TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

Využití algoritmů dataminingu pro rozpoznávání pojmenovaných entit

Diplomová práce

Studijní program: N2612 – Elektrotechnika a informatika
Studijní obor: 1802T007 – Informační technologie

Autor práce: **Bc. Vojtěch Houžvička**
Vedoucí práce: Ing. Pavel Tyl

Liberec 2015

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

Datamining algorithms for named entity recognition

Diploma thesis

Study programme: N2612 – Electrical engineering and informatics
Study branch: 1802T007 – Information technology

Author: **Bc. Vojtěch Houžvička**
Supervisor: Ing. Pavel Tyl

Liberec 2015

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: 2014/2015

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Bc. Vojtěch Houžvička**
Osobní číslo:	**M13000188**
Studijní program:	**N2612 Elektrotechnika a informatika**
Studijní obor:	**Informační technologie**
Název tématu:	**Využití algoritmů dataminingu pro rozpoznávání pojmenovaných entit**
Zadávající katedra:	**Ústav mechatroniky a technické informatiky**

Zásady pro vypracování:

1. Seznamte se s problematikou rozpoznávání a identifikace pojmenovaných entit.
2. Prostudujte dataminingové algoritmy, které lze využít pro rozpoznávání či identifikaci pojmenovaných entit.
3. Analyzujte existující postupy a nástroje řešící uvedenou problematiku a navrhněte vlastní implementaci.
4. Navržené řešení vyhodnoťte například pomocí přesnosti a úplnosti. K tomu použijte libovolnou množinu dat odpovídající danému návrhu.

Rozsah grafických prací: **dle potřeby dokumentace**

Rozsah pracovní zprávy: **40–50 stran**

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **ŠEŠERA, L. - MIČOVSKÝ, A. - ČERVEŇ, J.: Datové modelování v příkladech. Grada Publishing, Praha, 2001. ISBN 80-247-0049-2.**

[2] **KNUTH, Donald Ervin: Umění programovat. Computer Press, Praha, 2008. ISBN 80-251-2025-2.**

[3] **PARR-RUD, Olivia: Datamining. Computer Press, Praha, 2001. ISBN 80-722-6577-6.**

[4] **Message Understanding Conference (MUC) Proceedings. Online: http://www-nlpir.nist.gov/related_projects/muc.**

Vedoucí diplomové práce: **Ing. Pavel Tyl**
Ústav mechatroniky a technické informatiky

Datum zadání diplomové práce: **10. října 2014**

Termín odevzdání diplomové práce: **15. května 2015**

V. Kopecký

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

Technická univerzita v Liberci - 14 -

Kolář

doc. Ing. Milan Kolář, CSc.
vedoucí ústavu

V Liberci dne 10. října 2014

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 15. 5. 2015

Podpis: Jhoužvička

Abstrakt

Tato práce se zabývá vyhledáváním pojmenovaných entit v textu pomocí dataminingových algoritmů a jejím cílem je navrhnout nástroj, který by tento problém řešil. Součástí práce je rešerše existujících nástrojů a algoritmů, které se danou problematikou zabývají.

Dále byl navržen a implementován nástroj, který využívá několik dataminingových algoritmů zároveň a kombinuje tak jejich výhody. Konkrétní algoritmy jsou realizovány pomocí externích knihoven a výsledky vyhledávání jednotlivých algoritmů jsou sloučeny pomocí vlastní navržené metody. Tato metoda bere v potaz předchozí úspěšnost nástrojů a vybírá ze všech výsledků ten nejvíce pravděpodobný. Práce také popisuje vytvoření datového modelu pro naučení nástroje. Pomocí vlastního modelu lze nástroj použít nad libovolnou doménou dat. Celý nástroj je uzpůsobený na vyhledávání entit v českém jazyce. V závěru práce je nástroj na vytvořeném datovém modelu otestován pomocí přesnosti a úplnosti.

Klíčová slova

Vyhledávání pojmenovaných entit, datamining, dolování dat, strojové učení, extrakce informace, algoritmy dataminingu, morfologická analýza, pojmenovaná entita

Abstract

This thesis concerns itself with named entity recognition and use of data mining algorithms for this purpose. Its main objective is to design and implement a tool, that solves the problem of named entity recognition. This thesis contains research of existing tools for named entity recognition and research of data mining algorithms.

A new tool for named entity recognition was designed and implemented. This tool combines several data mining algorithms and dictionary method and takes advantage of their strong points by merging their results using own designed method. Each algorithm is implemented by external tool. The method for results merging uses previous precision of included tools to determine most probable results. The thesis also covers the topic of creating own training data set. The tool was trained and tested using data set created within the diploma thesis.

Key words

Named entity recognition, data mining, machine learning, information extraction, data mining algorithms, morphological analysis, named entity

Poděkování

Děkuji vedoucímu práce Ing. Pavlu Tylovi za cenné rady, věcné připomínky a vstřícnost při konzultacích a při vypracování diplomové práce.

Děkuji svým rodičům, Janovi a Jitce, za jejich duchovní i materiální podporu a poskytnutí zázemí pro odpočinek a nabírání nových sil v samém srdci Českého středohoří.

Děkuji také slečně Markétě Kostelencové, za častou motivaci k práci, podporu ve chvílích nejistoty a pomoc s korekturami textu.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

API	Application Interface
CNEC	Czech Named Entity Corpus
CRF	Conditional Random Field
CSS	Cascading Style Sheets
ČSFD	Česko-Slovenská Filmová Databáze
DOM	Document Object Model
HMM	Hidden Markov Model
HTML	HyperText Markup Language
ICDM	International Conference on Data Mining
IP	Internet Protocol
JS	JavaScript
LIPI	LIng PIpe
MANER	Multiple Algorithm Named Entity Recognizer
MorphoDiTa	Morphological Dictionary and Tagger
MUC	Message Understanding Conference
NE	Named Entity
NER	Named Entity Recognition
NLP	Natural Language Processing
RAM	Random Access Memory
SNER	Stanford Named Entity Recognition
SVM	Support Vector Machines
XML	eXtended Markup Language

1 Úvod

Informace je ropou 21. století a analýza spalovacím motorem.

Peter Sondergaard, Gartner Research

Podle sedmé výroční studie EMC Digital Universe objem dat obsažených v digitálním světě vzroste desetkrát do roku 2020 [1]. Ačkoliv data nesou sama o sobě užitečné informace, jejich analýzou lze získat informace nové, nacházet dříve neviděné vztahy a objevovat nové souvislosti. A tak data (informace) v dnešní době tvoří jednu z nejvzácnějších komodit.

Nechat data bez analýzy znamená ztrácet jejich potenciální hodnotu. Nejedná se pouze o prvoplánově užitečná data, ale také o takzvané *datové zplodiny*, neužitečná data, ze kterých lze jejich analýzou získat užitečné informace. Tento pojem je použit v knize Big Data [2], kde je uveden na názorném příkladu shromažďování informací o chování čtenáře elektronické knihy. Jak dlouho stráví čtením jedné stránky, kde čtou, kde si podtrhnou nějakou pasáž atd. Výrobce elektronických knih pak tyto informace shromáždí, analyzuje a pokusí se poskytnout čtenáři ještě lepší zážitek ze čtení. Jiný příklad užití je analýza známých dat o zákaznících finanční společnosti, na základě které pak společnost předvídá, zda novému zákazníkovi udělí půjčku či nikoliv [3].

Tento proces, kdy se z dat těží nová netriviální data, se nazývá datamining (doslova dolování dat). A právě s rychlým nárůstem digitálního obsahu stoupá využití a obliba dataminingu. Jednou z mnoha oblastí, kterou se datamining zabývá, je extrakce informace z textu, konkrétněji vyhledávání pojmenovaných entit v textu. O tom, jaké nástroje a postupy datamining při vyhledávání pojmenovaných entit nabízí, pojednává tato diplomová práce. Zaměřuji se v ní především na to, které konkrétní algoritmy pro dolování dat jsou k této činnosti vhodné a jak je použít.

2 Pojmenované entity a dataminingové algoritmy

Označením *pojmenovaná entita* rozumíme slovo nebo víceslovné spojení, které jednoznačně identifikuje objekt či entitu. Těmito entitami jsou nejčastěji osoby, organizace, města, geografická území, data nebo časová rozmezí, výrazy množství a jiné [4].

Termín *pojmenovaná entita* byl zaveden v roce 1996 na šesté Message Understanding conference. Tato konference, která se poprvé konala roku 1987, se organizuje za účelem lepšího porozumění a vyvinutí přesnějších metod pro extrakci informace. Konference je založena na principu soutěžení několika týmů o nejlepší výsledky v adaptování různých postupů při snaze o extrakci informace z textu [5].

2.1 Úloha rozpoznávání pojmenovaných entit v textu

Rozpoznávání pojmenovaných entit (NER) je jedna z úloh používaných v dataminingu. Jejím cílem je odhalit, kde se v textu nachází pojmenované entity. Vstupem pro vyhledávání je blok textu a výsledkem pak stejný blok textu, ve kterém jsou anotovány pojmenované entity. Například pro vstup:

Roy Raymond, zakladatel značky Victoria's Secret, která má k datu 1. ledna 2014 cenu zhruba pět miliard dolarů, spáchal v roce 1993 sebevraždu skokem z Golden Gate Bridge deset let poté, co značku prodal Lesliemu Wexnerovi za čtyři miliony dolarů.

