VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Návrh modulu systému pro detekci plagiátů

Design of Module for Plagiarism detection

2012

Sergey Kostin

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Sergey Kostin**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Návrh modulu systému pro detekci plagiátů
Design of Module for Plagiarism Detection

Zásady pro vypracování:

Pro zpracování textu a detekci plagiátů v současné době existují různé metody a algoritmy. V této práci se diplomant v teoretické části zaměří na přehled existujících přístupů k detekci plagiátů. V praktické části provede aktualizaci uživatelského rozhraní software Amphora v prostředí .NET Framework a navrhne rozhraní mezi GUI a vyhledávacím engine Amphora. Dále pomocí modulů dodaných vedoucím práce vytvoří vyhledávací engine pro otestování funkčnosti vytvořeného rozhraní. Vše bude směrováno od klasického systému pro vyhledávání v dokumentech na systém vyhledávání plagiátů. S tím souvisí i řada změn jak v použitých rozhraních, tak v samotné GUI.

Jednotlivé body práce jsou:
1. Seznámení se s problematikou vyhledávání plagiovaných dokumentů a části dokumentů.
2. Analýza existujících přístupů k detekci plagiátů.
3. Návrh a implementace GUI pro vyhledávání plagiátů.
4. Implementace a optimalizace vyhledávacího engine pro detekci plagiátů.
5. Výběr vhodné metody pro zhodnocení kvality vyhledávání plagiátu (statistické vyhodnocení, vizuální vyhodnocení).
6. Experimenty a jejich zhodnocení.

Seznam doporučené odborné literatury:

J. Malcolm, P. C. R. Lane, Compare A Journal Of Comparative Education , 1-6, 2008.
Alzahrani, S., Salim, N. & Abraham, A.: Understanding Plagiarism Linguistic Patterns , Textual Features and Detection Method, 2011.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Jan Martinovič, Ph.D.**

Datum zadání: 18.11.2011

Datum odevzdání: 04.05.2012

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.*

V Ostravě 15. červenec 2012 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 15. červenec 2012 .....

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 15. červenec 2012 .....

Rád bych tímto poděkoval Ing. Janu Martinovičovi, Ph.D. za odborné vedení mé diplomové práce a také za podnětné nápady a tvůrčí invenci, se kterou se mi laskavě věnoval.

Abstrakt

Diplomová práce pojednává o plagiátorství jako pojmu, je zde rozebrána teoretická otázka o plagiátech a jejich typech, dále se pak zabývá metodami pro vyhledávání plagiovaných prací, dokumentů, článků a podobně. Jsou zde podrobně rozebrány současné možnosti zpracování datových kolekcí pro možnosti porovnávání, tedy takové metody, které vhodně rozdělují jednotlivé texty na menší části spolu s odstraněním zavádějících informací. Jsou probrány jednotlivé modely a metody pro vyhledávání a v neposlední řadě práce popisuje také současná hotová řešení pro detekci plagiátů. Cílem této práce pak bylo navrhnout a zrenovovat rozhraní vyhledávacího systému Amphora, dále pak vytvořit webovou aplikaci pro vizualizaci vyhledávání, která by vhodně reprezentovala nalezené podobnosti v jednotlivých textech. Nakonec byly provedeny experimenty, které názorně ukazují a zhodnocují dosažené výsledky.

Klíčová slova: Detekce plagiátorství, plagiáty, vyhledávání, před-zpracovávání dokumentů, indexace

Abstract

This thesis is concerned with the plagiarism as a term, thesis also deals with the theoretical questions about the plagiarism and its types, then it is describing the methods used for searching plagiarized work, documents, articles etc. Current possibilities of processing huge data collections for ability of comparing texts are described in detail; these are methods which are correctly separating text to small parts together with the unimportant information disposal. Each of the models and methods for searching is discussed and this work is also describing present ready-made solutions for plagiarism detection. The aim of this work was to design and renew the interface of the searching engine - Amphora, and then to implement a web application for visualization of the search results which could properly represent found similarities in each text files. In the end there have been done few experiments which can illustrate and evaluate reached results.

Keywords: Plagiarism detection, plagiarism, data searching, pre-processing documents, indexation

Seznam použitých zkratek a symbolů

API	–	Application Programming Interface (rozhraní pro programování aplikací)
ASP .NET	–	Web Application Framework
CSSM	–	Common Semantic Sequence Model
ČSN	–	Česká technická norma
ČTK	–	Česká Tisková Kancelář
EG	–	Exponentiated Gradient
EPP	–	Evolutionary Plagiarism Probability
FN	–	False negative (falešně negativní)
FP	–	False pozitive (falešně pozitivní)
GUI	–	Graphical User Interface (grafické uživatelské rozhraní)
HFV	–	Heavy Frequency Vector
IDF	–	Inverse Document Frequency (převrácená četnost slova ve všech dokumentech)
ILR	–	Interagency Language Roundtable
IR	–	Information Retrieval
ISO	–	International Organization for Standardization (mezinárodní organizace pro normalizaci)
LSA	–	Latent Semantic Analysis (latentní sémantická analýza)
LSH	–	Locality Sensitive Hashing
MDI	–	Multiple Document Interface
PPV	–	Positive Predictive Value
SCAM	–	Standard Copy Analysis Mechanism
SOM	–	Self Organizing Maps (Samo-organizačí mapy)
SVD	–	Singular Value Decomposition
SVM	–	Support Vector Machines
TF	–	Term Frequency (četnost slova v dokumentu)
TN	–	True negative (skutečně negativní)
TP	–	True pozitive (skutečně pozitivní)

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Důvodů, proč vzniká nový projekt pro detekci plagiátů, je hned několik. Jelikož se efektivní způsob detekce plagiátů potýká s velkými požadavky na výpočetní složitost, a to hlavně díky obrovskému a stále rostoucímu rozsahu dat, je potřeba navrhnout metodu pro detekci takovou, která bude dostatečně rychlá, přesná, komplexní (odhalí dostatečné množství plagiátů) a umožní realizaci inteligentního rozhraní pro podporu rozhodování.

Ano, v současné době existuje již velké množství hotových systému pro detekci plagiátů, většina těchto řešení má ale své nevýhody. Klíčový problém existujících řešení je bezesporu jejich rychlost s jakou jsou schopny detekovat podezření z plagiátorství. Samozřejmě výsledné posouzení zda se opravdu jedná o plagiát je již vždy na člověku, proto je tento časový faktor označen jako velmi důležitý. Dalším problémem současných aplikací se může zdát jejich špatná míra vizualizace při zobrazování výsledku hledání plagiátů a vazeb mezi plagiátem a originálem.

Cílem této práce bylo navrhnout hned několik zásadních věcí, které by spojily již hotové metody pro vyhledávání informací a vytvořily tak mnohem dokonalejší metodu pro detekci plagiovaných dokumentů a prací, a to právě na základě jednotlivých hotových modelů pro inteligentní vyhledávání, jakými jsou například vektorový model a existující systém pro vyhledávání informací (již dříve realizovaný projekt) Amphora.

Dalším dílčím úkolem bylo navrhnout grafické uživatelské rozhraní, které by vhodným způsobem pomáhalo řídit proces vyhledávání plagiátů a přehlednou vizualizací umožňovalo uživateli rozhodnout o pravosti daného výsledku.

V neposlední řadě práce také popisuje velmi důležitou složku takového vyhledávání, a to co možná nejrychlejší procházení velmi rozsáhlých kolekcí dat, dokumentů a prací podobného druhu. Obsahem této diplomové práce je také obecná teorie o plagiátorství, rozlišných hotových metodách pro vyhledávání, indexaci a strukturování dat, a také o metodách pro detekci plagiátů. Poté co bude nastíněna teorie, následuje návrh vlastního řešení a řešení jednotlivých dílčích úkolů.

1.1 Struktura práce

Kapitola 3 se zabývá analýzou existujících přístupů pro detekci plagiátů a vyhodnocování kvality plagiátorských systémů.

Metody před-zpracovaní vstupního textu pro indexovaní jsou popsány v sekci 4.1. Mezi ně patří takové metody, jako jsou tokenizace (odstavec 4.1.1), eliminace stop-slov (odstavec 4.1.2), lemmatizace (odstavec 4.1.3) a normalizace (odstavec 4.1.4). Dále v sekci 4.2 budou krátce nastíněny modely vyhledávaní.

V kapitole 5 je provedena analýza hotových systémů pro odhalení plagiátů. Praktická část diplomové práce je uvedena v kapitole 6, ve které je detailně popsána architektura navrhnutého systému (sekce 6.1), způsob, jakým se zpracovává vstupní kolekce dat (odstavec 6.1.1). V neposlední řadě jsou v dané kapitole navrženy způsoby možné vizualizace a reprezentace získaných výsledku.

Testovací korpus, nad kterým byly provedeny experimenty, je popsán v sekci 7.1. Postup experimentu a jejich výsledky jsou poté popsaný v sekci 7.3 a v sekci 7.4. Realizovaná část práce, námět na další rozšíření a optimalizaci jsou zahrnuty v závěrečné kapitole 8.

2 Úvod do problematiky plagiátorství

Tato kapitola se zabývá pojmem a problematikou plagiátorství. Pro další využití je potřeba vyspecifikovat či definovat pojmy plagiátu a plagiátorství, a to hlavně také z hlediska využití právě pro automatickou detekci. V neposlední řadě si řekneme také něco o historii a obecných pojmech plagiátorství.

2.1 Plagiátorství jako pojem včetně souvislostí

Není jednoduché přesněji vymezit či striktně vydefinovat pojem plagiátorství, jelikož takových druhů existuje dlouhá řada a v každém oboru se od sebe ještě většinou citelně odlišují. Jednu základní vlastnost však mají všechny obory a typy společnou, a to v případě, že kdokoli, kdo vydává cizí myšlenku či celé dílo za vlastní, dopouští se bezesporu plagiátorství jako takového.

S jistotou však můžeme říct, že plagiátorství a jeho různé podoby se většinou překrývají nebo splývají. Můžeme vyčíst některé formy plagiátorství:

- Zkopírování celého obsahu či části práce z cizího zdroje, aniž by tento zdroj byl označen.
- Doslovně zkopírovat část cizího textu a neuvést jej jako citaci se zdrojem.
- Vydávat práci jiného studenta za svou, a to bez jeho vědomí.
- Nechat si napsat práci někým jiným (či koupit) a vydávat za vlastní.
- Překlad jiné práce z cizího jazyka a vydávat za vlastní (bez uvedení zdroje).
- Uvést kolektivní práci za svou bez uvedení zbylých účastníků.

Různé formy těchto plagiátů mohou mít také různý dopad pro daného ″autora″. V praxi ale musíme rozlišovat plagiátorství, kdy si je autor plně vědom, že se jedná o podvod, anebo naopak zda jde pouze o neznalost správné práce s uvedením zdrojů apod. Z těchto a mnoha dalších důvodů je potřeba si uvědomit, že ani automatizovaný nástroj, byť sebedokonalejší (alespoň zatím) není schopen plně odhalit veškeré výše uvedené prohřešky. Jak již bylo řečeno, pojem plagiátorství se týká různých oborů. V této diplomové práci se ale zaměříme zejména na práce textové.

Pomyslná hranice mezi plagiátorstvím a výzkumem je překvapivě zahalena v šeru. Nakonec přiznejme si, pokročilý výzkum je vždy možný pouze s asistencí ostatních. V různých oborech (např. literatura či právo) školní práce často obsahují rozlišné domněnky následované stovkami citací z různých zdrojů pro ověření či zkreslení dané práce. V těchto případech je nemožné klasifikovat cokoli jako originál či plagiát pouze na základě počtu řádků, které jsou doslova vytržené z ostatních zdrojů. V dalších oborech, jakým je například matematika, může být nezbytné citovat standardní literaturu, aby se autor ujistil, že čtenáři mají dostatečné informace k pochopení dané problematiky, například důkazu nového výsledku, který může přesáhnout i třetinu práce. V jiných disciplínách

strojírenství či počítačové vědě je reálná hodnota příspěvku většinou v podobě vyvíjeného zařízení či algoritmu (což ani nemusí být přesně v článku zahrnuto), a tedy popis toho v čem je důležité dané zařízení či algoritmus se většinou dočteme až z řady jiných knih apod. Stručně řečeno, je velmi těžké určit univerzální definici plagiátu, a to ani textového, což tuto problematiku značně komplikuje [10].

2.2 Formy plagiátorství

Sledování chování plagiátorství v praxi odhaluje nespočet běžně používaných metod pro nelegální využívání cizích textů, tyto metody jsou popsány níže [10]:

- copy & paste (tedy slovo od slova) - specifikuje vytržení textu od cizího autora;
- změna stylu plagiátu;
- ideové plagiátorství.

Maskovací plagiátorství zahrnuje postupy určené k zahalování zkopírovaných částí. Byly označeny 4 různé maskovací techniky:

- Shake & paste - jedná se o kopírování sloučením věty nebo odstavců z různých zdrojů s mírnou úpravou nezbytnou pro vytvoření srozumitelného textu.
- Rozsáhlé plagiátorství - označuje vložení dalšího textu do zkopírovaných pasáží.
- Sumarizační plagiátorství - popisuje souhrn nebo výtah kopírovaného materiálu.
- Mozaika - tento druh plagiátorství je popsán jako sloučení textových segmentů z různých zdrojů spolu se zamlžením způsobem záměny pořadí slov, nahrazení synonymy či přidáváním / odebíráním některých slov.

Tématem této diplomové práce jsou zejména první typ plagiátorství a jeho verze s použitím maskovacích technik.

2.3 Plagiát

Pojem plagiát bývá definován různě. Norma ČSN ISO 5127-2003 uvádí, že plagiátem je ″představení duševního díla jiného autora propůjčeného nebo napodobeného v celku nebo z části, jako svého vlastního″[20].

3 Analýza existujících přístupů k detekci plagiátů

V dnešní době navrženo velké množství přístupů k detekci plagiátu, v této kapitole jsou čtenáři představen obecný přehled o některých existujících metodách.

- *Metoda n-gramů*
  Souvislá posloupnost n prvku dané sekvence textu. N-gram muže obsahovat libovolnou kombinace písmen.

- *Metoda frekvence slov (TF)*
  Term Frequency - četnost slova v dokumentu.

- *Metoda inverzní frekvence slov (IDF)*
  Inverse document frequency - převrácená četnost slova ve všech dokumentech. Modifikace, která vylučuje často se vyskytující slova - počítá se frekvence vzácnějších slov [18].

- *Latentní sémantická analýza (LSA)*
  Latentní sémantická analýza (LSA) - způsob zpracování informací v přirozeném jazyce, který analyzuje vztahy mezi kolekcí dokumentů a výrazy vyskytující se v nich. Metoda latentní sémantické analýzy, založena na principu faktorové analýzy, zejména identifikace latentní souvislostí jevů nebo objektů. Matice obsahující slovo se počítá podle odstavce (řádky představují unikátní slova a sloupce reprezentují každý odstavec) a je zkonstruovaná z velké časti textu. Matematická metoda singulárního rozkladu SVD (Singular value decomposition) se používá ke snížení počtu sloupců při zachování struktury podobnosti mezi řádky. Slova se pak porovnávají výpočtem cosinového úhlu mezi dvěma vektory tvořené dvěmi řadami. Hodnoty blízké 1 představují velmi podobná slova, zatímco hodnoty blízké 0 představují velmi rozdílné slova.

