = 2, phaseY = 0.7853
0.7559006	powerY = 2, phaseY = 0.7853
0.7551294	powerX = 2, phaseX = 1.5707
0.7464800	powerY = 2, kX = 2
0.7335674	powerX = 2, phaseY = 0
0.7236128	powerY = 2, phaseX = 2.3561
0.7186395	powerX = 2, phaseY = 2.3561
0.7166766	powerX = 2, kY = 1
0.7108358	phaseX = 1.5707 kX = 1
0.7016591	powerX = 1, kY = 0
0.6992780	powerY = 2, kX = 0
0.6976014	powerX = 2, kY = 2
0.6941802	phaseX = 1.5707, powerY = 2
0.6921887	powerX = 2, kX = 2
0.6799250	phaseX = 0.7853, powerY = 2
0.6719689	powerX = 1, phaseX = 2.3561
:	:
0.1619969	powerX = 0, powerY = 1
0.1567299	powerX = 0, phaseY = 0
0.1508249	powerX = 0, kY = 2
0.1492961	powerX = 0, phaseY = 1.5707
0.1469316	powerX = 0, phaseY = 0.7853
0.1298371	powerX = 0, phaseX = 2.3561
0.1245591	powerX = 0, phaseY = 2.3561
0.1175755	powerX = 0, phaseX = 0
0.1137368	phaseX = 0, kX = 1
0.1111894	powerX = 0, kX = 1
0.1058684	powerX = 0, phaseX = 0.7853
0.1043080	powerX = 0, kY = 1
0.0283322	powerX = 0, kX = 2
0.0000000	powerX = 0, powerY = 0

Tabuľka 3.2: Štatistiky - kombinácia s dvomi parametrami

3.3 Porovnanie výsledkov s výskumom [FS91]

Popri štatistikách hľadajúcich najprínosnejšie a najúspešnejšie parametre pre naše funkcie, zaujímali sme sa aj o features, ktoré používali Frey, Slate v ich výskume. Kedže vieme, s akými parametrami dané features vyjadriť, použili sme náš program na výpočet Priemernej a Exponenciálnej štatistiky. Naším cieľom bolo stanoviť poradie features použitých vo výskume [FS91] podľa ich najväčšieho prínosu a zároveň ich aj porovnať s ostatnými features, ktoré boli použité v našom výskume a tým vyjadriť ich absolútnu úspešnosť.

Štatistiky v tabuľkách 3.3, 3.4 a 3.5 zobrazujú výsledky len jednej databázy, aby čitateľ dostal všeobecnú predstavu o úspešnostiach jednotlivých features. Výsledky, ktoré popíšeme sme pozorovali na 4 databázach. Napriek tomu, že sme v každej používali inak zmodifikované znaky, štatistiky o úspešnostiach jednotlivých features sa rapídne nemenili a udržiavali si podobné hodnoty, ako sú zobrazované v tabuľkách.

3.3.1 Relatívna úspešnosť features v [FS91]

Tabuľky 3.3 a 3.4 nám zobrazujú úspešnosti jednotlivých features použitých v [FS91]. V týchto štatistikách sme sa snažili vyjadriť ich relatívnu úspešnosť, t.j. určiť, ako veľmi sa od seba dané features líšia. Štatistiky nám jednoznačne preukázali, že najúspešnejšie boli features využívajúce moduláciu diff(axis=X), ktorá nám na jednotlivých písmenkách zvýrazňuje ich zvislé hrany. Modulácia diff(axis=Y) až tak úspešná nebola. Feature, o ktorej sme si nemysleli, že bude patriť medzi najslabšie, čo sa týka úspešnosti, je s identifikátorom 9, ktorá vyjadruje ohodnotenie, ako veľmi je znak roztiahnutý v horizontálnom smere. O tejto vlastnosti si totiž myslíme, že nám poskytuje pomerne silnú informáciu o tvare písmena.

Z celkového rozloženia úspešností jednotlivých features sa dá hlavne v Exponenciálnom type štatistík vidieť, že prvé dve najúspešnejšie features (16 - súčet horizontálnych okrajov a 15 - priemerný počet okrajov, pri iterovaní písmenom zdola nahor) majú značnú silu v porovnaní s ostatnými, ktoré dosahujú v priemere maximálne tretinu ich úspešnosti. Potvrdzujú to aj výsledky štatistík robené na ostatných databázach.

Vo feature s identifikátorom 9 sa využívajú parametre powerX s hodnotou 2 a kX s hodnotou 1. Kombinácia týchto dvoch parametrov bola v celkovej štatistike pomerne veľmi silná, z čoho vyplýva, že modulácia identita sa javí byť ten slabý článok, ktorý nám trochu degraduje úspešnosti. Jednotlivým moduláciám sa ešte v našom výskume budem venovať trochu podrobnejšie.