Může být výstupem:

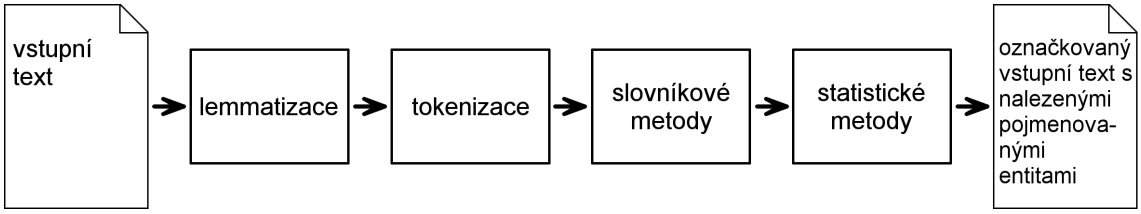
[Roy Raymond]$_{\text{Osoba}}$, zakladatel značky [Victoria's Secret]$_{\text{značka}}$, která má k datu [1. ledna 2014]$_{\text{Datum}}$ cenu zhruba [pět miliard dolarů]$_{\text{množství}}$, spáchal v roce [1993]$_{\text{Datum}}$

sebevraždu skokem z [Golden Gate Bridge]$_{\text{stavba}}$ [deset let]$_{\text{časový údaj}}$ poté, co značku prodal [Lesliemu Wexnerovi]$_{\text{osoba}}$ za [čtyři miliony dolarů]$_{\text{množství}}$.

V tomto případě bylo ve větě detekováno a klasifikováno osm pojmenovaných entit. Atomické části názvu entity (slova, interpunkční znaménka) jsou označovány jako *tokeny*. Toto pojmenování se do češtiny nepřekládá. Pojmenovaná entita *Golden Gate Bridge* je třítokenová a klasifikována třídou stavba. Zároveň je patrné, že jediná pojmenovaná entita může být definována mnoha různými úseky textu (Roy Raymond, zakladatel značky Victoria's Secret) a entity mohou být do sebe vnořené ([zakladatel značky [Victoria's Secret]$_{\text{značka}}$]$_{\text{Osoba}}$). Jak budeme entitu vnímat, pak zcela záleží na tom, za jakým účelem text zkoumáme.

2.2 Běžný postup při vyhledávání pojmenovaných entit

Obrázek 2.1 popisuje, jak se zpravidla postupuje při rozpoznávání pojmenovaných entit. Vstupní text se nejprve rozdělí na větné celky (lemmatizace), následně se se rozdělí na tokeny (tokenizace), provede se vyhledávání pojmenovaných entit pomocí slovníků a poté pomocí statistických metod. Výsledkem je označkovaný vstupní text, s nalezenými pojmenovanými entitami. Obrázek 2.1 popisuje způsob, jakým funguje hybridní nástroj (nástroj kombinuje užití slovníkových a statistických metod).



Obrázek 2.1: Běžný postup při rozpoznávání pojmenovaných entit

2.3 Způsoby rozpoznávání pojmenovaných entit v textu

Přístup k rozpoznání pojmenovaných entit je dvojí:

- Slovníkové metody
- Metody založené na statistickém modelu

2.3.1 Slovníkové metody

Slovníkové metody využívají k nalezení entit v textu předem definovaný slovník pojmů. Obvykle dosahují větší přesnosti než metody založené na statistickém modelu, za cenu času stráveného při přípravě slovníku – jeho vytvoření může zabrat i měsíce práce. Pomocí tohoto přístupu lze nalézt pouze entity definované slovníkem a pokrytí je tedy omezené. Dalším úskalím (nejen této metody) jsou začátky vět, kde nelze rozhodnout, zda velké písmeno označuje začátek věty, nebo se jedná o vlastní jméno. Například:

Nečas zahalil krajinu neproniknutelnou záclonou deště.
Nečas podal demisi 17. června 2013.

Že se ve druhém případě jedná o českého politika, tedy o pojmenovanou entitu, zjistíme pouze z kontextu věty. Stroj by v obou případech rozhodl stejně – označil by výraz *Nečas* jako pojmenovanou entitu buď v obou větách, nebo v žádné. V obou případech by se tak dopustil chyby.

2.3.2 Metody založené na statistickém modelu

Metody založené na statistickém modelu k rozpoznání entit využívají strojové učení, při kterém je anotována pouze malá část trénovacích dat. Stroj využije tuto trénovací sadu, aby se z ní *naučil* pracovat s daty, která v této sadě nejsou. K učení se používá některý z dataminingových algoritmů, který se vybírá podle typu a zaměření úlohy. Přesnost této metody je vždy menší než přesnost slovníkové metody. Tato metoda je naopak flexibilnější, protože dokáže rozhodovat i o datech, která nejsou v trénovací

sadě – má tedy větší pokrytí. V příkladu uvedeném v kapitole 2.3.1 by statistický model už mohl rozhodnout správně v obou případech, pokud by k rozpoznávání pojmenovaných entit využil například širšího kontextu věty.

2.4 Metriky pro měření úspěšnosti NER nástroje

Aby jednotlivé nástroje bylo možné mezi sebou porovnávat, byly zavedeny ukazatele pro měření jejich výkonu [6]. Tyto ukazatele vychází z následujících hodnot naměřených při práci nástroje.

- *True Positive (TP)* – stroj správně označil výraz jako pojmenovanou entitu.
- *False Negative (FN)* – výraz, který je pojmenovanou entitou, nebyl strojem rozpoznán.
- *False Positive (FP)* – stroj označil výraz za pojmenovanou entitu, ačkoliv se o pojmenovanou entitu nejedná.

Pomocí těchto hodnot se vypočtou tři ukazatele, které určují kvalitu použité metody pro rozpoznání pojmenovaných entit. Těmito ukazateli jsou:

- Přesnost (precision) – dána jako poměr TP / (TP + FP), udává poměr správně nalezených pojmenovaných entit vůči všem nalezeným entitám.
- Úplnost (recall) – definováno jako TP / (TP + FN), udává poměr entit, které byly správně označeny jako pojmenované entity, vůči všem entitám v textu.
- F-míra (F-measure) – je nejčastěji definována jako harmonický průměr přesnosti a úplnosti. Tedy vztahem:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2.1}$$

  F-míra zahrnuje přesnost i úplnost a má nejpřesnější vypovídací hodnotu o použité metodě. Ve výše uvedeném vzorci je na přesnost i úplnost kladen stejně velký důraz. Někdy se používá vyjádření, které klade větší důraz na

přesnost (např. $F_{0,5}$) nebo na úplnost (např. F_2). Výsledná F-míra se pak dopočítává podle vztahu:

$$F_\lambda = (1 + \lambda^2) \cdot \frac{precision \cdot recall}{(\lambda^2 \cdot precision) + recall} \tag{2.2}$$

Tyto ukazatele se vzájemně doplňují v tom smyslu, že souvisí s opačným typem chyb.

2.5 Strojové učení a hledání pojmenovaných entit

Strojové učení je vědecká disciplína, která se zabývá algoritmy, pomocí nichž se stroj dokáže naučit samostatnému rozhodování. Učení probíhá následujícím způsobem: Stroji je předložena sada trénovacích dat, ze kterých vytvoří model. Na základě tohoto modelu je pak schopen rozhodnout, jak zacházet s novými daty, aniž by se tato data nacházela v trénovací sadě. Učení probíhá některou z obecných technik strojového učení, typicky za využití algoritmu SVM (Support Vector Machines), HMM (Skryté Markovovy Modely) nebo CRF (Conditional Random Fields).

V úloze hledání pojmenovaných entit tvoří trénovací sadu text, ve kterém jsou pojmenované entity anotovány. Z tohoto textu stroj vytvoří model, pomocí kterého klasifikuje pojmenované entity v neanotovaném textu. Účinnost tohoto přístupu je podmíněna jak velkým množstvím anotovaných dat, tak algoritmem požitým pro učení a rozhodování stroje. Aby se předešlo zbytečnému úsilí věnovanému anotaci dat, používá se takzvaný *semisupervised* přístup – kombinace strojového učení s učitelem a bez učitele – anotovaná data tvoří pouze malou část trénovací sady, zbytek dat je neanotovaný.

Některé nástroje mohou tyto metody kombinovat a ze správně určených entit vytvářet slovníky, ze kterých se dále učí a používají je při budoucích vyhledáváních.

2.6 Úskalí vyhledávání pojmenovaných entit

Ačkoliv se na úloze NER pracuje již od devadesátých let, jsou dnes systémy NER stále omezené v tom smyslu, že systém vyvinutý pro konkrétní doménu textů nebude

ideálně fungovat nad jinou doménou. Přetvoření systému tak, aby fungoval nad novou doménou, může stát stejné úsilí jako tvorba nového systému. To platí jak pro systémy využívající statistický model, tak pro systémy založené na slovníkových metodách.

Stěžejní domény, kterými se dnes NER zabývá jsou: novinové články, bioinformatika, molekulární biologie, vojenské zprávy, dotazy zadávané do vyhledávačů, lékařské zprávy atd.

Dalším úskalím je rozdílnost jazyků a jejich bohatost na různé výjimky. V češtině se jediná entita může vyskytovat v mnoha různých tvarech a je potřeba, aby ji systém vždy klasifikoval správně. Jiným specifikem jsou jazyky, jejichž abecedy obsahují velké množství znaků, jako například čínština se zhruba padesáti tisíci znaky.