  Poprvé byl LSA použit pro automatickou indexaci textů, identifikaci sémantické struktury textu a pro získání pseudo-dokumentu. V posledních letech se LSA metoda často používá pro vyhledávání informací (indexování dokumentů), klasifikaci dokumentů a dalších oblastí [19].

- *Metody stylometrie (odhad stylu autora)*
  Stylometrie zahrnuje statistické metody pro kvantifikaci autorova unikátního stylu psaní a je využívána zejména pro určování autorství. Sestavením a porovnáváním stylometrických modelů pro odlišné části textu, takzvané pasáže, které jsou stylisticky odlišné od ostatních, tedy potencionálně opsané z jiných zdrojů, mohou být tímto způsobem správně detekovány [4].

- *Karp-Rabinův algoritmus*
  Karp-Rabinův algoritmus - algoritmus pro vyhledávání podřetězce v textu pomocí hashovaní. Byl vyvinut v roce 1987 Michaelem Rabinem a Richardem Karpem. Pro text délky n a řetězec délky m, průměrná doba provedeni a nejlepší doba realizace je O(n) v nejhorším případě O(nm), což je jeden z důvodu, proč daný algoritmus není příliš používán [11].

Jedním z nejjednodušších praktických využiti algoritmu je detekce plagiátorství. Metoda se snaží urychlit zjištění ekvivalence vzorku pomoci hash funkce, která převede každý řetězec v hash hodnotu.

- *Exponentiated gradient (EG) algoritmus*
- *Support vector machines (SVM)*
  Řada metod a algoritmů strojového učení, používaná pro úkoly klasifikace a regresní analýze. Základní myšlenka spočívá v převodu původního vektoru v prostoru vyšší dimenze a vyhledávaní oddělující nadroviny s maximálním omezením v daném prostoru. Dvě paralelní nadroviny jsou postaveny na obou stranách nadroviny oddělující třídy. Oddělující nadrovina maximalizuje vzdálenost dvou paralelních nadrovin. Algoritmus funguje ze předpokladu, že čím vetší je rozdíl nebo vzdálenost mezi těmito paralelními nadrovinami, tím menši je průměrná chyba klasifikátora [8].
- *Markovove řetezce a sekvenční kernely*
- *Sémantické sítě*
  Sémantická síť - je informační model doménové oblasti ve tvaru orientovaného grafu, jehož vrcholy reprezentují objekty doménové oblasti a hrany definují vztahy mezi nimi [17].

  Všechny sémantické sítě lze rozdělit podle arity, velikosti a počtu typů vztahů.

  Podle počtu typů vztahů:

  – Homogenní
  – Nehomogenní

  Podle arity:

  – Binární
  – N-ární

  Podle velikosti:

  – Sémantická síť pro řešení konkrétní problémy;
  – Sémantická síť průmyslového rozsahu;
  – Globální sémantická síť.

  Počet typů vztahů v dané síti je definován jejím autorem na základě konkrétních cílů.
- *Coh-Metrix*
  Je výpočetní nástroj, který poskytuje více než 200 indexů soudržnosti, obtížnosti a čitelnosti. Coh-Metrix je citlivý na širokou škálu úrovní textových funkcí, které odrážejí soudržnost vztahů, světové znalosti, jazyky a diskuzní charakteristiky. Daný nástroj používá různé moduly: syntaktické analyzátory, latentní sémantika analýza a další funkce počítačové lingvistiky [6].

- *Evolutionary plagiarism probability funkce (EPP)*
- *Winnowing algoritmus*
- *Kullback-Leiblerová vzdálenost*
  V teorii pravděpodobnosti a informační teorii je nesymetrická míra rozdílů a to pravděpodobnostním rozdělením P a K. KL posuzuje očekávaný počet dalších potřebných bitů přikladu kódu P, při použití kódu na základě Q. Zpravidla P reprezentuje ″true″rozdělení dat, měřeni nebo výpočet teoretické distribuce. Míra Q obvykle reprezentuje teorii, model, popis nebo aproximace P [12].
- *Common semantic sequence model (CSSM)*
- *Google web API*
- *Asymetrický model podobnosti*
- *Heavy frequency vector (HFV)*
- *Standard copy analysis mechanism (SCAM)*

3.1 Kvalita plagiátorských systému

V Tabulce 1 jsou představeny kvantitativní a kvalitativní faktory ovlivňující kvalitu a komplexnost systémů pro detekce plagiátů a příslušných vyhledávácích algoritmu. V rámci této práci budeme se zabývat především mírou přesnosti a úplnosti.

3.2 Jaké kolekce se používají

Pro detekce plagiátu se používají následující kolekce:

- Extrakorpální nástroj, používá pro detekce plagiátu externí kolekce dokumentu (externí databáze, Internet). Výhodou daného nástroje - přistup k velké množině zdrojů dokumentu. Ale pak detekce plagiátu se stává náročnější. Třeba zvolit vhodné metody vyhledávaní, aby nástroj zachovával rozumnou rychlost a dobu odezvy.
- Intrakorpální nástroj není schopen detekovat plagiát, který byl vytvořen z jiné kolekce dokumentu, než aktuální zdroj. Při dané metodě jsou všechny dokumenty během kontroly považovaný za možný zdroj a také možný plagiát. V případě detekce dvojice se určuje autor a plagiátor.
- Kombinované nebo smísené nástroje, které kombinují intrakorpální a extrakorpální přístupy.
- Vnitřní nástroj založený na analýze obsahu dokumentu, kdy je zkoumáno, zdali nějaká část dokumentu je odlišná svou formou od ostatní časti dokumentu. V případě nalezení shodné části, lze předpokládat, že jde o plagiovanou část dokumentu.

Faktor	*Popis a možnosti*
Oblast vyhledávání	Veřejně na internetu, používáním lokálních kolekcí apod.
Doba indexaci	Čas potřebný pro vložení nového dokumentu do datového skladu.
Doba analýzy	Zpoždění mezi vložením dokumentu do systému a dobou vyhodnocení.
Kapacita zpracování	Počet dokumentů, které je schopen systém zpracovat za určitý čas.
Volba intenzity	Jak často a pro jaké typy částí dokumentů (odstavce, věty atd.) jsou vyhledávány.
Typ porovnávacího algoritmu	Volba algoritmu, který definuje, jakým způsobem se bude porovnávat mezi dokumenty.
Přesnost a úplnost	Míra označení dokumentů označených za plagiované. Vysoká přesnost znamená pouze pár přesnějších výsledků, nízká přesnost naopak více potencionálních nepřesných výsledků apod.
Dostupnost nástroje	Licence, informace o použitých metodách, provozovatel (nástroj, služba).
Podporované typy dokumentů	Obsah, jazyk, formát.

Tabulka 1: Faktory vyhodnocení systému

3.3 Míry vyhodnocení

V rozpoznávání a vyhledávání informace je přesnost podíl získaných případu, které jsou relevantní, zatímco úplnost je podíl příslušných instancí, které jsou získané.

Proto přesnost a úplnost jsou založeny na porozumění a měření jejich významů. Přesnost může být chápána jako míra přesnosti a kvality, zatímco úplnost je míra komplexnosti nebo množství. Vysoká úplnost ukazuje na to, že algoritmus vrátil většinu relevantních výsledků; vysoká přesnost pak ukazuje, že algoritmus vrátil více relevantních výsledků než irrelevantních.

V případě vyhledávání informace, kde cílem je vrátit sadu relevantních dokumentů s ohledem na podmínky vyhledávání nebo přiřadit každý dokument do jedné ze dvou kategorií ″relevantní″a ″není relevantní″. V daném případě jsou ″relevantní″dokumenty pouze ty, které patří do kategorie ″relevantní″. Úplnost je definovaná jako počet relevantních dokumentu získaných vyhledáváním děleno celkovým počtem existujících relevantních dokumentů, zatímco přesnost je definovaná jako počet relevantních dokumentu získaných vyhledáváním děleno celkovým počtem dokumentů, získaných daným vyhledáváním.

V případě klasifikace, přesnost třídy je počet pozitivních ″true″(tj. počet prvků správně označených jako náležející do pozitivní třídy) děleno celkovým počtem prvků, označených jako patřící do pozitivní třídy (tj. součet pravých pozitivních a falešně pozitivních výsledků, které jsou nesprávně označeny jako patřící do skupiny). Úplnost v daném kontextu je definovaná jako počet pravých pozitivních prvků dělen celkovým počtem prvků, které ve

skutečnosti patří do pozitivní skupiny (tj. součet pravých pozitivních a falešně negativních, což jsou prvky, které nebyly označeny jako náležející do pozitivní skupiny, ale měly by být).

Při vyhledávání informace hodnota ideální přesnosti 1.0 označuje, že každý výsledek, získaný vyhledáváním byl relevantním (ale neříká nic o tom, zda všechny relevantní dokumenty byly získány). Ideální hodnota úplnosti 1.0 ukazuje, že všechny příslušné dokumenty byly získány vyhledáváním (ale neříká nic o tom, kolik irelevantních dokumentů bylo získáno) [14].

V kontextu vyhledávání informace přesnost a úplnost jsou definovány řadou získaných dokumentů a řadou relevantních dokumentů.

Přesnost

V oblasti vyhledávání informace je přesnost podíl získaných dokumentů, které jsou relevantní při vyhledávaní:

$$precision = \frac{|\{relevant\ dokuments\} \cap \{retrieved\ dokuments\}|}{|\{retrieved\ dokuments\}|}$$

Úplnost

Úplnost v oblasti vyhledávání informace je podíl dokumentů, které jsou relevantní k dotazu:

$$recall = \frac{|\{relevant\ dokuments\} \cap \{retrieved\ dokuments\}|}{|\{total\ retrieved\ dokuments\}|}$$

Pro účely klasifikace pojmy skutečně pozitivní, skutečně negativní, falešně pozitivní a falešně negativní porovnávají výsledky klasifikace v rámci testu s externě důvěryhodnými rozsudky. Pojmy ″pozitivní″a ″negativní″odkazují na predikce klasifikace (někdy známa jako pozorování), pojmy ″true″a ″false″odkazují na to, zdali odpovídá předpověď na externí rozsudky (viz Tabulka 2).

	Aktuální třída (očekávání)	
Třída předpokladu (pozorování)	Skutečně pozitivní (true positive - **TP**) Správný výsledek	Falešně pozitivní (false positive - **FP**) Neočekávaný výsledek
	Falešně negativní (false negative - **FN**) Chybějící výsledek	Skutečně negativní (true negative - **TN**) Správná absence výsledku

Tabulka 2: Pojmy pro klasifikační účely

Přesnost a úplnost pak definovaná jako:

$$precision = \frac{TP}{TP+FP}$$

$$recall = \frac{TP}{TP+FN}$$

Úplnost v daném kontextu je taky označovaná jako ″skutečně pozitivní míra″(true positive rate) a přesnost je taky označovaná jako ″pozitivně prediktivní hodnota″(positive

predictive value - PPV); další související míry používané při klasifikaci včetně ″skutečně negativní míra″tz. specifičnost (true negative rate) a přesnost.

$$true\,negative\,rate = \frac{TN}{TN+FP}$$

$$accuraty = \frac{TP+TN}{TP+TN+FP+FN}$$

Pravděpodobnostní interpretace

Existuje možnost interpretovat přesnost a úplnost jako pravděpodobnost. Úplnost je pravděpodobnost, že (náhodně vybraný) relevantní dokument je získán vyhledáváním. Systém pro detekce plagiátu muže obsahovat následující míry pro hodnocení dosazených výsledků [14]:

- přesnost

  $$precision = \frac{1}{|S|}\sum_{i=1}^{S}\frac{\#detected\,chars\,of\,s_i}{|s_i|}$$

- úplnost

  $$recall = \frac{1}{|R|}\sum_{i=1}^{R}\frac{\#plagiarized\,chars\,of\,r_i}{|r_i|}$$

- zrnitost

  $$granularity = \log 2(1 + \frac{1}{|S_R|}\sum_{i=1}^{S_R}\#detections\,chars\,of\,s_i\,in\,R)$$

- míra F1

  $$f1 = \frac{2 * precision * recall}{precision + recall}$$

Kde S je množina detekovaných částí textu a s_i je jedna detekována část textu, R je množina reálných plagiovaných částí textu a r_i je jedna plagiovaná část textu. S_R je množina reálných plagiovaných částí textu, ve kterých existuje alespoň jedna detekce.

S, R a S_R označují počet pasáží v množině, s_i a r_i - počet znaků v pasáži [2].

4 Vyhledávání dokumentů a částí dokumentů

Proto, abychom mohli správně identifikovat vyhledávanou informaci, je potřeba nejprve daný dokument či kolekci dokumentů vhodně rozdělit na dílčí kusy, případně odstranit zbytečná slova z dokumentů apod. Jinými slovy potřebujeme před-zpracovat dané dokumenty, a k tomu právě slouží níže popsané metody.

Pokud bychom tyto metody pro zpracování textu nepoužili, bylo by velmi pravděpodobné, že bychom například neobdrželi požadovaný výsledek, a to z důvodů nejrůznějších koncovek slov, skloňování. V případě že bychom neodstranili zbytečná slova, která pro vyhledávání nejsou důležitá, bylo by vyhledávání o poznání pomalejší atd. Toto je pouze malý výčet důvodů pro před-zpracování.

4.1 Před-zpracovaní textu a indexace

Popišme si nyní jednotlivé metody pro předzpracování textu a indexaci v přehledných bodech.

4.1.1 Tokenizace

Jedná se o rozdělení textového korpusu na jednotlivé slovní tvary - tokeny. V první fázi dojde k rozdělení textu na jednotlivé úseky znaků, které jsou vnímány jako slova, čísla atd. a k odstranění doplňujících znaků, kterými jsou zejména interpunkční znaménka. Tyto jednotlivé úseky znaků pak ve finále vytvoří indexová slova (termy).

Nejjednodušší metodou jak získat vhodné tokeny, je nahradit všechny nepísmenné znaky tzv. bílými, a následně podle těchto bílých znaků rozdělit text na jednotlivé tokeny. Toto řešení má však své nevýhody. V prvé řadě je zde riziko, že přijdeme o některé podstatné informace z textu, ale také například můžeme narazit u čísel na problém s vyjádřením některých údajů (např. datum zapsané jako 22. 4. 2012 se rozdělí na 3 tokeny ″22″, ″4″a ″2012″). Jednotné řešení, které by neznevýhodnilo některé typy řetězců, zřejmě nenajdeme. Téměř vždy najdeme případ, kdy proces tokenizace nebude vhodný (např. u zdrojových kódů). Proto je nasnadě najít takové řešení, které poskytne správný podklad pro vyhledávání, tedy takový tvar, který by zachoval původní význam řetězce. V neposlední řadě je také vhodné podotknout, že tokenizace se také potkává s problémem jako je velikost písmen či diakritika u českého jazyka [21].

4.1.2 Eliminace stop-slov

Odstranění stop-slov je základní před-zpracovatelský přístup, který odstraňuje běžná slova. Jeho primární využití je předejít přehlcením velmi frekventovanými slovy. Pro detekci plagiátů to však může znamenat komplikaci, a to v případě odstranění těchto slov můžeme porušit autorův styl psaní. Pro tyto důvody je efekt odstranění stop-slov spíše nepředvídatelný. Obvyklý způsob určování toho, co se počítá jako stop-slovo je pouze používáním slovníků, které je definují [5].