Typ štatistiky = Priemerová		
Feature nr.	Úspešnosť	Kombinácie parametrov
16	1.0000000	mod = diff, axis = X, powerY = 1, kY = 1
15	0.9724016	mod = diff, axis = X, powerY = 2, kY = 1
13	0.7217900	mod = diff, axis = Y, powerX = 2, kX = 1
11	0.6005207	mod = ident, powerX = 2, kX = 1, powerY = 1, kY = 1
6	0.4411295	mod = ident, powerX = 1, kX = 0, phaseX = 0, kY = 0, phaseY = 0
10	0.4157125	mod = ident, powerY = 2, kY = 1, phaseY = 0
8	0.4071106	mod = ident, powerX = 1, kX = 1, powerY = 1, kY = 1
14	0.3377779	mod = diff, axis = Y, powerX = 1, kX = 1
7	0.2441167	mod = ident, powerY = 1, kY = 0, phaseY = 0, kX = 0, phaseX = 0
12	0.2016985	mod = ident, powerX = 1, kX = 1, powerY = 2, kY = 1
9	0.0000000	mod = ident, powerX = 2, kX = 1, phaseX = 0

Tabuľka 3.3: Priemerová štatistika - features použité v [FS91]

Typ štatistiky = Exponenciálna		
Feature nr.	Úspešnosť	Kombinácie parametrov
16	1.0000000	mod = diff, axis = X, powerY = 1, kY = 1
15	0.9656130	mod = diff, axis = X, powerY = 2, kY = 1
13	0.4961636	mod = diff, axis = Y, powerX = 2, kX = 1
6	0.2917358	mod = ident, powerX = 1, kX = 0, phaseX = 0, kY = 0, phaseY = 0
7	0.2520411	mod = ident, powerY = 1, kY = 0, phaseY = 0, kX = 0, phaseX = 0
11	0.2443894	mod = ident, powerX = 2, kX = 1, powerY = 1, kY = 1
10	0.2089067	mod = ident, powerY = 2, kY = 1, phaseY = 0
8	0.1880863	mod = ident, powerX = 1, kX = 1, powerY = 1, kY = 1
14	0.1757814	mod = diff, axis = Y, powerX = 1, kX = 1
12	0.0453082	mod = ident, powerX = 1, kX = 1, powerY = 2, kY = 1
9	0.0000000	mod = ident, powerX = 2, kX = 1, phaseX = 0

Tabuľka 3.4: Exponenciálna štatistika - features použité v [FS91]

3.3.2 Absolútna úspešnosť features v [FS91]

Tabuľka 3.5 nám zobrazuje jednu zo 4 štatistík, na ktorých sme skúmali absolútne úspešností jednotlivých features použitých v práci [FS91]. Porovnávali sme ich so všetkými features použitých v našom výskume.

Typ štatistiky = Priemerová	
Feature nr.	Úspešnosť
16	0.8219207
15	0.8081631
11	0.6951432
10	0.6297079
6	0.6162618
13	0.5287018
8	0.5160866
12	0.5029242
7	0.4818208
14	0.2976022
9	0.2581065

Typ štatistiky = Exponenciálna	
Feature nr.	Úspešnosť
16	0.6880289
15	0.6858214
6	0.3919627
10	0.3787913
11	0.3749093
7	0.3473986
13	0.2973269
8	0.2610946
12	0.1840842
14	0.1144202
9	0.0652537

Tabuľka 3.5: Štatistiky - absolútna úspešnosť features použitých v [FS91]

Výsledky minulej štatistiky nám už napovedali približné poradie jednotlivých features. Tieto tabuľky nám poskytli informácie, že features, ktoré používali P. Frey a D. Slate, spojite pokrývajú celú škálu úspešností v porovnaní s našimi features, z čoho sa dá predpokladať, že ich P. Frey a D. Slate vyberali náhodnou voľbou, resp. sa im úspešne podarilo nájsť všeobecné vlastnosti písmen. Features s najväčším prínosom sú opäť s identifikátormi 15 a 16.

V týchto tabuľkách už je vidieť pomerne veľký rozdiel medzi jednotlivými hodnotami v Priemerovom a Exponenciálnom type štatistiky. Je to spôsobené tým, že sada, z ktorej sme tieto features selektovali obsahuje obrovské množstvo funkcií zoradených od najprínosnejších po najmenej prínosné, a teda až tu sa mohol naplno prejaviť skutočný rozdiel medzi jednotlivými výsledkami v meraní úspešností pri oboch typoch štatistík.

3.4 Štatistiky modulácií

V doterajších štatistikách sme sa podrobne nevenovali úspešnostiam jednotlivých modulácií. Na základe predošlých výsledkov sa zdá, že modulácia diff(axis=X) bude tou najprínosnejšou. Tabuľka 3.6 nám znázorňuje úspešnosti daných modulácií.