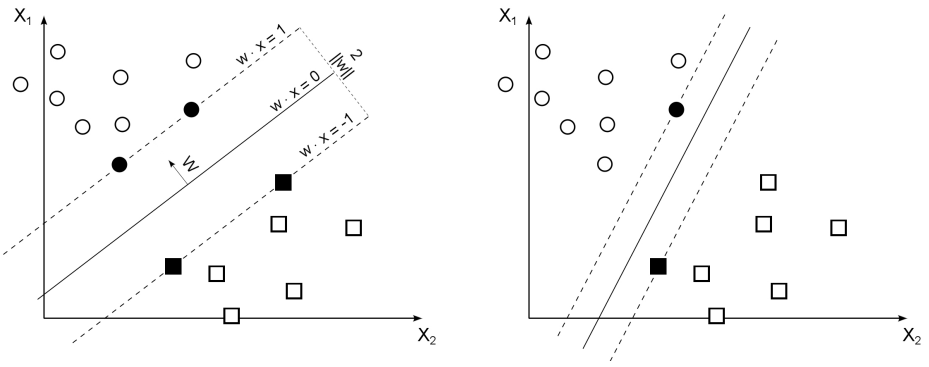
2.7 Algoritmy pro vyhledávání pojmenovaných entit v textu

Obecně je dataminingový algoritmus konečná sada pravidel, která ze vstupních dat vytvoří statistický model. Tento model je dále využit ke klasifikaci neznámých dat. Použitelnost algoritmu závisí na povaze dataminingové úlohy – algoritmus vhodný pro jednu úlohu, nemusí být vhodný pro jinou. Neexistuje tedy takový algoritmus, který by byl použitelný na všechny typy úloh a zároveň vykazoval u všech nejlepší výsledky.

Pro rozpoznávání pojmenovaných entit je vhodných algoritmů více. Nejčastěji používané a časem ověřené jsou algoritmus podpůrných vektorů (Support Vector Machines – SVM), Skrytý Markovův model (Hidden Markov Model – HMM) a podmíněná náhodná pole (Conditional Random Fields – CRF). Tyto algoritmy jsem také využil při realizaci vlastní implementace nástroje pro NER. Algoritmy jsem vybral na základě výsledků práce ICDM, která srovnává nejpoužívanější základní algoritmy dataminingu [7].

2.7.1 Support Vector Machines

Algoritmus podpůrných vektorů je relativně novým přístupem ke strojovému učení, který umožňuje řešit problém rozdělení do dvou tříd (jedná se tedy o binární klasifikátor). Podpůrný vektor je reprezentant trénovací sady, který slouží k vytvoření rozhodovací nadroviny, podle které algoritmus rozděluje vstupní vektory do tříd. Těchto rozhodovacích nadrovin můžeme nalézt více než jednu, proto je hledání rozhodovací roviny optimalizační úloha, viz obrázek 2.2. Podpůrné vektory jsou právě body, které tuto rovinu popisují.



(a) Optimální nadrovina (b) Neoptimální nadrovina

Obrázek 2.2: Hledání optimální nadroviny v úloze SVM

Mějme sadu trénovacích dat, která jsou klasifikována do dvou tříd:

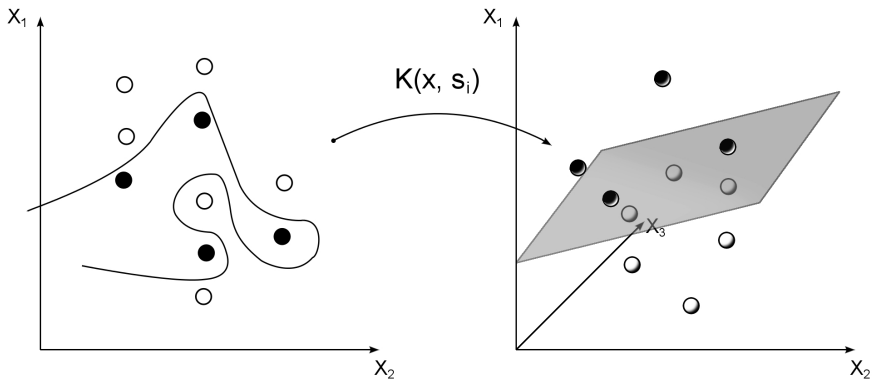
(x_1, Y_1) ... (x_n, Y_n), $Y_i \in$ -1, +1

kde $x_i \in \mathbb{R}^n$ je vektor vlastností i-tého vzorku ze sady trénovacích dat a Y_i je třída do které x_i náleží. Cílem algoritmu je nalézt rozhodovací funkci, která s dostatečnou přesností určí třídu Y pro vstupní vektor x. Nelineární SVM klasifikátor přiřadí každému vstupnímu vektoru x_i rozhodovací funkci $f(x) = sign(g(x))$, kde

$$g(x) = \sum_{i=1}^{m} w_i K(x, s_i) + b$$

Pokud je pro vstupní vektor x $f(x) = 1$, znamená to, že x je prvkem třídy Y, pokud $f(x) = -1$, pak x není prvkem třídy Y. Symbol s_i označuje podpůrný vektor a m je počet podpůrných vektorů. Výpočetní složitost funkce $g(x)$ je tedy přímo úměrná číslu m. $K(x, s_i)$ je *jádro*, které mapuje vstupní vektory do prostoru s vyšší

dimenzí, než je dimenze vektoru. Umožňuje tak separovat lineárně neseparovatelná data transformací ze vstupního prostoru (viz obrázek 2.3 vlevo) do prostoru s vyšší dimenzí, ve kterém jsou data separovatelná (viz obrázek 2.3 vpravo).



Obrázek 2.3: Transformace lineárně neseparovatelných dat

Jádrových funkcí používaných v SVM je několik. Často se používají jádra, která využívají skalární součin a jsou definována předpisem

$$K(x, s_i) = k(x \cdot s_i)$$

Další, méně užívané jádro je například polynomické jádro, definované předpisem

$$K(x, s_i) = (1 + x)^d$$

Proměnná d je zadávána uživatelem. Volba jádra a jeho implementace zásadně ovlivňuje výkonnost SVM algoritmu.

2.7.2 Hidden Markov Model

Skrytý Markovův model je statistická metoda, která modeluje systém se skrytými stavy. Ze systému je tedy pozorovateli viditelný pouze jeho výstup. Vnitřní stav systému, který je pozorovateli skrytý, má na výstup pravděpodobnostní vliv. Matematické základy modelu vyvinul v roce 1966 Leonard E. Baum [8]. Mimo extrakci informace z textu a POS tagging je skrytý Markovův model vhodný také pro rozpoznávání řeči, ručně psaného textu či gest.

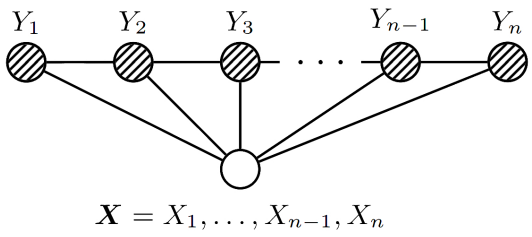
2.7.3 Conditional Random Fields

Metoda podmíněných náhodných polí je opět statistická modelovací metoda. Na poli NLP je oblíbená zejména proto, že na rozdíl od běžných klasifikátorů dokáže vzít v potaz sousední vzorky klasifikovaného vzorku. V případě NER se tedy jedná o kontext, ve kterém je klasifikované slovo zmíněno. Využití této vlastnosti je názorně ukázáno v kapitole 2.3.1. Časté využití CRF je právě v oblasti NLP, dále v oblasti počítačového vidění (segmentace obrazu, rozpoznání objektů) nebo vyhledávání genů.

Na CRF se lze dívat jako na pravděpodobnostní grafický model, který znázorňuje známé závislostní vztahy mezi jednotlivými pozorováními [9], nebo jako na Markovova náhodná pole [10]. Tento model je závislý na náhodné proměnné X, která představuje posloupnost pozorování. Uvažujeme neorientovaný graf

$G = (V, E)$, kde $\{v \in V\}$ jsou vrcholy grafu G a $\{e \in E\}$ jsou jeho hrany.

Každý vrchol v představuje jednu z náhodných proměnných $y_v \in Y$. Pokud každá proměnná y_v zachová Markovovu vlastnost vzhledem ke grafu G, pak (Y, X) je podmíněné náhodné pole. Struktura grafu G může být teoreticky libovolná. V praxi se však při modelování posloupností nejčastěji využívá takové struktury, ve které vrcholy v představující prvky Y, tvoří jednoduchý řetěz prvního řádu. Toto seřazení je ilustrováno na obrázku 2.4 – vyšrafované proměnné jsou generovány modelem, bílé nejsou. Obrázek je převzat z [10].



Obrázek 2.4: Grafické znázornění podmíněných náhodných polí s řetězovou strukturou.

3 Nástroje pro vyhledávání pojmenovaných entit v textu

Nástrojů pro NER existuje celá řada. Níže jsou uvedeny některé z nejznámějších, které jsou dostupné veřejnosti zdarma pro vědecké účely [11]. Tyto nástroje se liší v mnoha ohledech:

- Metody, kterými nástroje pracují – slovníkové, automatické se strojovým učením nebo hybridní, které kombinují slovníkové a statistické metody.
- Třídy entit, které nástroj vyhledává a rozpoznává.
- Doménový rozsah – některé mohou být obecné a aplikovatelné na jakýkoliv text, jiné naopak specificky zaměřené na konkrétní doménu.
- Implementace – některé se používají ve formě knihovny nebo pluginů, jiné jako webové služby atd.
- Výstup – jelikož neexistuje standard pro zadávání pojmenovaných entit, i výstupy nástrojů se liší. Většinou je výstup ve formě objektů nebo textových souborů.

Ačkoliv se mohou nástroje pro NER v mnohém lišit, jedno mají společné – použitý slovník, trénovací sada a dataminingový algoritmus mají zásadní vliv na výkon a efektivitu nástroje.