4.1.3 Lemmatizace

Lemmatizace je proces určení základního tvaru k určitému tvaru slova. Během tohoto procesu je k určení významu slova využit kontext. Někdy se lemmatizace zaměňuje s pojmem ″kořen″, nicméně je tady zásadní rozdíl. Kořen se vztahuje ke slovu bez jakékoliv znalosti kontextu, a tedy nemůže rozlišit mezi slovy mající různý význam. Příklad takového kořenu může být v angličtině slovo ″does″a ″done″. Zde by se kořen určil jako ″do″, což je zavádějící. Na druhou stranu lemmatizace umí rozlišit podle kontextu význam slova. Toto je částečně důležité pro jazyky, které mají bohatý systém na skloňování jako například čeština [5].

4.1.4 Normalizace

Hlavní myšlenkou normalizace je zpracování různých specifických slov více obecným výrazem, či nadřazeným. Například slova ″pes″a ″kočka″mohou být obě nahrazena slovem ″zvíře″, anebo jiným obecnějším nadřazeným slovem (např. ″savec″atd.). To má za následek dva cíle, prvním je snížení počtu různých slov, která musejí být zpracována, druhým cílem je odhalení přítomnosti plagiátorství, tedy v případě, že se někdo snažil jakýkoli text parafrázovat, opsat jinými synonymy, obecnými pojmy apod.

Slovník synonym WordNet propojuje jednotlivé skupiny synonym, které jsou sémanticky podobné, přičemž využívá četnou mezi-slovníkovou sadu referencí (ILR) kde skupina synonym obsahuje alespoň jedno či více slov stejného významu. Vztah hyperonyma pak popisuje hierarchii takových skupin synonym – tedy určuje nadřazenost obecných slov vůči ostatním apod.

Myšlenka nahrazování konkrétních slov více obecnými slovy je jednoduchá. Problémem je akorát jak vyřešit volbu která konkrétní slova použít. Což znamená, že pokud nejsme dostatečně obecní, dosáhneme malé výhody. Pokud jsme až moc obecní, pak budou všechny nominální výrazy nahrazeny nějakým ″subjektem″a ztratíme příslušnou informační množinu. Nejlepší metoda je tedy zvolit individuální zobecnění pro každou sub-hierarchii. Nicméně, taková metoda je víceméně nepraktická. V praxi je tedy nejlepší cestou zvolit pevnou úroveň pro globální zobecnění. Což znamená, že od tohoto bodu všechna slova, která jsou pod touto úrovní, budou nahrazena obecným výrazem, a naopak slova, která budou na stejné nebo vyšší úrovni zůstanou zachována [5].

4.2 Klasifikace modelů vyhledávání

Většina dnešních informačních systémů je stále založena na klasickém booleovském vyhledávání. Dalšími způsoby vyhledávání patří například vektorové, bayesovské a neuronové sítě, ale také latentní sémantické indexování. Tyto metody jsou zatím ale povětšinou ve vývojové fázi, ale mají přinést daleko větší efektivitu při použití velkých objemů dat. Je potřeba poznamenat, že metody vyhledávání dělíme do dvou základních skupin, a to podle toho jakým způsobem jsou vyhledávány údaje. Prvním je ″vyhledávání a prohlížení″, druhým způsobem je ″vyhledávání a filtrace″.

Způsob vyhledávání a prohlížení zahrnuje aktivní vyhledávání příslušného uživatele, a to ve formě dotazu formulovaného v příslušném dotazovacím jazyce informačního systému.

Jako výsledek obdrží sestavu dokumentů, které by měly odpovídat zadanému požadavku uživatele. Prohlížení pak vychází z toho, že samotný uživatel nemusel zadat úplně přesně správná klíčová slova nebo naopak byl dotaz příliš široký, a proto díky interaktivnímu rozhraní může jednoduše projít výsledky vyhledávání a může tak nalézt relevantní dokument, případně být odkázán na další dokumenty apod. Navíc se zavádí možnost průběžně požadavek upravovat či lépe specifikovat. Nejnovější informační systémy většinou kombinují vyhledávání a prohlížení.

Druhý způsob, vyhledávání a filtrace, využívá jakýsi profil uživatele a jeho definici na základě nejčastěji vyhledávaných dokumentů. Informační systém tedy uživateli nabídne i data, ke kterým nebyl zadán požadavek, avšak je zde pravděpodobnost, že by se o ně mohl zajímat.

Jednotlivé modely lze pak ještě rozdělit na klasické (kde patří booleovské, vektorové a pravděpodobnostní vyhledávání) a na strukturované (pro vyhledávání ve strukturovaném textu), kde pak dále patří modely ″nepřekrývajících se seznamů″a ″sousedních uzlů″. Aby to nebylo tak jednoduché, ke klasickým modelům ještě existují další alternativy, jako například rozšířený booleovský model, zobecněné vektorové vyhledávání, neuronové či bayesovské sítě, nebo také latentní sémantické vyhledávání [2, 3].

4.2.1 Booleovský model

Booleovský model je jeden z nejstarších a také stále nejpoužívanějších modelů pro vyhledávání informací. Jeho stavebním kamenem je teorie množin a booleovská algebra. Jednotlivé dotazy pro vyhledávání jsou dány booleovskými výrazy, které se aplikují na každý dokument. Výsledkem pak jsou pouze takové dokumenty, které přesně odpovídají zadanému dotazu. Takový je princip booleovské algebry, který neumožňuje najít pouze částečnou shodu. Proto se jako model pro vyhledávání informací používá tam, kde je tato jeho vlastnost žádoucí.

Hodnoty indexových slov jsou tedy binární (0 či 1) a dotaz je tvořen jednotlivými indexovými slovy a operátory AND, OR a NOT. Z toho vyplývá, že je v tomto případě možné používat logiku jako takovou. Hlavní výhodou tedy je jeho jednoduchost, avšak právě ona zmíněná logická striktnost je jeho zásadní nevýhodou [13].

4.2.2 Vektorový model

Vektorový model na rozdíl od modelu booleovského pokrývá jeho negativní vlastnosti – nemožnost nalézt pouze částečnou shodu. Indexová slova mají opět přiřazenou váhu, ovšem není zadána pouze 1 nebo 0, ale je reprezentována t-rozměrným vektorem. Dimenze takového vektoru se skládají z váhy všech indexovaných slov. Takovéto vektory jsou využívány k výpočtu stupně podobnosti dokumentu či dotazu. Díky tomu je možné nalézt i částečnou shodu ve vyhledávání, a navíc podle stupně podobnosti řadit výsledky vyhledávání podle relevance.

Vektorový model tedy umožňuje určit stupeň podobnosti dokumentu či dotazu, a tím určit meze, podle kterých budou dokumenty hodnoceny podle tohoto stupně podobnosti

jako relevantní. Pro pochopení vektorového modelu si můžeme představit, že vyhledáváme v databázi podle následující množiny klíčových slov:

- horské kolo,
- trekingové kolo,
- přehazovačka,
- kotoučové brzdy,
- klasické brzdy,
- přední vidlice,
- sedátko.

Tato databáze pak bude obsahovat jednotlivé dokumenty, které zahrnují daná klíčová slova:

1. *První dokument obsahuje:* Horské kolo, přehazovačka, sedátko;
2. *Druhý dokument obsahuje:* Trekingové kolo, přehazovačka, sedátko;
3. *Třetí dokument obsahuje:* Trekingové kolo, kotoučové brzdy, přední vidlice;
4. *Čtvrtý dokument obsahuje:* Horské kolo, přehazovačka.

Vektorově pak podle klíčových slov můžeme tyto dokumenty zapsat jako:

1. První: (1,0,1,0,0,0,1),
2. Druhý: (0,1,1,0,0,0,1),
3. Třetí: (0,1,0,1,0,1,0),
4. Čtvrtý: (1,0,1,0,0,0,0).

Kde 0 a 1 reprezentují, zda se dané klíčové slovo nevyskytuje nebo vyskytuje v daném dokumentu. Pokud bychom tedy například vyhledávali pomocí klíčových slov **horské kolo** a **přehazovačka**, pak bychom dostali jako výsledek vyhledávání **první** a **čtvrtý** dokument. Naopak **třetí** dokument by vůbec neodpovídal danému vyhledávání. Vektorovým součinem je pak možné dané výsledky seřadit od nejvíce shodného dokumentu až po dokument, který se shoduje nejméně (nebo vůbec).

Tento systém lze dále rozšířit pomocí tzv. systému vážení. V praxi to znamená, že pokud se například v daném dokumentu vyskytuje klíčové slovo **přehazovačka** celkem čtyřikrát a například **sedátko** dvakrát, můžeme daný výsledek zapsat jako **(0,0,4,0,0,0,2)**. Z toho vyplývá, že dané vektory již nevyjadřují jen shodu v dokumentu, ale navíc přináší také kvantitativní hodnocení podobnosti dokumentu. Tato myšlenka je tedy základním prvkem vektorového modelu [22].

4.2.3 Pravděpodobnostní model

Model pravděpodobnosti se snaží vyřešit vyhledávání informací pomocí matematické teorie pravděpodobnosti. Myšlenka tohoto modelu je taková, že pro nějaký dotaz existuje množina dokumentů, která obsahuje pouze relevantní dokumenty a žádné jiné. Tato množina je pak označována jako ideální odpověď. Proces dotazování můžeme uvažovat jako proces hledání ideální množiny. V praxi to pak znamená, že se nejprve vytvoří počáteční odhad množiny relevantních dokumentů. Poté je určitá hodnota pravděpodobnosti taková, kdy je dokument relevantní k dotazu a naopak zase od určité hodnoty je pravděpodobnost, že dokument není relevantní k dotazu. Tento model ale není více používán a rozebírán v této diplomové práci [16].

4.2.4 Samo-organizační mapy (SOM)

Využití samo-organizačních map (z angl. názvu Self Organizing Maps - SOMs) pro automatickou organizaci fulltextových kolekcí dokumentů se ukázala jako neocenitelná pomůcka k tradičnímu vyhledávání informací. Tento systém má schopnost setřídit dokumenty s podobným obsahem v sousedních oborech, které jsou analogicky porovnatelné k dané situaci a vyskytující se běžně v knihovnách. Takové seřazení skloubené s tradičními nástroji pro vyhledáváním informací pomůže uživatelům nejen nalézt vyhledávané informace a získat dokumenty v rámci jednoho tématu, ale také získat přehled o celé kolekci dokumentů a prozkoumat další rozšiřující informace daného tématu, které by je mohly zajímat a pokrýt oblasti, o kterých ani netušili.

Samo-organizační mapy jsou typem samostatně fungujících neuronových sítí s umělou inteligencí, které poprvé představil finský vědec Teuvo Kohonen z kraje osmdesátých let. Tento typ neuronových sítí je většinou dvourozměrná mřížka neuronů, z nichž všechny mají za referenční model vážený vektor (kapitola 3.2.2). Jako většina neuronových sítí mohou SOM pracovat ve dvou režimech, učící se a mapování. Ve školícím režimu se staví mapa využívající vstupní vzorky. Jedná se o proces výběru, také nazývaný jako kvantizace vektoru. Mapování pak automaticky klasifikuje nový vstupní vektor.

SOM se skládá z jednotlivých uzlů (neboli neuronů). S každým takovým uzlem je pak svázán vážený vektor, a to stejného rozměru jako vstupní datové vektory a pozice v mapovém prostoru. Obvyklé uspořádání takových uzlů je pak ve stejné vzdálenosti ve tvaru šestiúhelníku či obdélníkové mřížky. SOM popisuje mapování z vyšších vstupních prostorových dimenzí na menší. Proces umístění vektoru z datového prostoru do příslušné mapy je prvně nalézt uzel s nejbližší váženým vektorem k vektoru vybraného z datového prostoru. Jakmile je nalezen nejbližší uzel, jsou mu přiřazeny hodnoty právě tohoto vektoru [7].

4.2.5 LSH – Locality sensitive hashing

Je metoda pro provádění pravděpodobnostního zmenšování rozsáhlých datových kolekcí. Základní myšlenka je hashování vstupních prvků tak, že podobné prvky jsou mapovány do stejných segmentů s vysokou pravděpodobností (počet segmentů je mnohem menší než prostor všech možných vstupních prvků). Daná pravděpodobnost pak slouží ke zjišťování

sousedních či podobných prvků. Je dobré si povšimnout, jak LSH ve mnoha ohledech napodobuje datové seskupování [23].

4.2.6 n-Gramy

V oblasti počítačové lingvistiky a pravděpodobnosti je n-gram souvislá posloupnost n prvků v dané sekvenci textu či řeči (v konceptu). Tyto prvky v otázce mohou být fonémy, slabiky, slova nebo základní sousloví podle kontextu. N-gramy jsou vybírány z textového korpusu či řeči.

N-gram o velikosti 1 se pak označuje jako ″unigram″, při velikosti 2 se jedná o ″bigram″ (či ″digram″), u velikosti 3 je to ″trigram″. U větších se pak často říká ″pěti-gram″(five-gram) apod.

Model n-gram je typ pravděpodobnostního jazykového předvídání dalšího prvku v závislosti na určitém pořadí nějaké formy – jedná se o model pořadí Markova. N-gram modely jsou dnes široce používány v pravděpodobnosti, teorií komunikací, výpočetní lingvistice (např. statistické zpracování přirozeného jazyka), výpočetní biologii (např. analýza biologických sekvencí) či kompresi dat. Dvě základní výhody modelů n-gramů (a algoritmů, které je využívají) jsou relativní jednoduchost a možnost rozšíření. A to jednoduchým zvýšením n parametru může být model použit k uložení více kontextů s velmi dobrým časo-prostorovým kompromisem, což znamená, že z malých projektů mohou vyrůst velké velmi efektivně [15].

5 Teorie a metody hotových systémů

Základní dvě rozdělení detekce plagiátů , které snad není ani třeba nijak zvlášť rozebírat, jsou manuální a automatické pomocí počítače. Manuální vyžaduje značné úsilí a excelentní paměť a je velice nepraktická. Počítač nám naopak bez většího úsilí může porovnávat rozsáhlé sbírky dokumentů a umožní detekci plagiátů o poznání jednodušeji. Navíc s nadsázkou můžeme říct, že počítač bude ve většině případů mnohem úspěšnější v nalezení plagiátu (samozřejmě v případě, že je systém řádně navržen).

Počítačem asistovanou detekci plagiátů obecně nazýváme jako systém pro detekci plagiátů.

5.1 Detekce plagiátů dokumentů

Systémy pro detekci opsaných dokumentů mohou být implementovány dvěma základními způsoby, externím či interním.

Externí typ porovnává podezřelý dokument s referenční kolekcí dat, což je sada dokumentů považovaných za originály.

Úkolem takové detekce je poté na základě předem definovaných kritérií podobnosti získat dokumenty, které obsahují podobný text. Podobný text se myslí do předem nastavené úrovně definované pro dokument podezřelý z plagiace.