Typ štatistiky = Priemerová	
Úspešnosť	Typ modulácie
1.0000000	mod = diff, axis = X
0.3933304	mod = diff, axis = Y
0.0000000	mod = ident

Typ štatistiky = Exponenciálna	
Úspešnosť	Typ modulácie
1.0000000	mod = diff, axis = X
0.2611019	mod = diff, axis = Y
0.0000000	mod = ident

Tabuľka 3.6: Štatistiky - modulácie

Na základe všetkých výsledkov, ktoré nám štatistiky vygenerovali je zjavné, že modulácia diff(axis=X) je mnohonásobne účinnejšia, ako identita alebo diff(axis=Y). Táto modulácia spôsobuje, že cez dané písmenko iterujeme horizontálne a vyjadrujeme rozdiel každých dvoch susedných pixelov, čiže v podstate vyznačujeme zvislé hrany písmena.

3.5 Najviac a najmenej prínosné features

Tabuľka 3.7 nám zobrazuje časť výsledkov, ktoré sme dostali po usporiadaní všetkých použitých features podľa ich prínosu (úspešnosti). Zo všetkých 4 štatistík, ktoré sme preskúmali sa dajú vyvodiť nasledujúce závery: Najúspešnejšie skončili features s moduláciou diff(axis=X), powerX=2, powerY=2 a powerY=1. Najmenej úspešné features boli tvorené prevažne moduláciou identita. Parametre powerX a powerY s hodnotami 0 sa ukázali ako najmenej prínosné pri akejkoľvek kombinácii s inými parametrami. Ostatné parametre, ako sú phaseX, phaseY, kX, kY sa pre všetky ich hodnoty vyskytujú vo funkciách rovnomerne po celej škále úspešností.

Najviac a najmenej prínosné features							
Feature nr.	Modulácia	powerX	kX	phaseX	powerY	kY	phaseY
1	diff(axis = X)	2	2	0	2	1	0.7853
2	diff(axis = X)	1	2	0	1	2	0.7853
3	diff(axis = X)	2	1	2.3561	2	1	0.7853
4	diff(axis = X)	2	2	0	2	2	2.3561
5	diff(axis = X)	2	1	1.5707	2	1	0.7853
6	diff(axis = X)	2	1	2.3561	2	1	1.5707
7	diff(axis = X)	2	1	2.3561	1	0	0
8	diff(axis = X)	2	1	2.3561	1	0	0.7853
9	diff(axis = X)	2	1	2.3561	1	0	1.5707
10	diff(axis = X)	2	1	2.3561	1	0	2.3561
11	diff(axis = X)	2	2	0	1	2	2.3561
12	diff(axis = X)	1	1	2.3561	1	2	0.7853
13	diff(axis = X)	2	1	2.3561	0	1	1.5707
14	diff(axis = X)	1	2	0.7853	1	2	0.7853
15	diff(axis = X)	2	1	2.3561	1	2	2.3561
16	diff(axis = X)	2	1	2.3561	2	2	1.5707
17	diff(axis = X)	2	1	2.3561	1	2	1.5707
18	diff(axis = X)	2	1	2.3561	2	1	0
19	diff(axis = X)	2	2	0.7853	1	1	1.5707
20	diff(axis = X)	2	1	2.3561	2	1	2.3561
:	:	:	:	:	:	:	:
3875	diff(axis = Y)	0	2	0	0	2	1.5707
3876	diff(axis = Y)	0	2	0.7853	1	1	0
3877	diff(axis = Y)	1	1	0	0	2	0
3878	diff(axis = Y)	0	2	0	2	2	1.5707
3879	identita	0	2	0.7853	0	2	0.7853
3880	diff(axis = Y)	0	2	0	1	1	0
3881	identita	0	2	0.7853	0	2	0
3882	diff(axis = Y)	0	2	0	1	0	2.3561
3883	diff(axis = Y)	0	2	0	1	0	0
3884	diff(axis = Y)	0	2	0	1	0	0.7853
3885	diff(axis = Y)	0	2	0	1	0	1.5707
3886	diff(axis = Y)	0	2	0	0	1	0.7853
3887	diff(axis = Y)	0	2	0	0	1	1.5707
3888	identita	0	2	0	0	2	0

Tabuľka 3.7: Najviac a najmenej prínosné features

3.6 Sumarizácia výskumu

Vo všeobecnosti, z celkového hodnotenia výsledkov štatistík sa dá povedať, že parametre, ktoré boli pre všetky features najpodstatnejšie, sú powerX, powerY a modulácia. V závislosti od hodnôt týchto parametrov sa v prevažnej časti určovalo, či bude daná feature viac alebo menej prínosná. Zaujímavým zistením však bolo, že aj goniometria, ktorú sme v našich funkciách implementovali, sa zachovala celkom dôležito a hrala v úspešnostiach jednotlivých features pomerne dôležitú úlohu. Domnievame sa, že ak by aj vo výskume [FS91] P. Frey a D. Slate implementovali goniometrické funkcie vo väčšom počte, dosiahli by ešte lepšie výsledky.