V následující podkapitole jsou stručně charakterizovány nejpoužívanější nástroje. Rozdělení na cizojazyčné a české nástroje je myšleno z pohledu vstupního textu, se kterým nástroj pracuje. Obecně lze nástroj použít i na jiný jazyk. Jeho úspěšnost pak závisí na míře zaměření nástroje na konkrétní jazyk. Pokud budou jazyky zásadně

odlišné, nástroj pravděpodobně nebude vykazovat takové výsledky jako pro jazyk, na který je uzpůsoben.

3.1 Cizojazyčné nástroje

Stanford NER (SNER)

SNER[1] je nástroj pro rozpoznávání pojmenovaných entit implementovaný v jazyce Java a zaměřený především na rozpoznání tří tříd – osoby, organizace a lokality. V základní verzi obsahuje modely pro anglický jazyk, ale je možné v programu načíst vlastní model a nad ním pak provádět NER. Tento systém využívá metodu CRF, která na rozdíl od běžných klasifikátorů dokáže vzít v potaz širší kontext a nerozhodovat pouze na základě jednoho, v daný moment klasifikovaného vzorku.

Illinois Named Entity Tagger (INET)

INET[2] je vydaván jako samostatný program, který je založen na několika metodách strojového učení: skryté Markovovy modely, vícevrstevné neuronové sítě a jiné statistické metody. V původní verzi rozpoznával čtyři třídy pojmenovaných entit, ve stávající verzi už dokáže rozlišit osmnáct tříd entit. Při práci mimo jiné využívá slovníky sestavené z hesel z wikipedie.

Alias-i LingPipe (LIPI)

LIPI[3] je robustní nástroj, který se používá nejen pro NER, ale i jiné úlohy na zpracování textu a extrakci informace (například automatická oprava překlepů). Pro NER využívá metodu HMM.

OpenCalais Web Service (OCWS)

OCWS[4] je nástroj v podobě webové služby. Metody rozhraní OCWS je možné volat několika protokoly: SOAP, REST a HTTP. Podobně jako u LIPI se jedná o robustní

[1] http://www-nlp.stanford.edu/software/CRF-NER.shtml
[2] http://cogcomp.cs.illinois.edu/page/software_view/NETagger
[3] http://alias-i.com/lingpipe
[4] http://www.opencalais.com/documentation/calais-web-service-api

nástroj s širším zaměřením, který vyhledává nejen entity, ale také například fakta a události. Výsledky pak dokáže mapovat na hesla na wikipedii. OCWS podporuje angličtinu, francouzštinu a španělštinu.

General Architecture for Text Engineering (GATE)

GATE[5] je software, který se zabývá zpracováním textu velmi zeširoka. Na jeho vývoji, který probíhá od roku 1995, pracuje rozsáhlá komunita vývojářů i uživatelů po celém světě. Celá architektura je rozdělena na několik produktů podle zaměření a užití. Jsou jimi:

- GATE Developer – integrované vývojové prostředí určené ke zpracování jazyka a textu spolu se systémem pro extrakci informace a sadou pluginů (tyto pluginy může vyvíjet sám uživatel). Celé toto prostředí je zaměřené na angličtinu.
- GATE Embedded – objektová knihovna určená pro import a použití v jiných aplikacích a programech. Umožňuje přístup ke všem službám GATE Developer.
- GATE Teamware – určeno pro rozsáhlé komerční projekty.
- Mímir (Multi-paradigm Information Management Index and Repository) – nástroj, který umožňuje fulltextové vyhledávání a anotaci dat a je použitelný na rozsáhlé textové korpusy (až TB textu).

3.2 České nástroje

Common Part-of-speech Tagger (COMPOST)

COMPOST[6] je program pro vyhledávání entit v českém a anglickém jazyce, holandštině a islandštině. Je psaný pouze pro operační systém linux a vyvíjený na fakultě matematiky a fyziky Univerzity Karlovy v Praze. Nemá grafické uživatelské rozhraní – spouští se pouze z příkazového řádku. Pojmenované entity vyhledává kombinováním metod strojového učení s učitelem a bez učitele.

[5] https://gate.ac.uk
[6] http://ufal.mff.cuni.cz/compost

MorphoDiTa

MorphoDiTa[7] (Morphological Dictionary and Tagger) je víceúčelový lingvistický nástroj vydaný pod licencí LGPL, který vznikl na Univerzitě Karlově v Praze. Nejedná se přímo o nástroj pro NER, ale umožňuje provádět tokenizaci, morfologickou analýzu a obsahuje také lingvistické modely. Nástroj MorphoDiTa lze použít jak samostatně, tak jako knihovnu pro jazyk Java. V práci jsem jej využil pro získání morfologických značek z textu.

Treex

Treex[8] (formálně TectoMT) je modulární nástroj pro NLP implementovaný v jazyce Python, který také vzniká na Univerzitě Karlově v Praze. Jeho silnou stránkou je vysoká modularizace, díky které je snadné jej začlenit do vlastního projektu a dále rozvíjet. Skládá se z takzvaných *bloků*, které mají jednotné, objektově orientované rozhraní a usnadňují tak vzájemnou interakci. Treex je také možné vyzkoušet přes webové rozhraní.

[7] http://ufal.mff.cuni.cz/morphodita
[8] http://ufal.mff.cuni.cz/treex

Tabulka 3.1: Srovnání vybraných nástrojů pro NER

Nástroj	SA[1]	WS[2]	LIB[3]	Algoritmus	Jazyk[4]	Licence	Implementace	Vývoj[5]
SNER	•	○	•	CRF	EN, DE, ES, ZH	GNU GPL	Java	2006 - 2015
INET	•	○	○	HMM	EN	zdarma pro vědecké účely	Java	2013 - 2015
LIPI	○	○	•	HMM	EN, NL, HI, ES, DE, FR, ZH, AR	zdarma pod AGPL	Java	2003 - 2011
OCWS	○	•	○	—	EN, FR, ES	zdarma pro komerční a nekomerční užití	—	2008 - 2014
GATE	•	○	•	FSA	EN	GNU LGP	Java	2003 - 2014
LIBSVM	○	○	•	SVM	—	zdarma se zachováním licence	C++, Java	2000 - 2014
Apache UIMA	○	○	•	HMM, BSF	EN	apache	C++, Java	2006 - 2015
Lingvo NER	○	○	•	Slovník. metody	CZ	GNU GPL	Java	2010 - 2013

[1]Stand Alone – nástroj lze použít jako samostatně běžící program.
[2]Web Service – nástroj je poskytován jako webová služba.
[3]Library – nástroj je poskytován jako knihovna, kterou lze vložit do vlastního programu.
[4]Myšlen je jazyk, ve kterém nástroj vyhledává entity.
[5]Uvedeny jsou roky vydání první a nejnovější verze nástroje.

4 Návrh a implementace nástroje pro NER

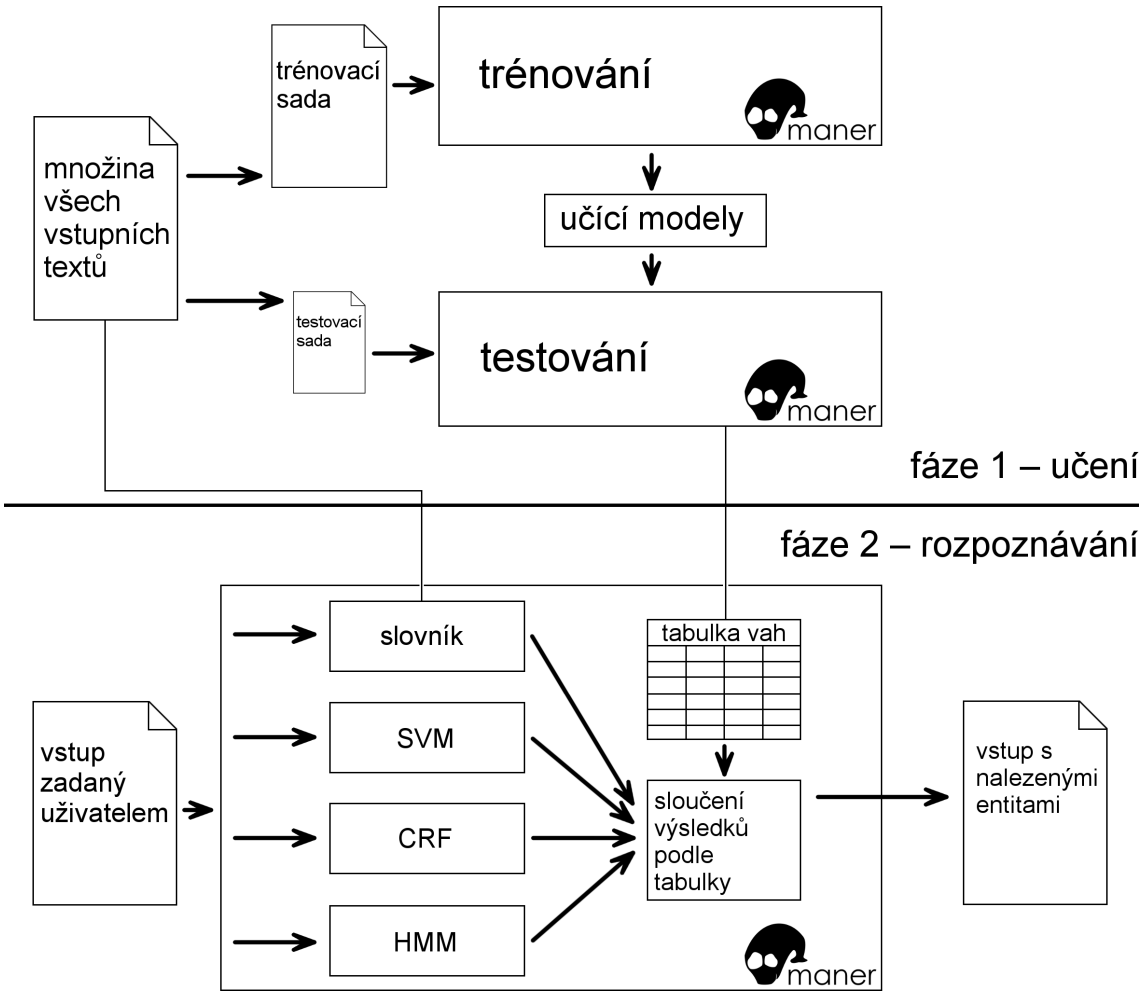
Vlastní implementaci jsem navrhl tak, aby využívala možností více dataminingových algoritmů zároveň a kombinovala jejich silné stránky. Ve vzniklém nástroji jsou použity dataminingové algoritmy SVM, HMM, CRF (viz kapitolu 2.7) a slovníková metoda pro vyhledávání pojmenovaných entit. Z pohledu metodiky práce se tedy jedná o hybridní nástroj. Algoritmy HMM, CRF a SVM jsou prováděny externími nástroji LIPI, SNER a LIBSVM (viz tabulku 3.1). Dále jsem implementoval slovníkovou metodu, která je součástí výsledného programu. Program jsem pracovně nazval MANER (Multiple Algorithm Named Entity Recognizer). Při psaní programu jsem se inspiroval knihou Umění programování [12], která je dobrým zdrojem návodů a příkladů programování složitějších datových struktur a matematických výpočtů.