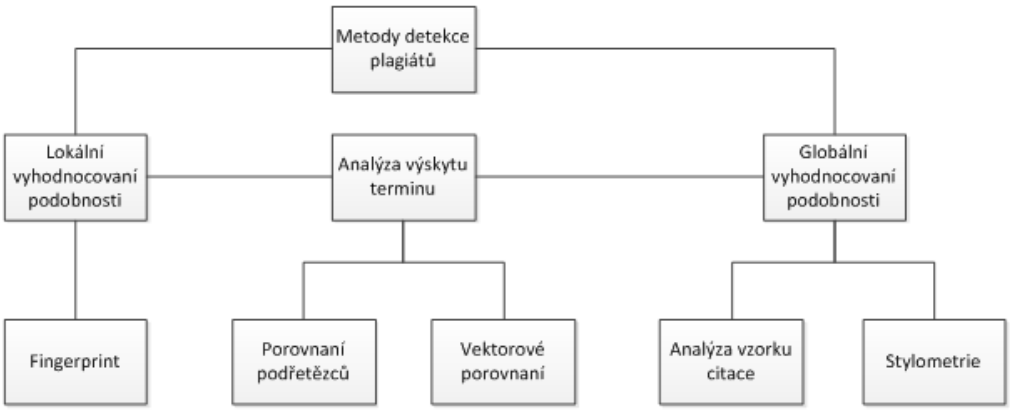
Interní typ pak pouze analyzuje text, který se bude vyhodnocovat, a to bez porovnávání s externími dokumenty. Tímto přístupem se detekce zaměřuje na rozpoznávání odlišností ve stylu psaní od originálního autora.

Ani jeden z těchto druhů se neobejde bez dodatečného schválení či finální analýzy samotného člověka, podobnosti jsou vyhodnocovány pouze jako pomůcka pro odhalení potencionálního plagiátu a často vykazují planý poplach.

5.1.1 Detekční metody

Obrázek 1 reprezentuje rozdělení metod pro detekci plagiátů, a to z technického pohledu. Tyto techniky jsou rozděleny podle toho, jaké využívají hodnocení podobnosti. Globální vyhodnocování podobnosti využívá prvků vytržených z větších částí dokumentů či textů a porovnává to jako celek, zatímco lokální vyhodnocování bere daný text pouze jako vstup.

Fingerprint (otisk) je v současnosti nejrozšířenější přístup k detekci plagiátů. Procedura sestavuje typický výběr z daného dokumentu prostřednictvím volby několika podřetězců tzv. n-gramů. Jednotlivé prvky n-gramů se potom nazývají ″minutiae″(což by volně přeloženo z angličtiny znamenalo ″markanty″nebo ″maličkosti ″). Podezřelý dokument
je pak překontrolován vyhodnocením jeho otisku a porovnáváním jeho markantů s předzpracovanými indexy všech dokumentů z referenční kolekce. Postupně se pak nacházejí shody markantů s částmi dané kolekce a potencionální plagiátorství se navrhuje ve chvíli, kdy se překročí určitá úroveň předem zvolené podobnosti. Obecně pak platí, že se porovnává pouze menší podčást markantů, aby se urychlil proces vyhledávání v rozsáhlých kolekcích, jakou je například internet apod.



Obrázek 1: Detekční metody

Kontrolování dokumentů na úseky citovaného textu představuje klasický problém s vyhledáváním podobnosti při **porovnávání podřetězců**, který je známý z jiných oblastí počítačových věd. Již nespočet způsobů byl představen k řešení tohoto úkolu, z nichž některé byly použity pro počítačem asistovanou detekci plagiátů. Vyhledávání podezřelého dokumentu v tomto případě vyžaduje příslušný výpočetní výkon a dostatečně efektivní úložiště v takové podobě, aby bylo možné porovnávat veškeré dokumenty z referenční kolekce vždy po párech s podezřelým dokumentem. Tedy pro automatickou detekci se využívají modely pracující se sufixy (jako sufixové stromy či vektory). Nicméně porovnávání takových podřetězců stále zůstává velice výpočetně náročné, což z něj činí spíše neschůdné řešení v porovnávání dokumentů s rozsáhlými kolekcemi dat.

″**Bag of Words**″analýza (sada slov) představuje vektorové porovnávání termínů a je klasickým způsobem pro IR model. Jednotlivé dokumenty jsou reprezentovány jako jeden či více vektorů. Tento způsob může být použit pro tradiční kosinové porovnávání podobnosti či více sofistikované podobnostní funkce.

Analýza vzorku citace je způsob počítačem asistovaná detekce plagiátů využívaný zejména pro akademické účely, a to protože se tento způsob nezakládá na samotném textu, ale na citacích a referencích. Jedná se o identifikaci podobných vzorků v posloupnosti citací ve dvou různých akademických pracích. Vzory citací představují nejen podsekvence obsahující citace sdílené oběma porovnávanými dokumenty, ale také identifikují podobné pořadí a blízkost citací v daném textu. Zváženy jsou i další faktory jako stejný počet nebo relativní část sdílených citací mezi dokumenty, nebo například pravděpodobnostní určení společných výskytů citací v dokumentech jsou považovány za kvantifikátory stupně podobnosti.

Stylometrie zahrnuje statistické metody pro kvantifikaci autorova unikátního stylu psaní a je využívána zejména pro určování autorství. Sestavením a porovnáváním stylometrických modelů pro odlišné části textu, pasáže, které jsou stylisticky odlišné od ostatních, tedy potencionálně opsané z jiných zdrojů, mohou být tímto způsobem správně detekovány.

5.1.2 Systémy detekce plagiátů pro textové dokumenty

Většina rozsáhlých systémů pro detekci plagiátů využívá objemnou interní databázi, která se rozrůstá s každým přidaným dokumentem pro analýzu. Nicméně někdy se může jednat o případné porušení studentských autorských práv.

Systemy uvedene v Tabulce 3 jsou většinou dostupné on-line a až na CopyTracker mají interní zdroje.

Volně dostupné	*Komerční*
Chimpsky	Attributor
CitePlag	Copyscape
CopyTracker	Iparadigms
eTBLAST	Plagiarismdetect
Plagium	PlugScan
SeeSources	Veriguide
The Plagiarism checker	
Plagiarism detect	

Tabulka 3: Existující řešení

5.1.3 Detekční výkon

Výkon systémů pro detekci plagiátů závisí na typu použitého přístupu. S výjimkou analýzy vzorků citace se všechny přístupy detekce zakládají na textové podobnosti.

Doslovné kopírování (Copy&Paste) či mírně upravené texty mohou být detekovány s vysokou přesností. Zejména pak procedury pro vyhledávání podřetězců dosahují velmi slušného výkonu v případě Copy&Paste plagiátorství, jelikož využívají bezeztrátové modely dokumentů, jakými jsou například sufixové stromy. Výkon systémů využívajících otisků či ″bag of words″analýzy pro detekci kopií závisí na informacích, které jsou vytraceny v závislosti na použitém modelu dokumentů. Při správném použití výběrových strategií jsou schopny lépe detekovat mírně upravené plagiované texty ve srovnání s procedurami porovnávání podřetězců.

Interní detekce plagiátorství za použití stylometrie může často do jisté míry překonat hranice podobnosti textů porovnáváním jazykové podobnosti. Vzhledem k tomu, že stylistické rozdíly mezi plagiovaným a originálním textem jsou velmi značné, mohou být spolehlivě detekovány. Stylometrie tak může napomoci k identifikaci upraveného či parafrázovaného textu. Stylometrie pak selhává ve chvíli, kdy se parafrázovaný text už po-

dobá spíše stylu psaní plagiátora, či v případě že text byl upraven již několika plagiátory po sobě.

V současné době se stále více zlepšují možnosti a systémy schopné detekovat plagiáty přeložené z cizího jazyka. Stále však nejsou tyto mezi-jazykové systémy považovány za vyspělou technologii a příslušné systémy zatím nebyly schopny dosáhnout uspokojivých výsledků v praxi.

Detekce na základě citací v textech (analýza vzorků citací) je schopna mnohem silněji a úspěšněji identifikovat parafráze a překlady v porovnání s ostatními metodami pro detekci, a to také proto, že je nezávislá na charakteristice textů. Nicméně protože taková analýza je závislá na příslušné dostupnosti dostatečných informací a citacích, je zaměřena pouze na akademické dokumenty.

5.1.4 Detekce plagiátorství zdrojových kódů

Plagiátorství zdrojových počítačových kódu je také velmi častá věc a vyžaduje jiné nástroje a metody, než ty uvedené pro textové dokumenty.

Významným rysem plagiátorství zdrojového kódu je, že zde neexistují žádné možnosti napsat kód jinými slovy, jak je tomu u textových dokumentů. Jelikož se u úkolů z programování předpokládá psaní kódu s velmi specifickými požadavky, je velmi těžké pro studenta najít již hotový kód, který by mohl použít. A vzhledem k tomu, že upravit nějaký složitý hotový kód je často o dost náročnější než jej napsat celý sám, většina studentských plagiátorů využívá své vrstevníky k napsání.

Algoritmy pro detekci podobnosti zdrojových kódů mohou pak být založeny na:

- **Řetězcích** - vyhledává pro přesné shody segmentů, například v cyklech po pěti slovech. Rychlé, avšak snadné ošidit přejmenováním identifikátorů.
- **Tokenech** - stejné jako u řetězců, avšak za použití lexikální analýzy k rozdělení programu na tzv. tokeny, což zahodí bílé znaky, komentáře, jména identifikátorů apod. Využíváno u akademických systémů.
- **Parsovacích stromech** - sestavení a porovnávání parsovacích stromů.
- **Grafech programových závislostí** - tyto grafy jsou schopné zachytit aktuální řídící toky programu a odhalit tak plagiát na vyšším stupni.
- **Metrice** - metrikou lze zachycovat různé počty výskytů parametrů apod., například stejné počty cyklů, proměnných atd.
- **Hybridních přístupů** - pro instance, parsovací stromy a suffixové stromy může kombinovat možnosti detekcí parovacích stromů a rychlostí jakou oplývá sufixová metoda.

Předchozí klasifikace byla vyvinuta pro refaktorizaci kódu, nikoli pro akademické rozpoznávání plagiátů (důležitým cílem refaktorizace je vyhnout se duplikovanému kódu odkazovaným jako klon kódu v literatuře). Výše popsané metody jsou efektivní proti různým

úrovním podobnosti, nízká úroveň podobnosti se popisuje jako identický text, kdežto vysoká úroveň pak jako podobná specifikace. Na akademické půdě, kde se u všech studentů předpokládá kódování podle stejných specifikací a funkčně stejných kódů, je detekce podle nízké úrovně podobnosti považována jako důkaz podvodu.

Existuje více systému pro detekci plagiátů zdrojových kódů, uveďme si dva nejznámější, které jsou navíc zdarma (požaduje se pouze registrace) - MOSS [1] a JPlag [2].

[1] http://theory.stanford.edu/ aiken/moss/
[2] https://www.ipd.kit.edu/jplag/

6 Návrh a implementace systému detekce plagiátu

Praktickou část můžeme rozdělit na dvě pomyslné části, ve kterých bylo potřeba navrhnout a implementovat hned několik různých prvků.

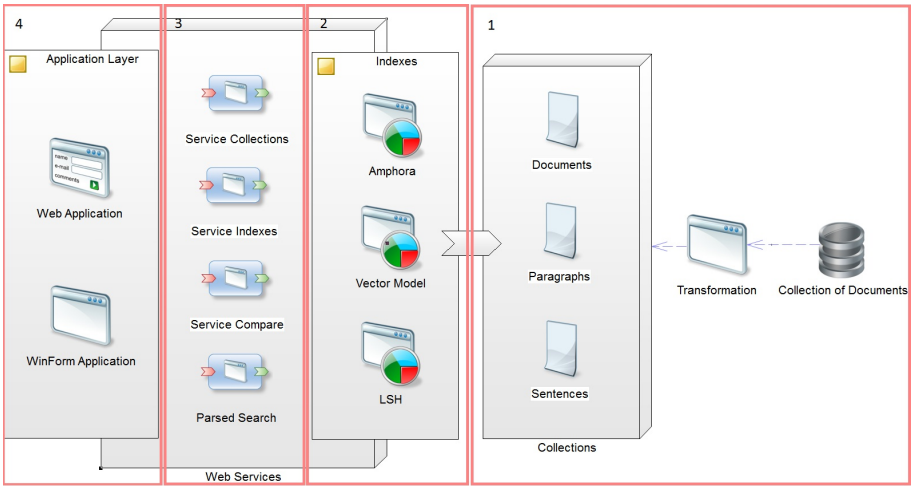
V první části bylo potřeba vytvořit GUI pro existující vyhledávácí systém Amphora. Toto GUI by mělo zachovat existující funkce systému Amphora a zároveň zmodernizovat uživatelské rozhraní pro lepší interakci s uživatelem. Dále se mělo uživatelské rozhraní propojit s rozhraním systému pro přístup k vyhledávácím prvkům, a to prostřednictvím webových služeb.

Do druhé pomyslné části bychom mohli zařadit návrh samotné webové aplikace pro vyhledávání a odhalování plagiátů. Dále návrh webových sužeb, které slouží jako rozhraní mezi různými druhy aplikací a vyhledávacími modely. Myšlenka je taková, že uživatel vloží přes webovou aplikaci nějaký dokument a zvolí, jestli se má dokument rozdělit na věty, odstavce, případně má zůstat celý, a následně vybere, podle kterého vyhledávacího modelu chce porovnávat daný dokument a také příslušnou kolekci dat, se kterou jej chce porovnat. Tento dokument je pomocí webových služeb zpracován, a podle vybraného modelu porovnán s příslušnou kolekcí dokumentů. Výsledek tohoto vyhledávání je poté odeslán zpět do webové aplikace, kde je vhodně provedena vizualizace a uživatel tak má možnost porovnat dané výběry

6.1 Návrh architektury

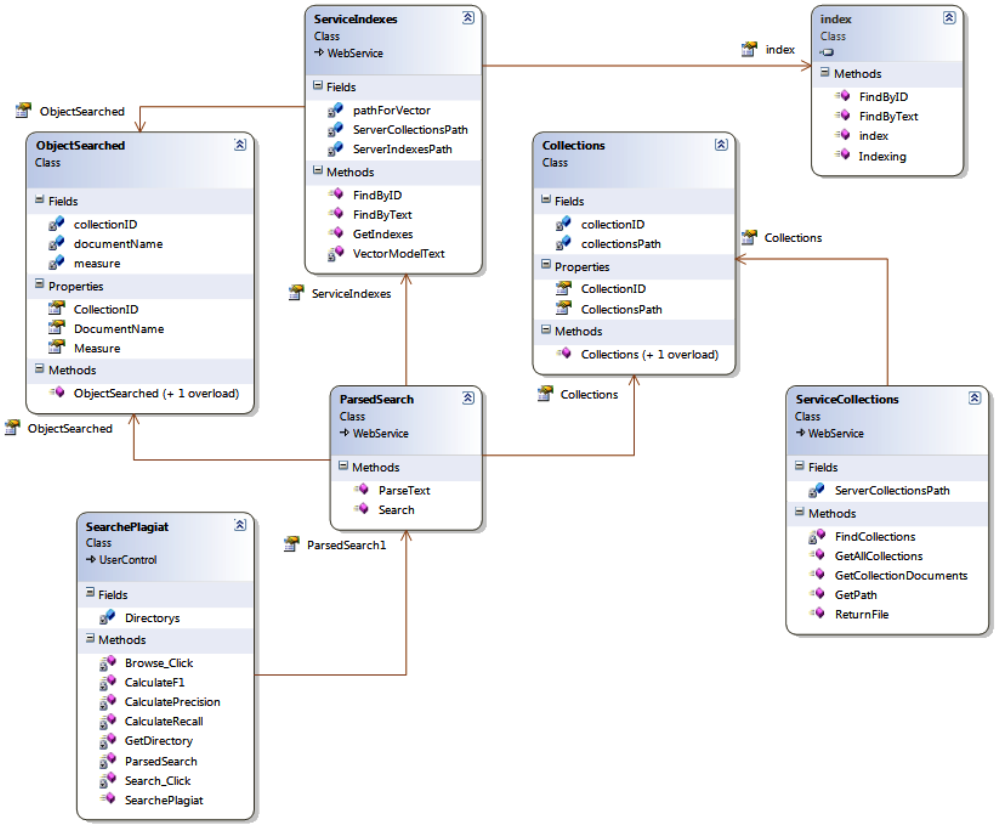
Celá architektura se skládá ze čtyř hlavních bloků:

- První blok slouží pro prvotní zpracování a rozděluje každý dokument na jednotlivé dílčí části (celý dokument, odstavce, věty). Po provedení prvotního zpracování vzniknou celkem 3 nové kolekce (viz Obrázek 2).