Čo sa týka úspešnosti pri rozpoznávaní písmen v testovacej fáze, dosahovali sme rôzne výsledky, nakoľko sme si vyprodukovali početné množstvo rôznych databáz, na ktorých sme testovacie sady skúšali. Najlepšie sa nám podarilo dosiahnuť úspešnosť 84,2%, čo si myslíme, že je veľmi dobrý výsledok. Sme si vedomý faktu, že v našej práci sa vyskytujú aj prípady, ktoré by sa dali riešiť možno lepšie, takže je vysoko pravdepodobné, že naše výsledky sa dajú vylepšiť ešte viac, a to hlbším skúmaním, analyzovaním a testovaním.

Záver

V našej práci sme skúmali jednu z mnohých techník strojového rozpoznávania písmen. Nadviazali sme na prácu P. Frey a D. Slate [FS91], ktorí sa tento problém snažili riešiť používaním klasifikátorského systému podloženým na báze rôznych techník strojového učenia. Zatiaľ čo sa snažili nájsť tú najvhodnejšiu kombináciu použitých metód pri strojovom učení, podarilo sa im dosiahnuť úspešnosť rozpoznávania približne 82%. V našom výskume sme chceli zistiť, či kompletná modifikácia niektorých podstatných procedúr dokáže zvýšiť kvalitu rozpoznávania. Oblasti, o ktorých sme si mysleli, že sú vhodní kandidáti na hlbšie úpravy, boli znaková sada, počet meraných features a strojové učenie.

V našej práci sme si znakovú sadu zloženú z rôznych fontov upravili tromi deformačnými funkciami. Dôvodom bolo zaistiť jedinečnosť každého písmena v sade a všeobecnosť pri používaní v praxi. Podstatne sme zmenili aj počet features, vďaka ktorým vieme číselne vyjadriť vlastnosti jednotlivých písmen. Táto zmena nám poskytla pozitívne výsledky, nakoľko sme si následne mohli strojovo vybrať tie funkcie, ktoré sú pre nás najprínosnejšie, a tým zvýšiť kvalitu rozpoznávania. Strojové učenie, ktoré sme praktizovali v našom výskume, sme oproti [FS91] zjednodušili a jednotlivé podprípady sme nerozvetvovali.

Na základe dvoch typov štatistík sme boli schopní zistiť úspešnosti všetkých features použitých vo výskume. Tieto štatistiky sme využili aj pri ohodnocovaní features použitých v práci Frey-Slate. Výsledky nám poskytli potrebné informácie na to, aby sme zistili, že ich features spojite pokrývajú celú škálu úspešností v porovnaní s našimi features. Vďaka štatistikám sme boli schopní vyselektovať a popísať tie kombinácie parametrov použitých vo funkciách, ktoré sú pre nás najprínosnejšie.

Touto prácou sme sa snažili nadviazaním na výskum [FS91] zlepšiť kvalitu rozpoznávania písmen, čo sa nám aj mierne podarilo. Sme si vedomý toho, že existuje ešte mnoho iných prístupov, ktoré sa túto problematiku snažia riešiť, a pravdepodobne aj poskytujú lepšie úspešnosti. Dokázali sme však, že metóda, ktorú sme skúmali, sa dá rôznymi modifiká-

ciami upravovať tak, aby nám poskytovala čoraz kvalitnejšie výsledky. V našej práci sme počas skúmania postrehli mnoho položiek, ktorých implementáciou alebo zmenou by sme možno dospeli k ešte priaznivejším výsledkom, avšak tieto kategórie neboli predmetom skúmania.

Literatúra

[Fog92] Terence C. Fogarty. *First Nearest Neighbor Classification on Frey and Slate's Letter Recognition Problem*. Kluwer Academic Publishers, Boston, 1992.

[Fou] Python Software Foundation. Python programming language. `http://www.python.org/`.

[FS91] Peter W. Fray and David J. Slate. *Letter Recognition Using Holland-Style Adaptive Classifiers*. Kluwer Academic Publishers, Boston, 1991.

[MSY92] Shunji Mori, Ching Y. Suen, and Kazuhiko Yamamoto. *Historical Review of OCR Research and Development*. Ricoh Co. Ltd., Yokohama, Japan, July 1992.

[Num] Numpy and scipy documentation. `http://docs.scipy.org/doc/`.

[Rou] Margaret Rouse. Machine learning. `http://whatis.techtarget.com/definition/machine-learning`.