4.1 Popis navržené implementace

Navržená implementace, znázorněná na obrázku 4.1, pracuje ve dvou základních fázích – fáze učení a fáze rozpoznávání.

4.1.1 Fáze učení

V této fázi je stroji poskytnuta trénovací sada, na které se jednotlivé části nástroje naučí rozpoznávat pojmenované entity. Protože jednotlivé nástroje nepracují se stejnými vstupními daty, je potřeba trénovací data vždy převést do formátu srozumitelného pro daný nástroj. Každý ze tří nástrojů si na základě trénovacích dat vytvoří vlastní model. Pomocí tohoto modelu pak rozpoznává pojmenované entity ve druhé fázi (rozpoznávání). Tuto činnost stačí provést pouze jednou, když je potřeba nástroj naučit rozpoznávat entity nad jinou doménou.



Obrázek 4.1: Návrh implementace nástroje pro NER

4.1.2 Fáze rozpoznávání

V této fázi jsou již nástroje natrénovány a připraveny rozpoznávat pojmenované entity ve vstupním textu. Rozpoznávání probíhá ve dvou krocích:

Krok 1 – vyhledání entit

V tomto kroku všechny čtyři nástroje provedou své vyhledávání (provede se tedy třikrát statistická metoda a jednou slovníková metoda). Výstupem jsou čtyři seznamy pojmenovaných entit.

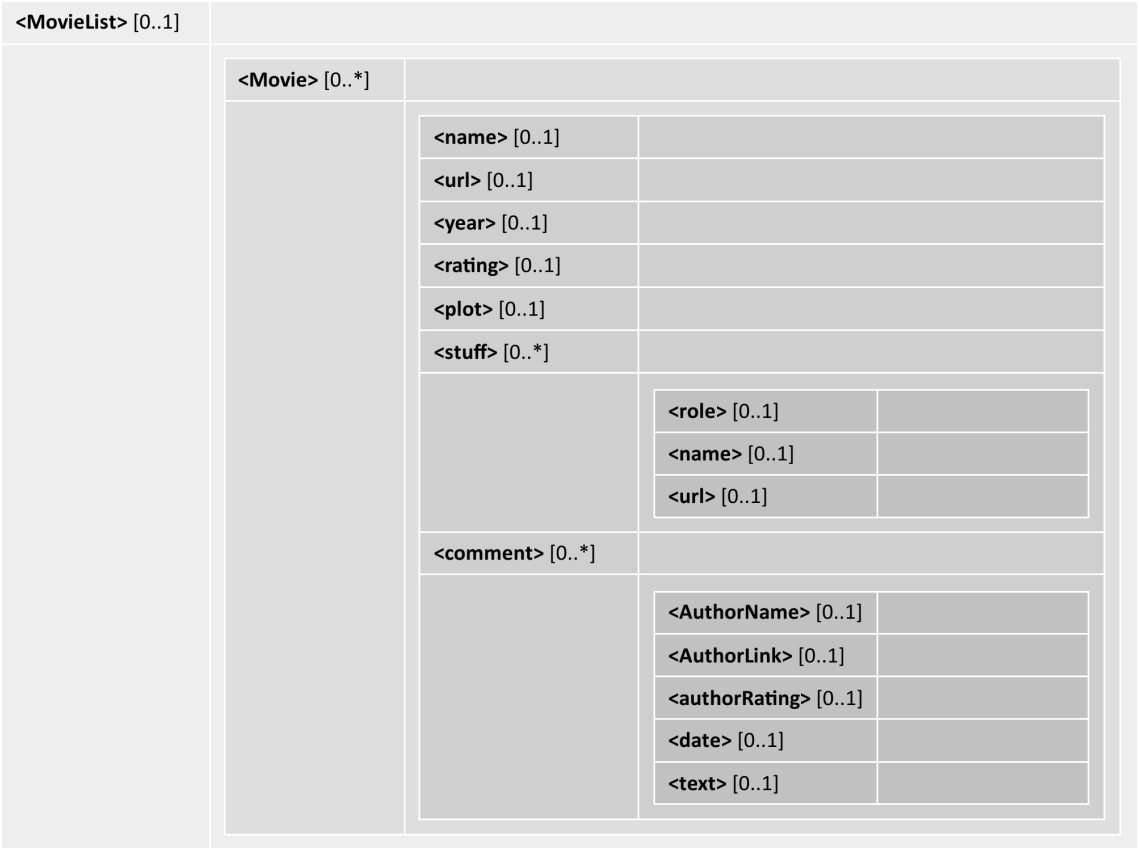
Krok 2 – sloučení výsledků

Čtyři seznamy nalezených entit z předešlého kroku jsou nyní sloučeny do jednoho. Sloučení se řídí předešlým úspěchem jednotlivých nástrojů ve vyhledávání entit. Každému nástroji je podle jeho přesnosti přidělena váha (číslo v intervalu <0; 1>). Na základě těchto vah se určí nejpravděpodobnější kandidáti na správně nalezené pojmenované entity. Vzniklý seznam (ve formě označkovaného vstupního dokumentu s nalezenými pojmenovanými entitami) je výstupem tohoto kroku a celého nástroje.

4.2 Použitý datový korpus

K demonstraci úlohy byl vybrán korpus dat získaný ze stránek ČSFD (Česko-Slovenská Filmová Databáze). Data byla získána postupným parsováním žebříčků filmů, amatérských filmů, TV pořadů a seriálů. Všechny filmy pochází z české produkce. Pro parsování dat jsem v jazyce Java implementoval parser, který využívá knihovnu jsoup[1] k parsování HTML a manipulaci s DOM. Výsledný korpus ve formátu XML obsahuje název filmu, url adresu filmu, rok vzniku, hodnocení, popis zápletky, výpis lidí, kteří se podíleli na vzniku filmu spolu s odkazy na jejich profily, výčet všech komentářů k filmu spolu s uděleným hodnocením, jménem komentátora a odkazem na jeho profil. Datová struktura dokumentu je znázorněna na obrázku 4.2. Datový korpus a zdrojové kódy parseru jsou obsaženy na přiloženém CD. Tabulka 4.1 obsahuje údaje o stažených datech.

[1] http://jsoup.org; intuitivní API pro práci s HTML dokumenty, které využívá podobnou anotaci jako CSS a JS.



Obrázek 4.2: Struktura stažených dat

Tabulka 4.1: Údaje o stažených datech

Počet vět	757 487
Počet filmů	3 571
Počet komentářů	227 893
Počet komentátorů (unikátní)	19850
Počet aktérů (unikátní)	9710

Nástroj MANER lze pro stahování dat o filmech použít následujícím způsobem:

```
java -jar maner.jar -download [yearFrom] [yearTo] [movieType]
```

Tyto tři argumenty slouží k filtrování filmů, o kterých se budou data stahovat. Argumenty *yearFrom* a *yearTo* omezují filmy rokem vzniku. Argument *movieType* omezuje filmy typem.

Povolené hodnoty tohoto argumentu jsou:

0 – filmy

1 – video filmy

2 – TV filmy

3 – TV seriály

4 – TV pořady

7 – studentské filmy

8 – amatérské filmy

Výsledek stahování je uložen do XML souboru s následující cestou:

./files/dumps/list-[movieType]-1-from-[yearFrom]-to-[yearTo].xml

Stahování dat je omezené na deset požadavků za minutu. Předejde se tak vysokému množství požadavků na server, možnému podezření na DoS[2] útok a následnému zablokování IP adresy.

Stažená data jsou dále použita k vytvoření testovací a trénovací sady a slovníku. Metody které jsou k tomu použity čtou data vždy pouze z jednoho souboru, a proto nástroj umožňuje spojit několik XML souborů do jednoho pomocí příkazu:

```
java -jar maner.jar -merge [directoryWithXMLFiles] [outputFile]
```

Data lze z XML formátu do textové podoby bez značek převést příkazem:

```
java -jar maner.jar -convert [inputXMLFile] [outputDirectory]
```

[2]Denial of Service

4.3 Použité značení pojmenovaných entit

Jako standard pro označování pojmenovaných entit ve výstupním textu jsem použil konvence podobné CNEC. Tedy nalezené entity jsou vyznačeny ve formě:

Hrdiny filmu <**PERS** Petera Kerekese> jsou vojenští kuchaři z různých koutů <**LOC** Evropy>.