Obrázek 2: Architektura systému

- Další část obsahuje mechanizmy a algoritmy sloužící pro indexaci a vyhledávání dat. Tyto mechanizmy byly detailně popsány v 3. Kapitole (viz Obrázek 2).
- K tomu abychom mohli přistupovat k indexům a provádět vyhledávání, bylo vytvořeno rozhraní webových služeb (viz Obrázek 2).
- Následně na tyto webové služby byla napojena aplikační vrstva, na které byly realizovány různé druhy vizualizací a vyhodnocování plagiátů (viz Obrázek 2).



Obrázek 3: Třídní diagram

Popišme si nyní, jakým způsobem je realizován celý koncept aplikační architektury v kostce pro lepší pochopení (viz Obrázek 3).

Všechno začíná vložením dokumentu a výběrem příslušného IR systému spolu s datovou kolekcí. Tímto se zavolá vytvořené rozhraní , které má nadefinovány webové služby (detailně viz Kapitola 6.1.2). Pro tento účel je zavolána konkrétní webová služba, tedy **ParsedSearch.asmx**. Tato služba následně provede parsování vstupních dat na jednotlivé prvky, a pro tyto prvky se pak volá další služba - **ServiceIndexes.asmx**, která se

stará o interakci s příslušnými IR systémy. Nekomunikuje však napřímo, nýbrž přes vytvořený adaptér, který má právě za úkol přeložit daný příkaz pro příslušný IR systém. Adaptér tedy zavolá vybraný IR systém s příslušným parametrem (indexem).

V druhé fázi IR systém vygeneruje seznam výsledků (ResultList), který pak předává zpět na adaptér. Adaptér obdrží veškeré výsledky od IR systému, a z nich vybere pouze příslušné prvky (ID dokumentu, míru podobnosti a cestu ke zdrojovému dokumentu) a předává pouze tento výběr zpět na rozhraní. Zde se opět reverzně předávají výsledky přes ServiceIndexes do služby ParsedSearch, kde se výsledky vyhledávání shromáždí a dále se pak předávají směrem k webové aplikaci, kde se již provede vizualizace.

Ke správné vizualizaci (porovnání dokumentů side-by-side) je potřeba zavolat webovou službu **ServiceCollections.asmx** přes rozhraní. Tato služba předá webové aplikaci příslušný zdrojový dokument z datové kolekce.

6.1.1 Příprava kolekce dokumentů

Aby bylo možno pracovat s dokumenty a dostávat korektní výsledky, bylo nutno zpracovat jednotlivé kolekce dokumentů. Pro tyto účely byl vytvořen vlastní nástroj, který zpracuje a transformuje kolekce dokumentů na menší dílčí části (dokumenty, odstavce, věty).

Toto se provede následovně. Vybere se příslušná kolekce dat (viz Kapitola 7.1) a vybere se typ transformace dokumentu na dílčí část. V zadané kolekci se vyhledají všechny podadresáře a veškeré textové soubory, které se zde nachází. Podle zvoleného typu se následně jeden po druhém rozdělí na daný typ dílčích části (odstavce, věty...) pomocí regulárních výrazů. Následně pro bude pro každou takovou část vytvořen samostatný soubor, jehož název se generuje automaticky z názvu původního souboru a typu rozdělení (např. *kniha_sentences.txt*)(viz Obrázek 4).

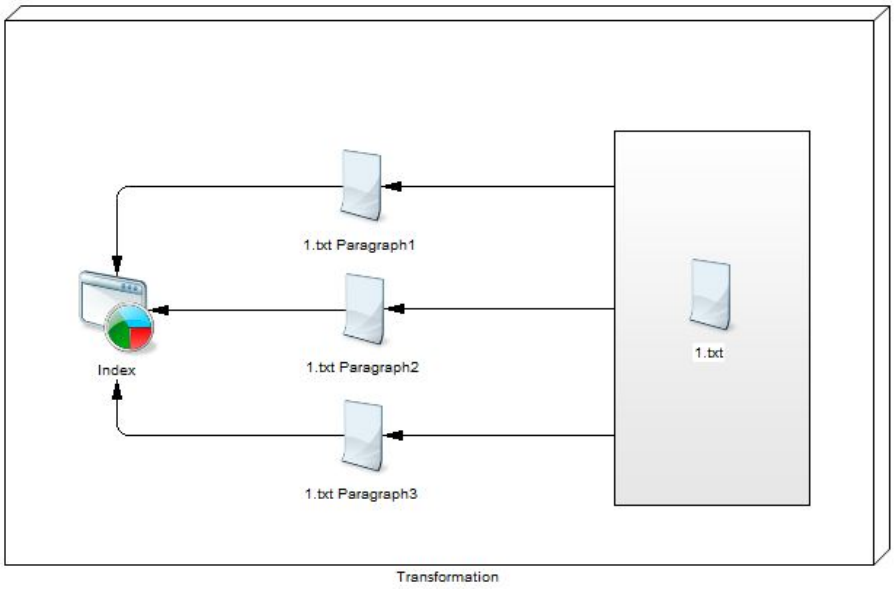
Kód a regulární výraz pro transformaci na věty (viz Výpis 1)

```
1 switch(m){
2     case ParseMode.Sentence:
3         string strMatch = @"(obr\.|např\.|popř\.|viz\.|resp\.| atd\.| ex\.|kap\.|sec\.|vol\.|č\.|
              tab\.|etc\.|i\.e\.| ap\.|tj\.| esp\.| diff\.|min\.|max\.|str\.|tzv\.| apod\.|e\.g\.|
              phdr\.|Bc\.| Ing\.|Doc\.|mgr\.|dis\.|Ph\.D\.|mba\.|Prof\.|scs\.|\d\.| Asp\.| .net|
              aspx\.|\.aspx\.|DrSc \.|[^.!?;]) +(([.]+\d+)+|[.!?;]+|$)";
4         for (Match mt = Regex.Match(text, strMatch); mt.Success; mt = mt.NextMatch())
5         {
6             block.Add(mt.Value);
7         }
8 }
```

Výpis 1: Kód a regulární výraz pro transformaci na věty

6.1.2 Návrh a implementace webových služeb

Webové služby se skládají ze 4 samostatných částí, všechny služby jsou implementovány v Microsoft .NET Framework verze 4. Každá z následujících webových služeb, jejich metody a způsoby jejich použití jsou důkladně popsány v příloze A:



Obrázek 4: Příprava kolekce dokumentů

- **ServiceCollections** slouží k získání informací o existujících kolekcích, vyhledání cesty ke konkrétnímu dokumentu a vrácení fyzického dokumentu.
- **ServiceIndexes** slouží k získání informací o dostupných indexech a vyhledávání v určeném indexu.
- **ServiceCompare** se používá k porovnání dokumentů a k vyhodnocení jejich podobnosti.
- **ParsedSearch** slouží pro rozdělení vstupních dat na menší části a vyhledávání těchto části v určeném indexu.

6.1.3 Aplikační vrstva

V současné chvíli aplikační vrstva obsahuje 2 základní položky. Windows Formuláře (WinForms) a samotnou webovou aplikaci. Windows Formuláře v podstatě obsahují pouze upravený IR systém Amphora. Webová aplikace pak poskytuje uživatelům možnost pracovat a vyhledávat informace v různých IR systémech (prostřednictvím webových služeb - rozhraní). Do aplikační vrstvy je možné v budoucnu zahrnout další návrhy uživatelských rozhraní a systémů jak pro vyhledávání specifických informací, tak pro vyhledávání plagiátů bez jakýchkoliv zásahů do systémové logiky.

6.1.4 Renovace GUI pro systém Amphora

Dílčím úkolem této práce bylo upravit grafické uživatelské rozhraní hotového IR systému Amphora. Toho bylo dosaženo návrhem WinForm aplikace s použitím rozhraní MDI. Win-

dows Form aplikace byla použita zejména jako vhodné uživatelské prostředí pro přehledné a intuitivní ovládání systému.

MDI (Multiple Document Interface) je prostředí vyvíjené pro Microsoft Windows a umožňuje vytvářet rozhraní pro aplikace, které uživatelům umožní pracovat s vícero dokumenty najednou. Můžeme si jej představit podobně, jako známe pracovní plochu operačního systému Windows. Každý dokument je pak v odděleném prostoru s vlastními ovládacími prvky pro pohodlné prohlížení. Uživatel tak může shlédnout a pracovat s různými dokumenty jako jsou tabulky, textové dokumenty či výkresy, a to pouhým přesouváním kurzoru s jednoho místa do druhého. Funkce, které systém Amphora obsahuje (původní funkce zachovány):

- vytváření indexů,
- indexace kolekce dat,
- otevření indexu,
- vyhledávání v indexech,
- zobrazení nalezeného dokumentu,
- zobrazení jednotlivých částí dokumentů,
- uzavření a zrušení indexu,
- hledání pomocí rozšířeného dotazu,
- výběr z možností dotazování (OR, AND apod.).

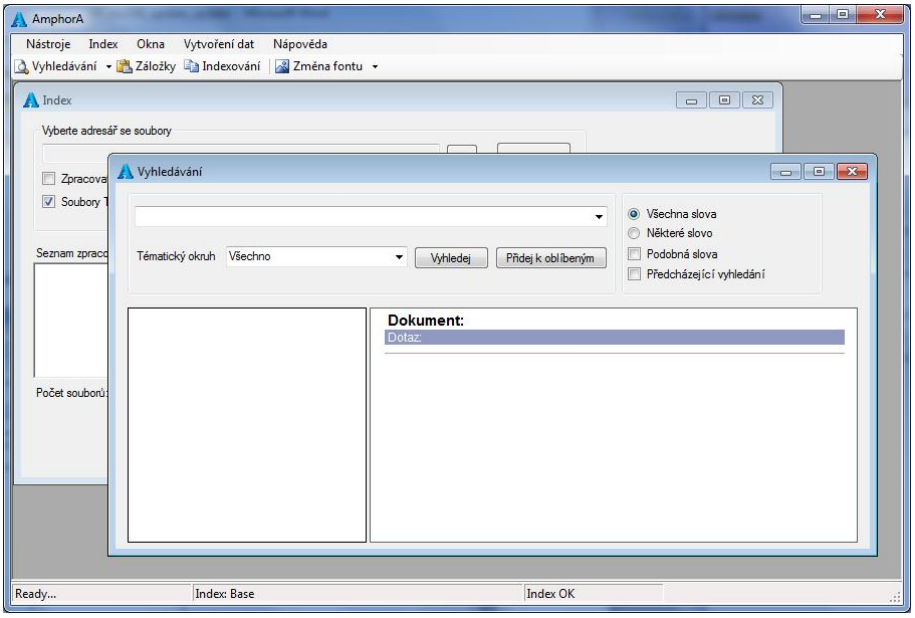
Dané uživatelské rozhraní systému Amphora pak vypadá následovně(viz Obrázek 5)

6.1.5 Návrh a implementace webové aplikace

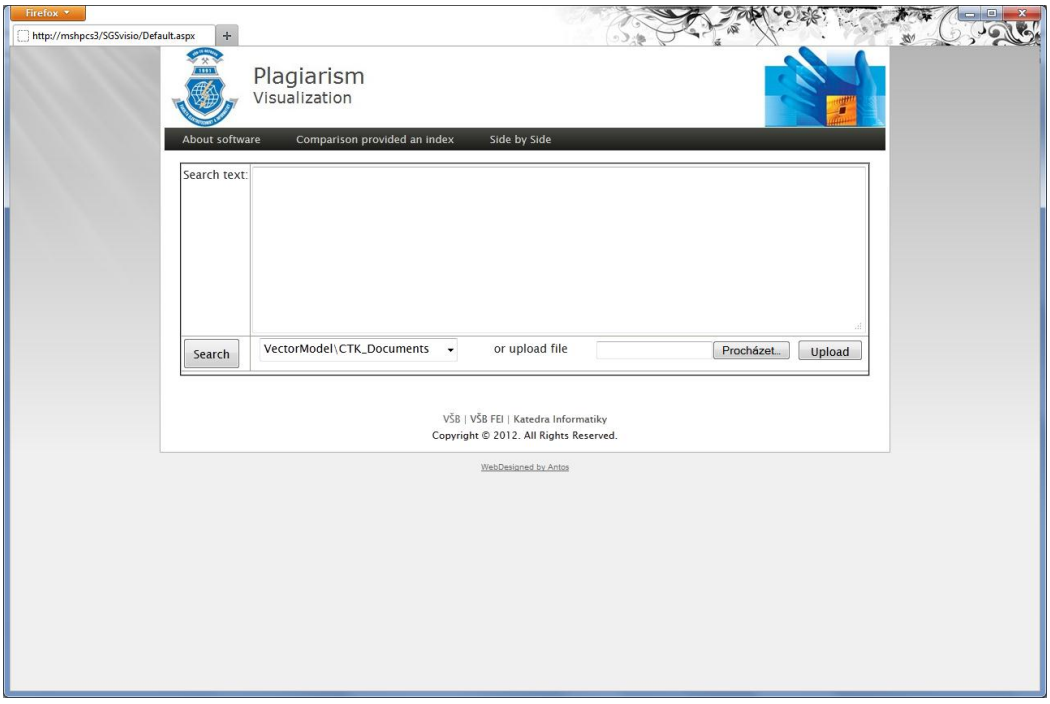
Pro vytvoření webové aplikace byla využita technologie ASP .NET, která v současné době patří mezi nejmodernější, robustní a velmi podporovanou, a zprostředkovává tak programátorovi téměř neomezené možnosti. Prostředí webové aplikace můžete vidět na Obrázku 6. Nyní si ale popíšeme jednotlivé funkce této aplikace.

Aplikace byla navržena tak, aby uživateli umožňovala vkládat celou práci výběrem práce z lokálního úložiště uživatele (a následně byla automaticky zpracována), anebo vkládat textovou část přímo prostřednictvím text-boxu. Potom co je tento dokument načten, má uživatel možnost vybrat index, se kterým chce pracovat a následně i příslušnou datovou kolekci k porovnání.

Po provedení dotazu a vyhledání v systému se uživateli provede vizualizace výsledku a bude mu nabídnuta možnost vybrat jednotlivé nalezené prvky k porovnání. Pro tyto účely byly použity prvky ImageButton, které jsou dynamicky generovány podle vrácených výsledků.