Značení ve slovníku a v trénovací a testovací sadě je odlišné od značení ve výstupním souboru, a to zejména kvůli odlišné struktuře souborů. Na rozdíl od výstupního souboru, který je ve formě souvislého textu, jsou trénovací a testovací sada a slovník vždy ve formě jednoho hesla na řádek. A proto jsou ve tvaru:

Hrdiny	**O**
filmu	**O**
Petra	**PERS**
Kerekese	**PERS**

Všechny druhy entit, které nástroj MANER vyhledává, jsou spolu s jejich označením shrnuty v tabulce 4.2. Ke klasicky vyhledávaným entitám jsem navíc přidal entity specifické pro vybranou doménu dat. Jedná se o název filmu a přezdívky (zejména přezdívky komentujících, tedy jména uživatelů ČSFD).

Tabulka 4.2: Značení druhů vyhledávaných entit

Druh entity	Označení (SNER, LIPI)	Označení (SVM)
Osoba	PERS	1
Lokalita	LOC	2
Časový údaj	TIME	4
Název organizace	ORG	6
Název filmu	MOV	7
Přezdívka	NICK	3
Slovo, které není entitou	O	0

4.4 Tvorba slovníku z dat z ČSFD

Pro vytvoření slovníku ze stažených dat použijeme příkaz:

```
java -jar maner.jar -dctbuild [inputXMLFile] [outputDictionary]
```

Argument *inputXMLFile* je soubor, který vznikl po vykonání příkazu *download* nebo *merge* (kapitola 4.2). Ve výsledném slovníku jsou obsažena všechna jména filmů, herců, tvůrců filmů a uživatelů, která byla k nalezení ve vstupním souboru. Tato jsou označena příslušným druhem entity a to stylem popsaným v kapitole 4.3.

4.5 Příprava trénovací a testovací sady

Knihovny SNER a LIPI využívají podobný vstupní formát trénovací a testovací sady: [token][tabulátor][třída entity příslušející tokenu]. Liší se pouze v pojmenování druhů entit. Ve formátu pro LIPI je rozlišeno, zda se jedná o začátek (prefix *B-*) pojmenované entity nebo pokračování vícetokenové entity, která začala na některém z předchozích řádků (prefix *I-*). Pro převedení trénovací sady nástroje SNER na trénovací sadu pro nástroj LIPI slouží příkaz:

```
java -jar maner.jar -lipiconvert [pathToSNERTrainFile] [outputFile]
```

Algoritmus provádějící konverzi vychází z předpokladu, že mezi dvěma entitami je alespoň jedno slovo nebo znak, které není entitou. Tedy každou entitu, jež následuje bezprostředně za jinou entitou a má stejnou třídu, označí jako pokračování předchozí entity. Tento přístup je do jisté míry naivní, ale porušení tohoto předpokladu nastane tak ojediněle (například v trénovací a testovací sadě nenastalo), že je možné jej použít, aniž by tím utrpěla následná přesnost vytvořeného učícího modelu. Pro zjednodušení jsem pro trénování SVM algoritmu použil stejný formát jako je formát SNER, který je před spuštěním programu převeden na vektory.

Obě sady byly vytvořeny z dat stažených z ČSFD. Ze všech komentářů a zápletek jsem náhodně vybral tři tisíce vět na trénovací sadu a 600 vět na testovací sadu,

rovnoměrně vždy polovinu vět ze zápletek a polovinu z komentářů. Vybraný text jsem prošel a ke každému tokenu přiřadil odpovídající druh entity. Takto označeným datům se říká *golden data* (zlatá data), neboť se vychází z předpokladu, že jsou správná a stroj se z nich může učit. Tvorba obsáhlejší sady je velmi zdlouhavá činnost a projevuje se to na vysoké hodnotě takto anotovaných dat. Vzniklá trénovací sada je považována za malou (ve srovnání například s trénovací sadou CNEC, která obsahuje 9000 vět).

4.5.1 Automatické rozšiřování trénovací sady

Aby se při rozpoznávání pojmenovaných entit dosáhlo co nejlepších výsledků, využil jsem automatické rozšíření trénovací sady. Protože manuální označování entit je časově náročná činnost, je možné trénovací sadu rozšířit automaticky. Obecně se v dataminingu využívá více přístupů (například generování virtuálních příkladů na základě známých dat). U této konkrétní úlohy je vhodné k rozšíření sady použít vzniklý slovník entit. Trénovací sadu jsem tedy rozšířil o slovník všech jmen a přezdívek stažených z ČSFD.

4.5.2 Trénovací data pro nástroj LIBSVM

Trénovací sada pro nástroj LIBSVM je zásadně odlišná od trénovacích sad dvou předchozích nástrojů. Je to způsobeno především podstatou algoritmu SVM. Před trénováním nástroje je potřeba každý token převést na vektor. Každá položka tohoto vektoru určitým způsobem charakterizuje daný token. Položky vektoru jsem sestavil z morfologických značek ke kterým jsem navíc přidal vlastní charakteristiky popsané níže.

Morfologické značky

Pro charakterizaci tokenu jsem využil morfologické značky. Morfologická značka je řetězec, který vznikne jako výstup morfologické analýzy. Každý jeho znak představuje jednu morfologickou kategorii.

Například značka

Praha: NNFS1-----A----

nese o slově *Praha* informaci, která je popsána v tabulce 4.3.

Tabulka 4.3: Význam morfologických značek pro slovo *Praha*

pozice	znak	význam
1	N	podstatné jméno
2	N	obyčejné substantivum
3	F	ženského rodu
4	S	jednotného čísla
5	1	v prvním pádě
11	A	afirmativ, tedy slovo je bez negativní předpony *ne*

Pokud je na některé pozici pomlčka, znamená to, že tato hodnota u daného slova nedává smysl (například značka na desáté pozici určuje stupeň). Kompletní dokumentaci s popisem každé značky a jejích možných hodnot lze nalézt v [13]. Pro získání morfologických značek k danému tokenu jsem použil nástroj MorphoDiTa.

Struktura vstupních vektorů

Vektory v trénovací sadě mají celkem dvacet položek. První položka představuje třídu entity daného tokenu. Jedná se o číslo v rozsahu nula až sedm. Nula je vyhrazena pro slova, která nejsou pojmenovanými entitami a ostatní čísla představují jednotlivé třídy pojmenovaných entit (viz tabulku 4.2). Dalších patnáct položek vektoru představuje morfologické značky daného tokenu. Položky na pozici 17, 18, 19 a 20 nabývají pouze hodnot jedna a nula, a uchovávají tyto informace:

pozice 17 – zda je daný token delší než dva znaky

pozice 18 – zda daný token začíná velkým písmenem

pozice 19 – zda je daný token celý velkými písmeny

pozice 20 – zda je daný token na začátku věty

Převod trénovací sady do formátu SVM

Pro převod trénovací sady slouží příkaz:

```
java -jar maner.jar -svmconvert [inputFile] [outputFile]
```

Vstupní soubor musí být ve formátu trénovací sady pro nástroj SNER (tedy vždy [token] [třída] na jednom řádku). Výstupní soubor je pak ve formátu jednoho podpůrného vektoru na řádek.

Tedy pro *Praha : NNFS1-----A----* bude výstupem:

2 1:6 2:37 3:1 4:3 5:1 11:1 16:1 17:1 18:0 19:1

Význam jednotlivých čísel je uveden výše. Je důležité podotknout, že první číslo udává třídu pojmenovaných entit ($2 = LOC$) do které vektor náleží a je uvedeno samostatně. Všechna ostatní jsou po dvojici a to ve formě: [pozice položky]:[hodnota položky]. Položky které nejsou určené (v morfologické značce mají pomlčku) se ve vektoru vůbec neobjeví.

4.5.3 Ruční anotace dat

Pro zjednodušení ruční anotace dat nástroj MANER implementuje následující postup:

1. lemmatizace libovolně velkého souboru s textem – data jsou převedena do podoby jedna věta na řádek,
2. náhodný výběr zadaného počtu vět,
3. tokenizace – data jsou převedena do podoby jeden token na řádek,
4. přiřazení výchozí třídy entity každému tokenu – data jsou převedena do podoby [token][tabulátor][výchozí třída entity].

Pro provedení dané sekvence operací slouží příkazy:

```
java -jar maner.jar -lemmatize [inputFile]
java -jar maner.jar -limitlemmas [inputFile] [n]
java -jar maner.jar -tokenize [inputFile]
java -jar maner.jar -defaultclass [inputFile] [defaultClass]
```

Každý příkaz vytváří nový soubor (se stejnou cestou jako vstupní soubor, pouze mu přidá novou koncovku), který přejímá následující příkaz. Po vykonání této sekvence je potřeba projít celý výsledný soubor a každé entitě přiřadit její odpovídající třídu. Výstupem tohoto kroku jsou již výše zmíněná *golden data*. V tabulce 4.4 jsou uvedeny údaje o ručně označených datech. Je patrné a očekávatelné, že většina slov nejsou pojmenované entity. Poměr rozdělení jednotlivých tříd je na náhodně vybraných datech velmi podobný v trénovací i testovací sadě. V tabulce je také uvedeno, jak se změnila trénovací sada po automatickém rozšíření.

Tabulka 4.4: Údaje o trénovací a testovací sadě

	Trénovací sada	Testovací sada	Rozšířená tr. sada
Počet vět	3 000	600	12 000
Počet slov	49 768	9 861	208 962
Počet jmen (PERS)	1 904	340	22 134
Počet lokalit (LOC)	519	66	519
Počet názvů filmů (MOV)	340	60	10 481
Počet čas. údajů (TIME)	148	31	14 432
Počet organizací (ORG)	136	38	136
Počet přezdívek (NICK)	59	10	28 695
Počet ne-entit (O)	46 609	9 304	132 647
Procento entit	6,35 %	5,65 %	36,5 %

4.6 Trénování NER nástrojů

Po vytvoření trénovacích sad je možné je použít k trénování jednotlivých nástrojů. Příkazy pro trénování jsou:

```
java -jar maner.jar -svmtrain [inputFile] [outputFile]
java -jar maner.jar -lipitrain [inputFile] [outputFile]
java -jar maner.jar -snertrain [propertiesFile]
```

U nástrojů LIBSVM a LIPI je rozhraní stejné. Argument *inputFile* je cesta ke vstupní trénovací sadě v příslušném formátu a argument *outputFile* je cesta, kam se uloží natrénovaný model. U trénování nástroje SNER je rozhraní odlišné. Jediný argument *propertiesFile* je cesta k souboru s nastavením trénovacího nástroje. Tento soubor mimo jiné obsahuje i nastavení vstupního a výstupního souboru. Příklad takového souboru lze nalézt na přiloženém CD. Cesta k souboru je:

./MANER/files/SNER/csfdner.prop.

Trénovací sada pro nástroj SVMLIB musí být nejprve převedena do formátu vektorů (viz kapitolu 4.5.2). Co se týče doby trénování nástrojů, jsou zde zásadní rozdíly. Nástroj LIBSVM se na trénovací sadě učil nejkratší dobu (necelých 7 sekund), nástroj SNER se učil 27 sekund a nástroji SNER vytváření modelu zabralo dokonce 9,5 minuty. Uvedené hodnoty byly naměřeny na počítači s konfigurací uvedenou v tabulce 4.5. Na stejném počítači byly prováděny i veškerá další testování.

Tabulka 4.5: Konfigurace počítače, na kterém bylo prováděno testování

Operační systém	Windows 8.1 Pro
Velikost RAM	6 GB
Procesor	Intel® Core™ i3-3227U, CPU @ 1,90 GHz 1,90 GHz
Typ systému	64bitový

V tabulce 4.6 jsou uvedeny doby trénování jednotlivých nástrojů na původní trénovací sadě i na rozšířené trénovací sadě. Časy jsou uvedeny ve formátu hh:mm:ss.

Tabulka 4.6: Doba trvání trénování jednotlivých nástrojů

	Základní trénovací sada	Rozšířená trénovací sada
SNER	00:09:11	01:29:42
LIPI	00:00:30	00:01:21
LIBSVM	00:01:02	02:35:44

4.7 Testování NER nástrojů

Součástí nástroje je také rozhraní pro jeho otestování. Pro otestování je potřeba sada dat ve stejném formátu jako trénovací data. Aby byly výsledky testování odpovídající, měla by se data v testovací sadě lišit, ale také by se mělo jednat o data ze stejné domény jako data v trénovací sadě. Otestování se provede příkazem:

```
java -jar maner.jar -test [inputFile] [snerModelFile]
[lipiModelFile] [svmModelFile] [outputFile]
```

Nástroj MANER načte ze vstupního souboru správné třídy jednotlivých entit a předá nástrojům pro rozpoznávání entit pouze čistá data. Po zpracování vyhodnotí výsledky všech nástrojů pomocí správných odpovědí a výsledky uloží do výstupního souboru. Tyto výsledky obsahují počty TP, FP, FN, přesnost, úplnost a F-míru.

4.8 Použití NER nástrojů samostatně

Mimo to, že lze nástroj MANER použít jako celek, lze také použít pouze jednotlivé nástroje. Můžeme tak zjistit, jak hodnotil daný vstupní text každý nástroj. Rozhraní pro samostatné použití nástrojů je:

```
java -jar maner.jar -svm [inputFile] [modelFile]
java -jar maner.jar -lipi [inputFile] [modelFile]
java -jar maner.jar -sner [inputFile] [modelFile]
```

Argument *inputFile* je cesta ke vstupnímu souboru s prostým textem. Argument *modelFile* je cesta k modelu, jehož tvorbu popisuje kapitola 4.6. Výstup jednot-

livých nástrojů je vytištěn na standardní výstup. Pokud bychom chtěli zkontrolovat vyhodnocení slovníkové metody, použijeme příkaz:

```
java -jar maner.jar -dictionary [inputFile] [dictionaryFile]
```

Argument *dictionaryFile* je soubor se slovníkem. Tvorba slovníku je popsána v kapitole 4.4.

5 Testování navrženého nástroje

V této kapitole jsou shrnuty výsledky naměřené při běhu programu. Program byl natrénován a otestován pomocí testovací a trénovací sady vytvořené v rámci práce – tedy pomocí dat stažených z webu ČSFD. V tabulce 5.1 jsou shrnuty výsledky testu programu před automatickým rozšířením trénovací sady pomocí slovníku. Uvedená hodnota vždy představuje F-míru (tedy kombinaci přesnosti a úplnosti) konkrétního nástroje. F-míra je v tomto případě harmonický průměr přesnosti a úplnosti (viz kapitolu 2.4). Trénovací a testovací sady použité pro otestování nástroje jsou popsány v tabulce 4.4.

Tabulka 5.1: Výsledky testování nástroje MANER před automatickým rozšířením trénovací sady

Třída	Testovaný nástroj				
	SNER	LIPI	LIBSVM	slovník	MANER
PERS	0,801	0,131	0,721	0,885	**0,952**
LOC	0,666	0,4	0,0	0,0	**0,652**
TIME	0,415	0,586	0,384	0,0	**0,578**
ORG	0,211	0,211	0,0	0,0	**0,4**
MOV	0,052	0,027	0,0	0,874	**0,787**
NICK	0,0	0,011	0,0	0,71	**0,551**
O	0,917	0,888	0,922	0,933	**0,952**

Z tabulky 5.1 je patrné, že nejlépe nástroj vyhledával jména lidí a datové údaje. Jména lidí vyhledával nejlépe proto, že jejich zastoupení v trénovací sadě bylo nejvyšší. Datových údajů nebylo mnoho, ale od běžného textu jsou jednoduše rozlišitelné. Nejlépe toho využívá algoritmus SVM, který na slova pohlíží z morfologického hlediska. Naopak ostatní třídy pro tento algoritmus byly problémové právě z tohoto důvodu.

Dále je patrné, že slovníková metoda nepřesahuje rámec definovaného slovníku, a rozpoznává pouze druhy pojmenovaných entit, které jsou ve slovníku definovány. Obecně si ze všech nástrojů nejlépe vedl nástroj SNER. Lze zde vypozorovat jistou souvislost mezi dobou trénování nástroje a jeho úspěšností, viz tabulku 4.6.

Před tím, než se provedlo testování celého nástroje MANER, byly výsledky testování dílčích nástrojů použity k sestavení tabulky vah. Jako váha nástroje pro konkrétní třídu entit slouží jeho přesnost při testování. V podstatě se tedy jedná o tabulku přesností jednotlivých nástrojů. Jedinou výjimkou je třída *O*. Ačkoliv je zde označována jako třída pojmenované entity, ve skutečnosti se o pojmenovanou entitu nejedná. Protože je její zastoupení v běžném textu nejvyšší, nástroj je touto třídou přeučen. Pokud by jí zůstala její přesnost, pak by většinu skutečných pojmenovaných entit v textu nástroj označil třídou *O*. Proto byla v tabulce vah u každého nástroje pro třídu *O* použita přesnost 0,01. Váhy, která vznikly při prvním testování, jsou uvedeny v tabulce 5.2.

Tabulka 5.2: Váhy použité při testování nástroje MANER před automatickým rozšířením trénovací sady

Třída	Nástroj			
	SNER	LIPI	LIBSVM	slovník
PERS	0,905	0,899	0,781	0,97
LOC	0,919	0,888	0,0	0,0
TIME	1,0	1,0	1,0	0,0
ORG	1,0	1,0	0,0	0,0
MOV	0,5	1,0	0,0	0,7
NICK	0,25	0,333	0,0	0,125
O	0,01	0,01	0,01	0,01

V tabulce 5.3 jsou výsledky běhu programu po automatickém rozšíření trénovací sady. Je patrné, že toto rozšíření na výsledky testování mělo značný vliv. Bylo zaznamenáno očekávané zlepšení zejména v rozpoznávání těch tříd entit, které byly rozšířené. Ve třídách, které rozšířeny nebyly (*LOC*, *ORG*) výsledky nejsou jednoznačné. Zatímco rozpoznávání třídy *LOC* se zlepšilo, rozpoznávání třídy *ORG* se zhoršilo.