Obrázek 5: Uživatelské rozhraní Amphora

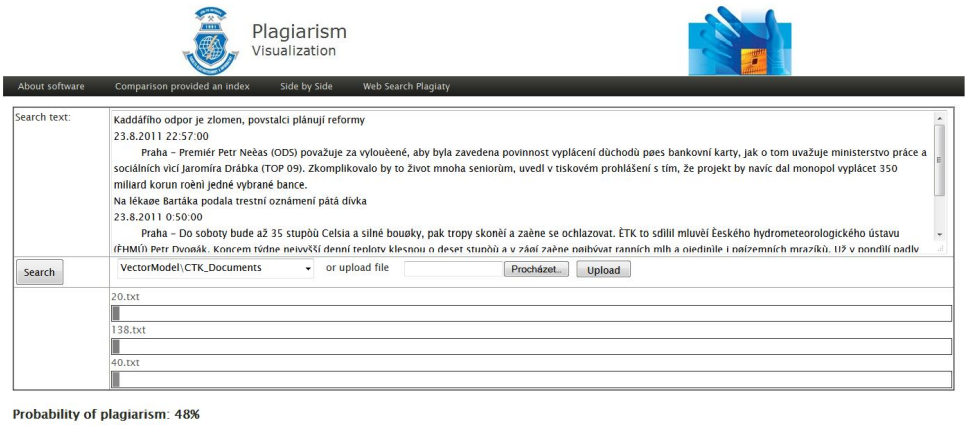


Obrázek 6: Uživatelské rozhraní webové aplikace

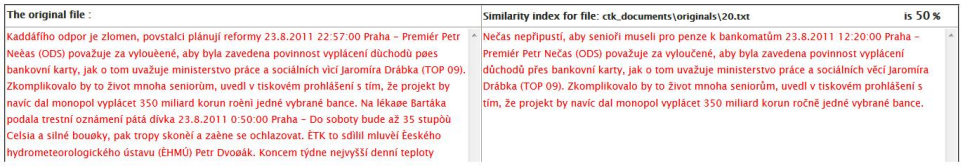
6.2 Návrh vhodné reprezentace výsledku

Samotnou vizualizaci obdržených výsledků je možné zobrazit více způsoby, v rámci této práce byly vybrány 3 způsoby vizualizace:

- Uživateli je zobrazen výsledek ve formě souhrnného zobrazení procentuální podobnosti. V praxi to znamená, že pokud je práce rozdělena na jednotlivé prvky, pak celková pravděpodobnost dokumentů je vypočítána jako průměrná podobnost nejpravděpodobnějších nálezů jednotlivých částí. Jednoduše, uživateli je výsledek zobrazen obecně ve formě procentuální celkové úspěšnosti. (viz Obrázek 7) neboli jejich jednotlivých části (viz Obrázek 8).

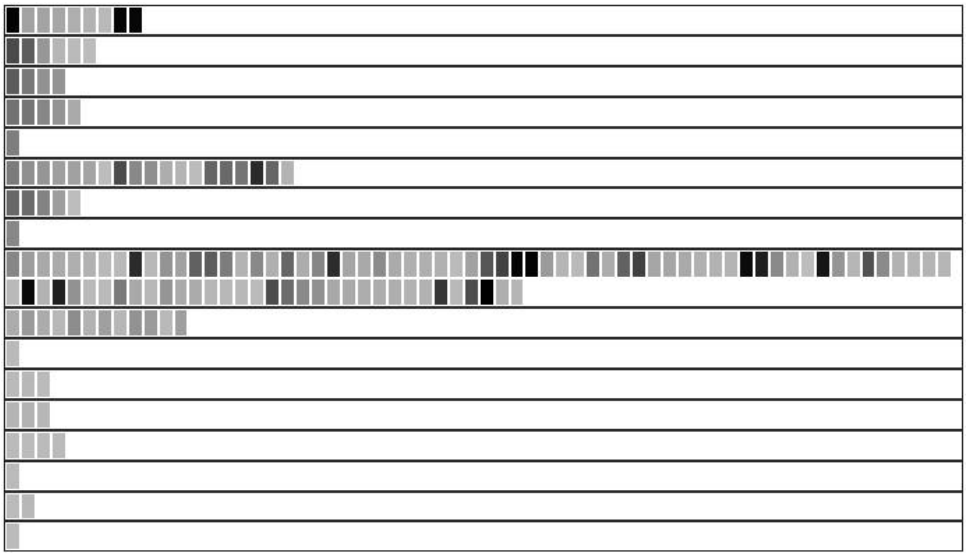


Obrázek 7: Vyhodnocování podobnosti



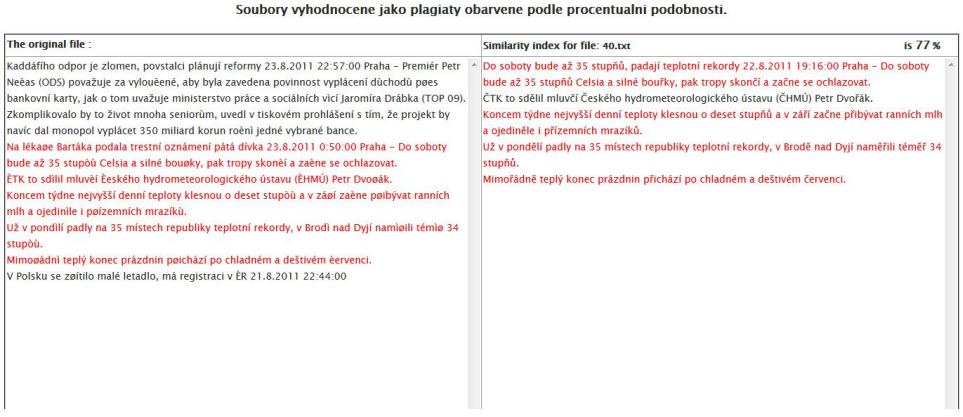
Obrázek 8: Porovnání jednotlivých nálezů s vyhodnocením podobnosti

- Druhým způsobem je zobrazení výsledku vyhledávání ve formě tzv. teplotní mapy, která představuje stínování jednotlivých pasáží v dokumentu podle toho, jak moc jsou podobné zdroji. Pro obarvení těchto části je použito 256 odstínů šedé barvy (viz Obrázek 9). Každý horizontální blok pak reprezentuje zdrojový dokument, se kterým byly nalezeny shody. Jednotlivé prvky pak odpovídají zvoleným dílčím částím, podle kterých byl dokument rozdělen.



Obrázek 9: Vizualizace nalezených částí pomocí teplotní mapy

- Poslední možností je nechat si zobrazit výsledky vyhledávání ve formě zadaného a nalezeného dokumentu vedle sebe (side-by-side forma - viz Obrázek 10), respektive je možné nechat si zobrazit přímo dané pasáže, které byly označeny jako shodné (viz Obrázek 8). Uživatel tak má možnost přehledně shlédnout veškeré nálezy v textu v dokumentech vedle sebe a může tak nejlépe určit, zda se opravdu jedná o plagiát, či chybu vyhledávání.



Obrázek 10: Porovnání dvou dokumentů

7 Experimenty

Obsahem této kapitoly je popis a hodnocení vlastní praktické části, experimentů realizovaných na základě výše popsané teorie. Dále zde naleznete hodnocení výsledků řešení vizualizace na základě teplotních map a jednotlivé datové kolekce, nad kterými se experimenty prováděly.

7.1 Použité kolekce dat

První kolekce dat pro experimentování a realizaci vyhledávání plagiátů byla sestavena tak, že do ní byly zahrnuty články a dokumenty z portálu ČTK, které obsahovaly přibližně 950 souborů. Po zpracování posléze obsahovala 5 508 odstavců a 11 913 vět.

Další kolekce obsahovala originální práce z PAN2010 vlastnící bakalářské, diplomové a disertační práce z Vysoké školy báňské - Technické Univerzity v Ostravě. Pro vytvoření kolekce bylo nejprve nutné převést všechny práce z různých textových formátů do formy plain-text. Tato kolekce se sestavovala z 11 148 souborů, přičemž po zpracování následně obsahovala 4 146 989 odstavců a 17 431 840 vět. Kolekci v číslech pak znázorňuje přehledně (viz Tabulka 4).

Plagiáty, na kterých se pak prováděly analýzy, byly sestaveny tak, že se různými způsoby uměle sestavily dokumenty náhodným zkopírováním odstavců z různých prací (metodou copy & paste), případně byly mírně upraveny.

	Dokumenty	*Odstavcy*	*Vety*	*Velikost (mb)*
ČTK	950	5 508	11 913	1.44
PAN2010	11 148	4 146 989	17 431 840	1 658.88

Tabulka 4: Rozsah kolekce

7.2 Použité kolekce plagiovaných dokumentu

Kolekce ČTK plagiátu (celkový pocet 550) mají název ve tvaru ″query_number source_1, source_2, . . . , kde query_number oznacuje unikátní císlo plagiátu zacínající písmenem ″q″s následujícím petimístným císelným kódem. Identifikátor source_x je šestimístné unikátní císlo zdrojového dokumentu, z kterého byl plagiát vytvoren. Každý plagiát muže být vytvoren z jednoho i více zdroju. Textové dokumenty plagiátu byly vytvoreny manuálne studenty dle stanovaných kritérií. Výsledný text je získán použitím jednoho ci více zdroju a vložením vlastních myšlenek.

7.3 Postupy experimentu

Před samotnými pokusy a vyhledáváním bylo zapotřebí zaindexovat příslušné kolekce pomocí vyhledávacích modulů. Pomocí každého z modulů se provedla indexace pro každý prvek z datových kolekcí, tedy pro jednotlivé dokumenty, odstavce a věty.

1. krok

Způsob vytvoreni	*Cca (%)*
Výměna odstavců, vet, několik po sobě následujících slov	40
Smazání celé vety, její části nebo jednoho slova	25
Výměna jednoho slova nebo náhrada synonymem	15
Vložení vlastních slov pro napojení významu vety	10

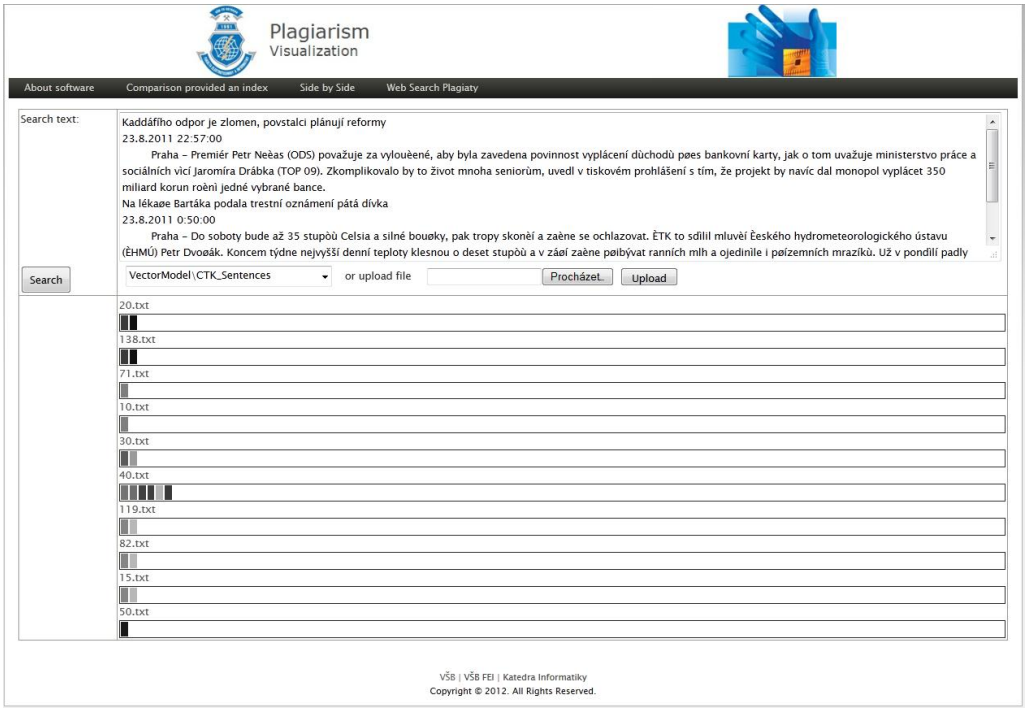
Tabulka 5: Způsob vytvořeni plagiátu

První experiment se týkal kontroly správnosti transformace kolekce (parsování) na dílčí části. Z důvodů časové náročnosti a rychlosti ověření byla zvolena pouze menší část kolekce. Kontrola byla prováděna na pěti dokumentech z každé kolekce a zjišťovalo se, zda je transformace prováděna správně na dokumenty, odstavce a věty. Tento krok je velice důležitý pro pozdější práci s dokumenty a částmi dokumentů, pokud by tento proces neproběhl správně, jednalo by se o velkou komplikaci a nemožnost realizace dalšího kroku. Při prvním pokusu bylo použito vyhledávání pomocí určitých symbolů v textu, tato metoda se však ukázala jako efektivní pouze pro dobře strukturované dokumenty. Jelikož kolekce dokumentů, se kterými se pracovalo, byly převedeny z různých formátů a jejich struktura nebyla vždy správně reprezentována, ihned po prvním testování se ukázala tato metoda jako neúčinná. Mnohem efektivnější způsob s příslušnými kolekcemi bylo využití regulárních výrazů pro lepší rozdělení jednotlivých částí. Těmito výrazy je možné přesně definovat, jak mají jednotlivé části vypadat.

2. krok
   Druhý experiment spočíval v testování správné funkčnosti systému a správnosti nalezených dat. Tento experiment byl rozdělen na 3 části testování podle parametru vyhledávání. Pro zjištění kvality vyhledávaní byli použity míry přesnosti (Precision) a úplnosti (Recall). Pro vypočet byli použity vytvořené kolekce plagiovaných dokumentu popsaných v Kapitole 7.2. Z následujících obrázku je patrné, že vytvořený systém byl schopen naleznout shodu pro dané plagiáty, a umožnil tak uživateli porovnat podezřelé části. Dále v kapitole 7.4 jsou znázorněny výsledky experimentu.

7.4 Výsledky experimentu

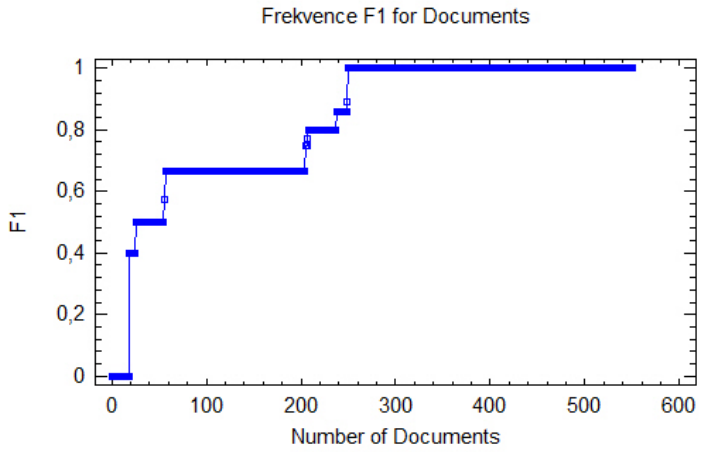
Ukázalo se, že metoda vyhledávání podle celých dokumentů závislá na rozsahu dokumentu, znamená to tedy, že čím rozsáhlejší dokument, tím nižší šance nalézt shodu, pokud autor používal různé zdroje při vytváření plagiátu. Lepší metodou je podle jednotlivých odstavců, tato metoda je zaručeně účinnější v případě rozsáhlých textů a datových kolekcí, ale i v případě krátkých článků jako je tomu u kolekce ČTK (viz Priloha B). Zaručeně nejpřesnější je pak vyhledávání podle jednotlivých vět, ovšem tady je zase problémem rychlost vyhledávání, jelikož takové vyhledávání je mnohonásobně náročnější na výpočetní dobu. Pokud se však jedná o velice krátké věty, je pak velká pravděpodobnost, že se nemusí jednat o plagiát, nýbrž o chybu v hledání.