Tabulka 5.3: Výsledky testování nástroje MANER po automatickém rozšíření trénovací sady

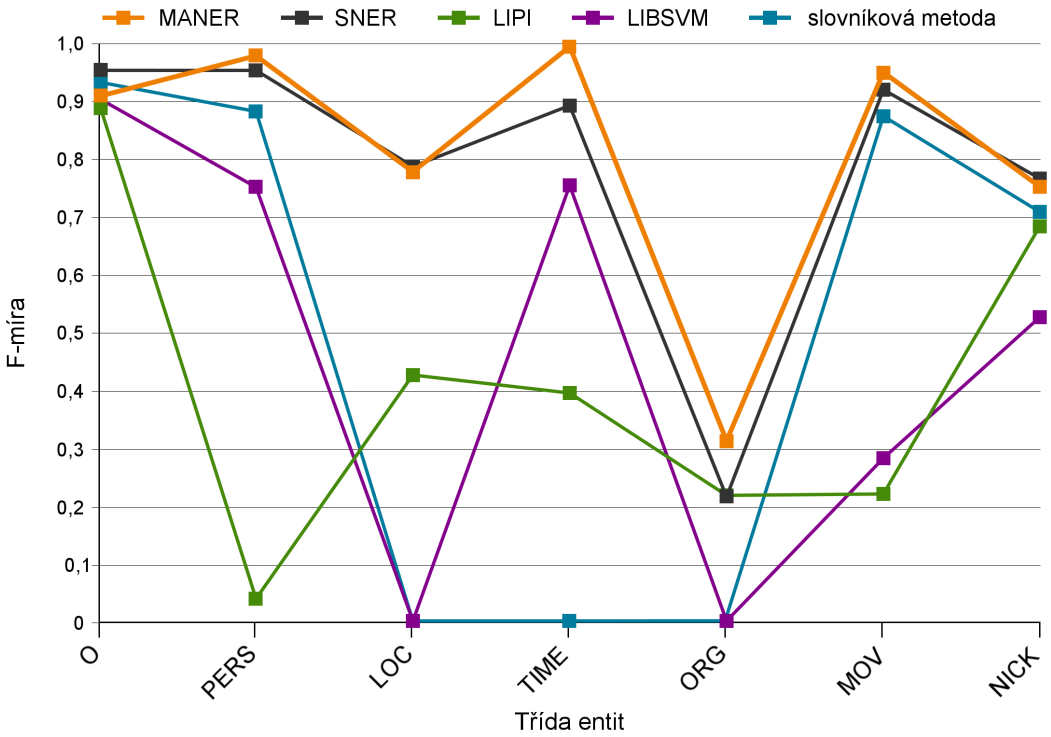
Třída	Testovaný nástroj				
	SNER	LIPI	LIBSVM	slovník	MANER
PERS	0,95	0,043	0,753	0,885	**0,98**
LOC	0,789	0,429	0,0	0,0	**0,769**
TIME	0,894	0,398	0,753	0,0	**0,996**
ORG	0,222	0,222	0,0	0,0	**0,316**
MOV	0,922	0,227	0,286	0,875	**0,952**
NICK	0,768	0,687	0,529	0,71	**0,753**
O	0,955	0,89	0,905	0,933	**0,911**

Váhy použité pro testování s rozšířenou trénovací sadou jsou uvedeny v tabulce 5.4.

Tabulka 5.4: Váhy použité při testování nástroje MANER po automatickém rozšířením trénovací sady

Třída	Testovaný nástroj			
	SNER	LIPI	LIBSVM	slovník
PERS	0,709	0,6	0,601	0,992
LOC	0,625	0,666	0,0	0,0
TIME	0,935	0,93	0,826	0,0
ORG	0,666	0,666	0,0	0,0
MOV	0,727	0,8	0,0	0,933
NICK	0,0	1	0,0	0,944
O	0,01	0,01	0,01	0,01

Graf na obrázku 5.1 přehledně znázorňuje, jak byl nástroj MANER úspěšný v nalézání pojmenovaných entit v porovnání s dílčími výsledky jednotlivých nástrojů.



Obrázek 5.1: Porovnání F-míry nástroje MANER s jednotlivými nástroji

Součástí vývoje byla také následná paralelizace běhu nástroje. Protože MANER původně pracoval v jednom vlákně, bylo možné změřit vliv paralelizace na rychlost běhu programu. Naměřené hodnoty jsou uvedeny v tabulce 5.5. Měření probíhalo na počítači se stejnou konfigurací, jako je uvedena v tabulce 4.5. Do měření byl zahrnut čas běhu programu od začátku do konce, včetně konstantní složky (inicializace nástrojů a načítání modelů). Časy jsou uvedeny ve formátu hh:mm:ss.

Tabulka 5.5: Vliv paralelizace na rychlost běhu programu

	2 kB textu (379 slov)	2,6 MB textu (370 000 slov)
Před paralelizací	00:00:32	02:13:42
Po paralelizaci	00:00:14	00:53:21

6 Závěr

V této práci byla řešena problematika vyhledávání pojmenovaných entit pomocí algoritmů dataminingu. Byla provedena rešerše nástrojů, které se k daným účelům používají, jejich výhody, nevýhody a specifika použití. Dále byla provedena rešerše algoritmů dataminingu, které se v uvedené oblasti extrakce informace používají. Na základě těchto poznatků byl navržen a v jazyce Java implementován nástroj, který inovativním způsobem řeší daný problém. Tento nástroj kombinuje několik existujících nástrojů a algoritmů a využívá výhody každého z nich. Pomocí navrženého nástroje lze docílit vyšší přesnosti i úplnosti než u jednotlivých nástrojů samostatně.

Práce dále na praktickém příkladě popisuje, jak pro navržený nástroj vytvořit datový model. Na vytvořeném modelu byl nástroj otestován pomocí přesnosti a úplnosti. Nástroj téměř ve všech případech vykazoval vyšší přesnost i úplnost než při použití dílčích nástrojů a algoritmů samostatně, což lze považovat za hlavní přínos práce. Po otestování nástroje pomocí přesnosti a úplnosti byly naplněny všechny body zadání práce.

Další vývoj nástroje MANER by mohl směřovat k přívětivějšímu uživatelskému rozhraní. Jelikož je uživatelské rozhraní programu ve formě příkazové řádky, mohl by být dalším krokem vývoj grafického uživatelského rozhraní, které by program zpřístupnilo širšímu spektru uživatelů. Lepších výsledků vyhledávání by také bylo možné docílit dalším cíleným rozšiřováním trénovací sady. A to zejména o ty třídy entit, které jsou v té stávající řídce zastoupené.

Literatura

[1] TURNER, Vernon, David REINSEL, John F. GANTZ a Stephen MINTON. The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things. *The Digital Universe of Opportunities: Rich Data and the Increasing Value of the Internet of Things* [online]. 2014 [cit. 2015-01-23]. Dostupné z: `http://idcdocserv.com/1678`.

[2] MAYER-SCHÖNBERGER, Viktor a Kenneth CUKIER. *Big Data*. 1. vyd. Brno: Computer Press, 2014, 256 s. ISBN 978-80-251-4119-9.

[3] RUD, Olivia Parr. *Data mining: praktický průvodce dolováním dat pro efektivní prodej, cílený marketing a podporu zákazníků (CRM)*. 1. vyd. Praha: Computer Press, 2001, 329 s. Rychle a jistě. ISBN 8072265776.

[4] RATINOV, Lev a Dan ROTH. *Design challenges and misconceptions in named entity recognition*. In: Proceedings of the Thirteenth Conference on Computational Natural Language Learning - CoNLL '09 [online]. 2009 [cit. 2015-02-20]. DOI: 10.3115/1596374.1596399.

[5] GRISHMAN, Ralph a Beth SUNDHEIM. *Message Understanding Conference-6*. In: *Proceedings of the 16th conference on Computational linguistics* - [online]. 1996 [cit. 2015-05-15]. DOI: 10.3115/992628.992709.

[6] GOUTTE, Cyril a Eric GAUSSIER. *A Probabilistic Interpretation of Precision, Recall and F -score, with Implication for Evaluation* [online]. Meylan, France, 2004 [cit. 2015-01-13]. Dostupné z: `http://www.xrce.xerox.com/content/download/16594/118473/file/xrce_eval.pdf`. Xerox Research Centre Europe.

[7] WU, Xindong, Vipin KUMAR, J. Ross QUINLAN, Joydeep GHOSH, Qiang YANG, Hiroshi MOTODA, Geoffrey J. MCLACHLAN, Angus NG, Bing LIU, Philip S. YU, Zhi-Hua ZHOU, Michael STEINBACH, David J. HAND a Dan STEINBERG. *Top 10 algorithms in data mining. Knowledge and Information Systems* [online]. 2007, vol. 14, issue 1, s. 1-37 [cit. 2015-05-15]. DOI: 10.1007/s10115-007-0114-2.

[8] BAUM, Leonard E. a Ted PETRIE. *Statistical Inference for Probabilistic Functions of Finite State Markov Chains*. The Annals of Mathematical Statistics [online]. 1966, vol. 37, issue 6, s. 1554-1563 [cit. 2014-12-11]. DOI: 10.1214/aoms/1177699147.

[9] KOLLER, Daphne a Nir FRIEDMAN. *Probabilistic graphical models: principles and techniques*. Cambridge: MIT Press, c2009, xxxv, 1231 s. Adaptive computation and machine learning (MIT Press). [cit. 2015-12-11]. ISBN 978-0-262-01319-2.

[10] WALLACH, Hanna M. 2004. *Conditional Random Fields: An Introduction* [online]. [cit. 2015-12-11]. Dostupné z: `http://www.inference.phy.cam.ac.uk/hmw26/papers/crf_intro.pdf`.

[11] ATDAG, Samet a Vincent LABATUT. A comparison of named entity recognition tools applied to biographical texts. In: 2nd International Conference on Systems and Computer Science [online]. 2013 [cit. 2015-01-07]. DOI: 10.1109/icconscs.2013.6632052.

[12] KNUTH, Donald Ervin. *Umění programování*. Vyd. 1. Brno: Computer Press, 2008, xix, 648 s. ISBN 978-80-251-2025-5.

[13] HAJIČ, Jan. *Popis morfologických značek — poziční systém.* [online]. s. 6 [cit. 2015-02-16]. Dostupné z: `https://ucnk.ff.cuni.cz/doc/popis_znacek.pdf`.