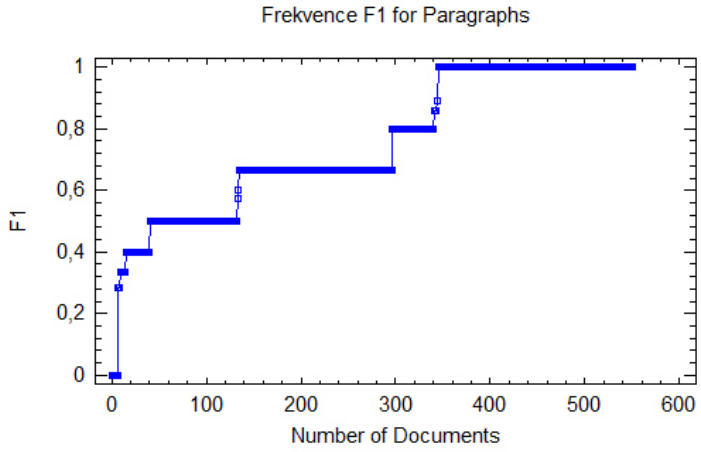
Obrázek 11: Výsledky hledání v ČTK kolekce

	Precision	*Recall*	*f1*
Dokumenty	0.844	0.869	0.857
Paragrafy	0.730	0.811	0.798
Sentences	0.374	0.924	0.533

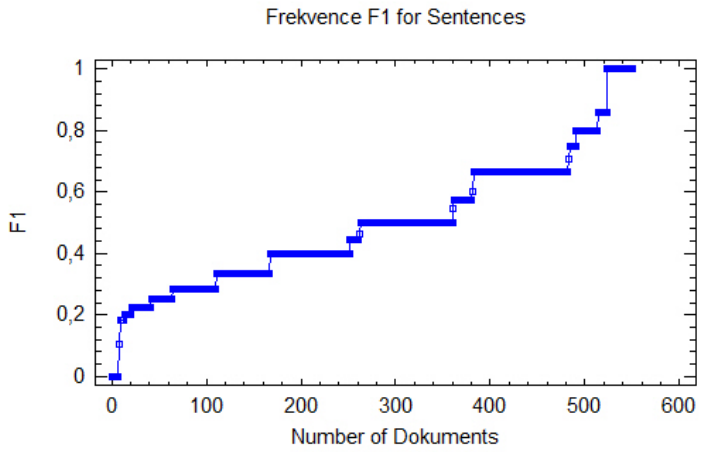
Tabulka 6: Výsledky experimentu



Obrázek 12: F míra ČTK dokumentu



Obrázek 13: F míra ČTK odstavců



Obrázek 14: F míra ČTK vět

8 Závěr

Cílem této diplomové práce bylo navrhnout prototyp plagiátorského systému, porovnat současné metody a algoritmy pro detekci plagiátů a pro zpracování textu. V teoretické části byly tedy popsány jednotlivé přístupy k detekci plagiovaných dokumentů, což úzce souvisí s různými metodami pro před-zpracovávání textů a také s indexací jednotlivých prvků kolekce pro možnosti efektivního vyhledávání. Dále byly popsány jednotlivé modely vyhledávání v rozsáhlých datových kolekcích a jejich možnosti. Jednalo se především o vektorový, pravděpodobnostní model, nebo například o n-gramy či LSH.

Jedna z dalších části práce byla o optimalizace systému vyhledávání. Moduly dodané vedoucím diplomové práci byli rozšířeni o možnost indexaci odstavců a vet bez zásahů do struktury modulů. V rámci této části byly navrhnuty a realizovány nástroje pro rozdělení celých dokumentu na jednotlivé prvky, vůči kterým se dalo otestovat chování IR systému.

V praktické části byl navržen a realizován systém, který by umožňoval propojení různých aplikací pro vizualizace z několika vyhledávacích modulů. Dále byla navržena testovací webová aplikace, která může být využita k vyhledávání plagiátů v rozsáhlých datových kolekcích a podporuje několik druhů vizualizací, které vhodně napomáhají k detekci a analýze výsledků podezřelých pasáží.

Jedním z možných cílů pak také bylo vytvořit systém takový, který by našel využití u dalších uživatelů, společností či institucí. Jinými slovy, navrhnout a implementovat systém, který by obecně splňoval současné požadavky potencionálních zájemců o detekci plagiátů, jelikož se jedná o velice aktuální téma dnešní doby.

Systém by pak bylo do budoucna možné rozšířit o některé další nástavby, které by například umožňovaly provádět lepší optimalizaci s pomocí paralelizace dotazu na vyhledávací moduly, využít hardwarovou akceleraci nebo například využití distribuovaných výpočtů (počítačových klastrů). Dále by bylo možné systém rozšířit o další možnosti vizualizace a ovládání některých částí systému, či napojení na další moderní IR systémy.

V neposlední řadě by pak bylo možné využít systém nejen pro různé způsoby vizualizace nalezených plagiátů, ale také pro vyhledávání vztahů mezi dokumenty a jejich autory cestou vizualizace pomocí komplexních sítí či grafů.

9 Reference

[1] Alzahrani, S., Salim, N. & Abraham, A.: Understanding Plagiarism Linguistic Patterns , Textual Features and Detection Method, 2011.

[2] Baeza-Yates, Ricardo; Ribeiro-Neto, Berthier: Modern Information Retrieval, New York, 1999.

[3] Berry, M.W., Browne. M.: Understanding Search Engines, Siam, 1999.

[4] Can, F., Patton,: J. M. Change of writing style with time, 2004.

[5] Chris Fox, The Influence of Text Pre-processing on Plagiarism Detection, Colchester, 2009.

[6] Danielle S. McNamara, Max M. Louwerse, Philip M. McCarthy, Arthur C.: GraesserCoh-Metrix: Capturing Linguistic Features of Cohesion, Memphis.

[7] Georg PÄolzlbauer, Michael Dittenbach, Andreas Rauber: Gradient visualization of grouped component planes on the SOM lattice, Vienna, 2005.

[8] Ingo Steinwart, Andreas Christmann: Support Vector Machines, Springer, New York, 2008.

[9] J. Malcolm, P. C. R. Lane: Compare A Journal Of Comparative Education , 1-6, 2008.

[10] Jude Carroll, Jon Appleton: Plagiarism A Good Practice Guide, Oxford, 2001.

[11] Karp, Richard M., Rabin, Michael O.: Efficient randomized pattern-matching algorithms, 1987.

[12] Kullback, S., Leibler, R.A.: Annals of Mathematical Statistics, 1951.

[13] Lashkari, A.H., Mahdavi, F., Ghomi, V: A Boolean Model in Information Retrieval for Search Engines, 2009.

[14] Mario Zechner, Markus Muhr, Roman Kern, Michael Granitzer: External and Intrinsic Plagiarism Detection Using Vector Space Models, Graz, 2009.

[15] Pataki, M.: Plagiarism Detection and Document Chunking Methods, Budapest, 2003.

[16] Robertson S. E., C. J. van Rijsbergen, Porter M.F: Probabilistic model of indexing and searching, Kent, 1987.

[17] Roussopoulos N.D.: A semantic network model of data bases,Toronto, 1976.

[18] Salton G; McGill MJ: Introduction to modern information retrieval, McGraw-Hill, 1986.

[19] Thomas K. Landauer, Darrell Laham: An Introduction to Latent Semantic Analysis, Boulder, 1998.

[20] URL: http://www.infogram.cz/

[21] URL: http://nlp.stanford.edu/

[22] URL: http://computerworld.cz/

[23] Wei Dong, Zhe Wang, William Josephson, Moses Charikar, Kai Li: Modeling LSH for Performance Tuning,Princeton, 2008.

A Popis webových služeb a způsoby použití

Popis webové služby ServiceCollections.asmx

V této části jsou popsány jednotlivé metody webové služby ServiceCollection.asmx. Služba je implementována pomocí Microsoft .NET Framework. Pro vysvětlení některých věcí byl použit místo textového popisu ukázkový kód. Ukázkové kódy jsou v programovacím jazyce C#.

Metoda GetAllCollections

Metoda slouží k získání informací o existujících kolekcích. Výsledkem metody bude seznam existujících kolekcí a fyzické cesty k odpovídajícím složkám na disku.

```
1 <CollectionsInfo> List<Collections> </CollectionsInfo>
```

Výpis 2: Výstup

Kolekce typu Collections.

Funkce **GetAllCollections** reprezentuje WGS bod.

```
1 SGSService.Collections[] result =
2 (SGSService.Collections[]) ws.GetAllCollections();
3 foreach(SGSService.Collections c in result)
4 {
5 Console.WriteLine(c.CollectionID, c.CollectionsPatch);
6 }
```

Výpis 3: Příklad práce s výstupem

Metoda GetPath

Metoda slouží k dotazování indexu na cestu k danému dokumentu. Index obsahuje *DocumentID* a *CollectionID* v které se dokument nachází. Vrácena je fyzická cesta k danému dokumentu.

Vstup

```
1 <DokumentID> long </DokumentID>
```

Výpis 4: Číselný název dokumentů

```
1 <CollectionID> string </CollectionID>
```

Výpis 5: Název příslušné složky na disku

Výstup

```
1 <TxtFilePath> string </TxtFilePath >
```

Výpis 6: Řetězec obsahující fyzickou cestu k .txt dokumentu

Funkce *GetPath* reprezentuje WGS bod.

```
1 string result = ws.GetPath();
2 Console.WriteLine(result);
```

Výpis 7: Příklad práce s výstupem

Metoda ReturnFile

Dle zadaného *DocumentID* a *CollectionID* vrátí fyzický dokument nacházející se na disku.

Vstup

```
1 <DokumentID> long </DokumentID>
```

Výpis 8: Číselný název dokumentů

```
1 <CollectionID> string </CollectionID>
```

Výpis 9: Název příslušné složky na disku

Výstup

```
1 <TxtFilePath> byte[] </TxtFilePath >
```

Výpis 10: .txt soubor

Funkce *ReturnFile* reprezentuje WGS bod.

```
1 byte[] result = ws.ReturnFile(1, \textacutedbl CTK\textacutedbl);
2 File.WriteAllBytes(path, result);
```

Výpis 11: Příklad práce s výstupem

Popis webové služby ServiceIndexes.asmx

V této části jsou popsány jednotlivé metody webové služby ServiceIndexes.asmx. Služba je implementována pomocí Microsoft .NET Framework. Pro vysvětlení některých věcí byl použit místo textového popisu ukázkový kód. Ukázkové kódy jsou v programovacím jazyce C#.

Metoda FindeByID

Metoda slouží k vyhledání v určeném indexu dokumentu podobných danému. Vrácen je seznam dokumentu a úroveň jejich podobnosti.

Vstup

```
1 <DokumentID> long </DokumentID>
```

Výpis 12: Číselný název dokumentů

```
1 <CollectionID> string </CollectionID>
```

Výpis 13: Název příslušné složky na disku

```
1 <IndexName> string </IndexName>
```

Výpis 14: Název Indexu

Výstup

```
1 <ObjectSearchedList> List<ObjectSearched> </ObjectSearchedList>
```

Výpis 15: Kolekce typu ObjectSearched

Funkce *FindeByID* reprezentuje WGS bod.

```
1 SGSService.ObjectSearched[] result =
2  (SGSService.ObjectSearched[])ws.FindByID(1,\textacutedbl CTK\textacutedbl, \textacutedbl
       TestIndex\textacutedbl);
3 foreach(SGSService.ObjectSearched c in result)
4 {
5    Console.WriteLine(\textacutedbl{0} {1} {2}\textacutedbl, c.CollectionID, c.Id, c.Measure);
6 }
```

Výpis 16: Příklad práce s výstupem

Metoda FindeByText

Metoda slouží k hledání v indexu na základě zadaného textu. Vrácen je seznam dokumentu a úroveň jejich podobnosti.

Vstup

```
1 <Text> string </Text>
```

Výpis 17: Hledaný text

```
1 <IndexName> string </IndexName>
```

Výpis 18: Název Indexu

Výstup

```
1 <ObjectSearchedList> List<ObjectSearched> </ObjectSearchedList>
```

Výpis 19: Kolekce typu ObjectSearched

Funkce *FindeByText* reprezentuje WGS bod.

```
1 SGSService.ObjectSearched[] result =
2  (SGSService.ObjectSearched[])ws.FindByID(text, \textacutedbl TestIndex\textacutedbl);
3 foreach(SGSService.ObjectSearched c in result)
4 {
5    Console.WriteLine(\textacutedbl {0} {1} {2}\textacutedbl, c.CollectionID, c.Id, c.Measure);
6 }
```

Výpis 20: Příklad práce s výstupem

Metoda GetIndexes

Metoda slouží k získání informací o dostupných indexech. Vrácen je seznam všech indexů.

Výstup

```
1 <Indexes> List<string> </Indexes>
```

Výpis 21: Kolekce řetězců s názvy indexů

Funkce *GetIndexes* reprezentuje WGS bod.

```
1 string [] result = ws.GetIndexes();
2 foreach(string c in result)
```

```
3 {
4 Console.WriteLine(c);
5 }
```

Výpis 22: Příklad práce s výstupem

Popis webové služby ServiceCompare.asmx

V této čísti dokumentu jsou popsány jednotlivé metody webové služby ServiceCompare.asmx. Služba je implementována pomocí Microsoft .NET Framework. Pro vysvětlení některých věcí byl použit místo textového popisu ukázkový kód. Ukázkové kódy jsou v programovacím jazyce C#.

Metoda CompareTwoDocuments

Metoda slouží k porovnání dvou dokumentů. Vrátí třídu obsahující informace o podobnostech a společných částech.

Vstup

```
1 <DokumentID_1> long </DokumentID_1>
```

Výpis 23: Číselný název dokumentů

```
1 <CollectionID_1> string </CollectionID_1>
```

Výpis 24: Název příslušné složky na disku

```
1 <DokumentID_2> long </DokumentID_2>
```

Výpis 25: Číselný název dokumentů

```
1 <CollectionID_2> string </CollectionID_2>
```

Výpis 26: Název příslušné složky na disku

Výstup

```
1 <ResultCompare> SimilarityClass </ResultCompare>
```

Výpis 27: Instance objektu typu SimilarityClass

Metoda CompareDocuments

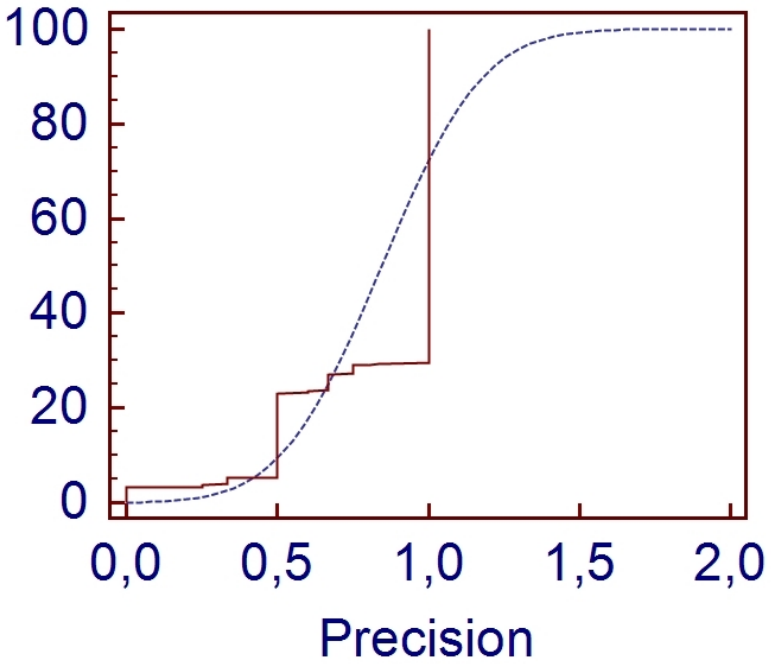
Metoda slouží k porovnání kolekce dokumentů. Vrátí matici podobností.

Výstup

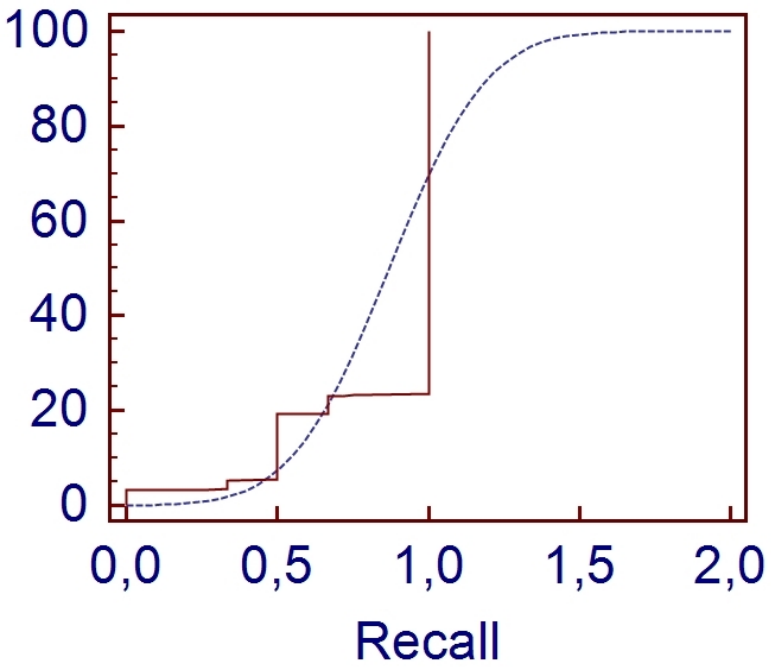
```
1 <SimilarityMatrix> string[,] </SimilarityMatrix>
```

Výpis 28: Matice podobnosti dokumentu

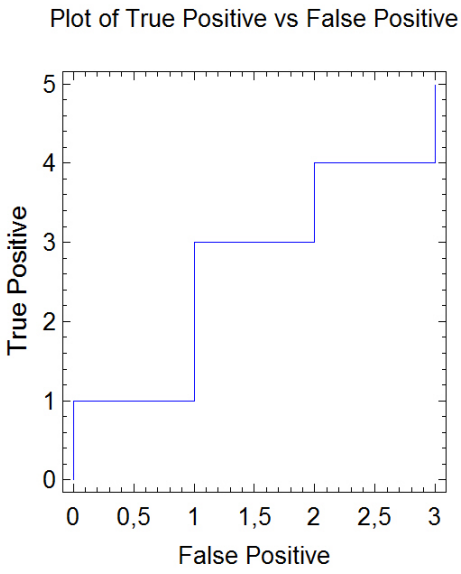
B Výsledky experimentu



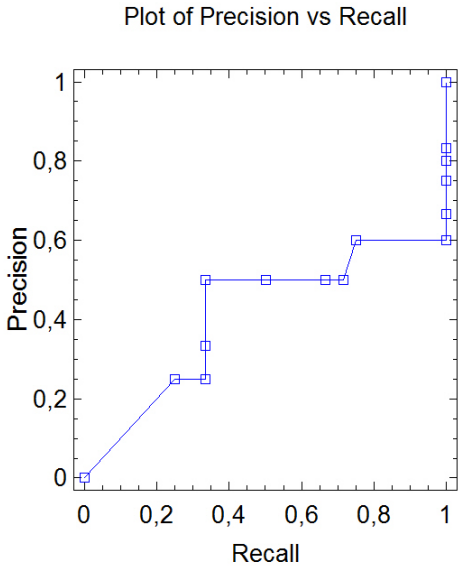
Obrázek 15: Frekvence precision ČTK dokumentu



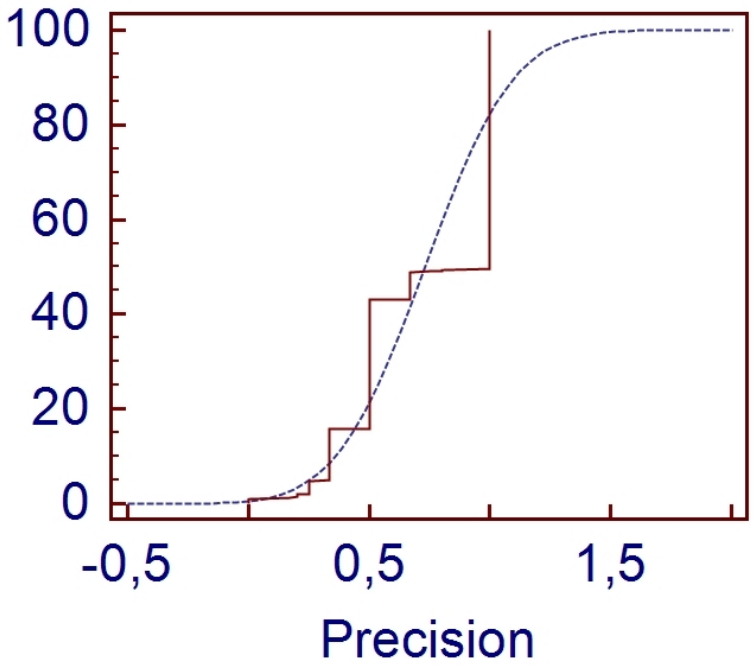
Obrázek 16: Frekvence recall ČTK dokumentu



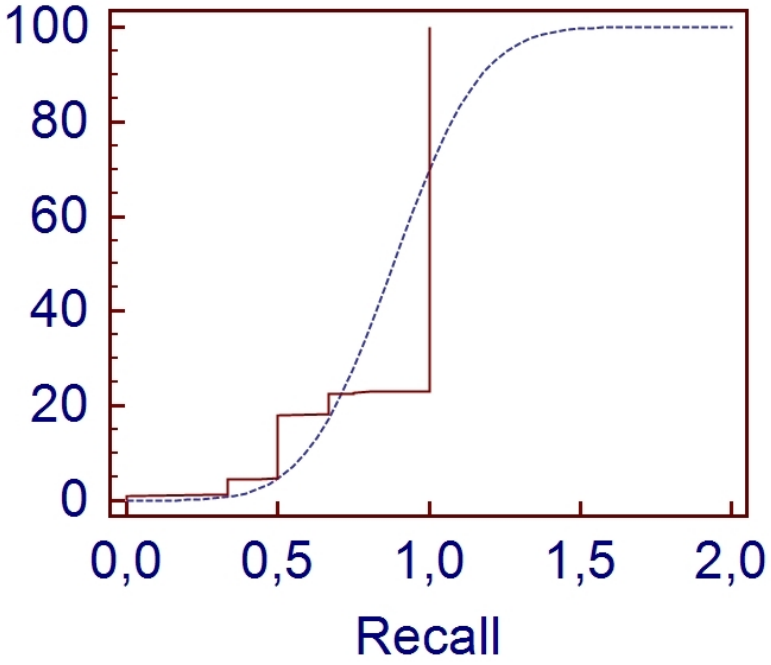
Obrázek 17: TP a FP ČTK dokumentu



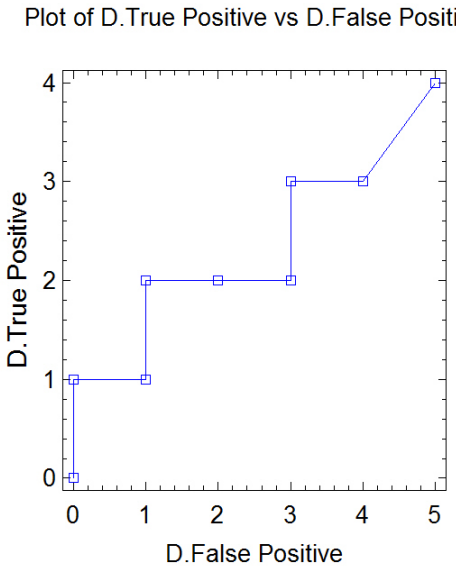
Obrázek 18: Precision/Recall ČTK dokumentu



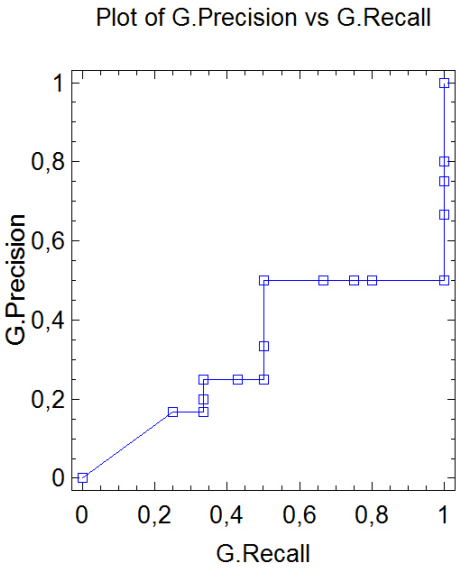
Obrázek 19: Frekvence precision ČTK odstavců



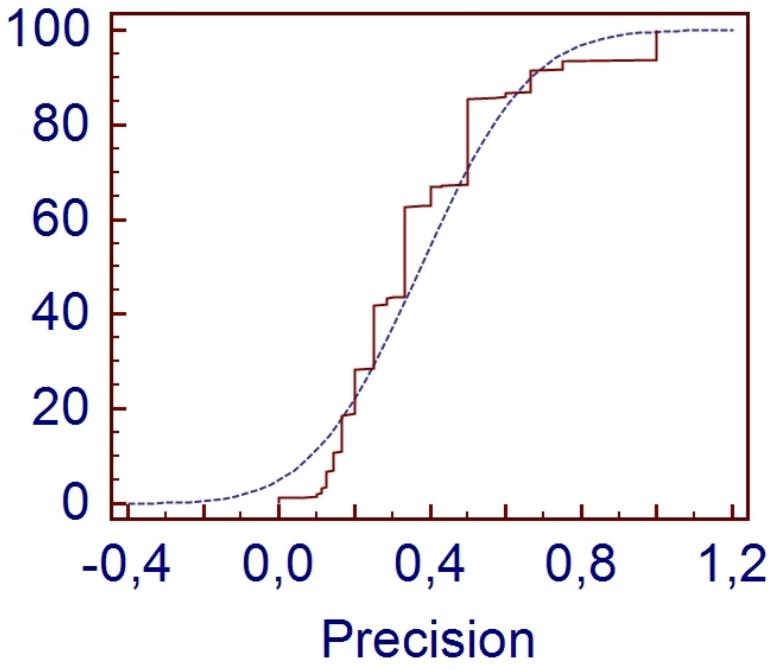
Obrázek 20: Frekvence recall ČTK odstavců



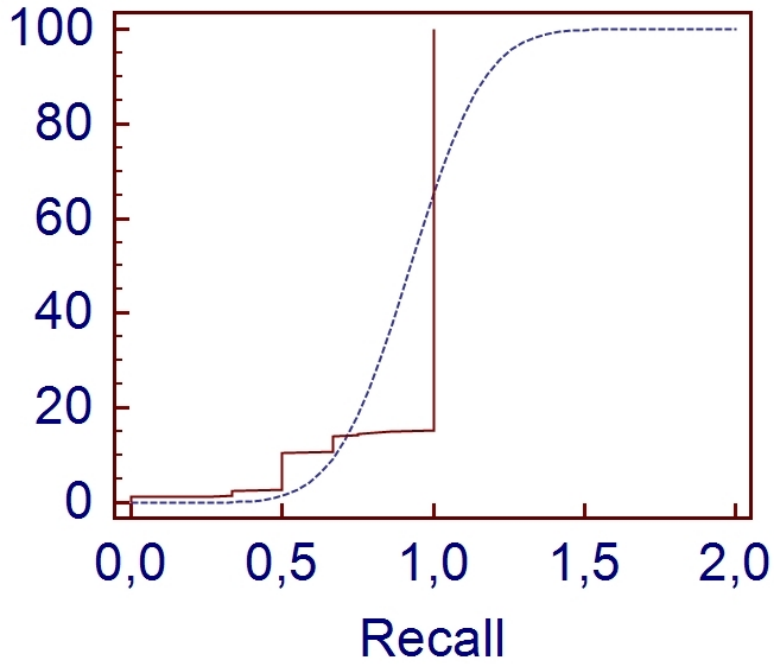
Obrázek 21: TP a FP ČTK odstavců



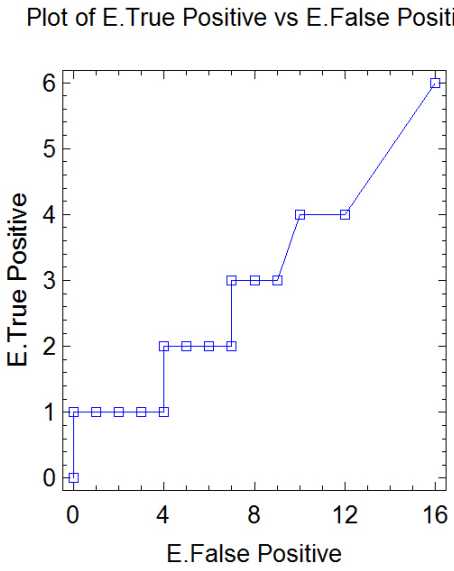
Obrázek 22: Precision/Recall ČTK odstavců



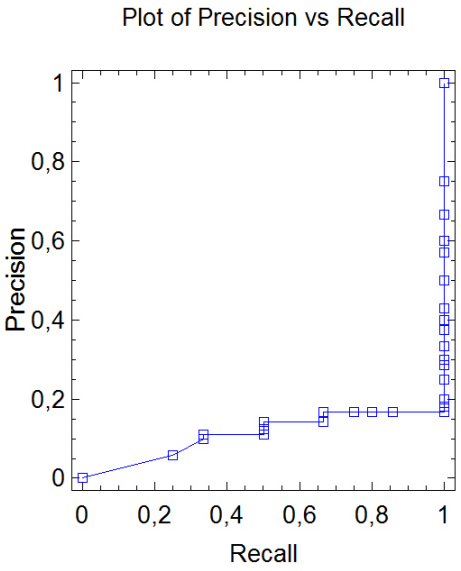
Obrázek 23: Frekvence precision ČTK vět



Obrázek 24: Frekvence recall ČTK vět



Obrázek 25: TP a FP ČTK vět



Obrázek 26: Precision/Recall ČTK vět