TECHNICKÁ UNIVERZITA V LIBERCI

Fakulty mechatroniky, informatiky a mezioborových studií

DIPLOMOVÁ PRÁCE

Liberec 2012 Michal Rott

TECHNICKÁ UNIVERZITA V LIBERCI

Fakulty mechatroniky, informatiky a mezioborových studií

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

Automatická sumarizace textových dokumentů

Automatic summarization of text documents

Diplomová práce

Autor: Bc. Michal Rott
Vedoucí práce: Ing. Petr Červa, Ph.D.
Konzultant: Ing. Ladislav Šeps

V Liberci 15. května 2012

!!! Originál zadání !!!

Místo tohoto listu bude originál zadání...

Prohlášení

Byl(a) jsem seznámen(a) 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(a) samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum

Podpis

Poděkování

Rád bych poděkoval vedoucímu mé diplomové práce panu Ing. Petru Červovi Ph.D. ze rady a čas, které mi věnoval během konzultací. Také bych rád poděkoval všem, kteří se účastnili tvorby databáze referenčních souhrnů.

Abstrakt

Dnešní svět je přehlcen informacemi a právě tato práce se snaží lidem usnadnit práci s informacemi vytvářením souhrnů těchto informací. V rámci výzkumu byly převážně z anglické literatury nastudovány metody vytvářející z rozsáhlých článků extrakty. Byly nastudovány sumarizační metody heuristické a statistické využívané v počátcích digitalizace textů, ale i moderní metody analyzující texty hlouběji. Hlavní pozornost byla věnována Luhnovu sumarizátoru a latentní sémantické analýze. Tyto metody byly také implementovány v jazyku C# na platformě Mono.

Druhá část diplomové práce řeší problematiku evaluace implementovaných sumarizačních metod. Z literatury a vědeckých článků byly nastudovány techniky používané pro měření a hodnocení automaticky generovaných souhrnů. Pro vlastní provedení evaluace byl využit program ROUGE, využívaný pro tento účel i na konferencích Text Analysis Conference. V rámci evaluace bylo provedeno několik experimentů s různými nastaveními sumarizace a byly vyhodnoceny i volně dostupné sumarizátory.

Klíčová slova

sumarizace, souhrn, Luhnův sumarizátor, Latentní sémantická analýza, evaluace, ROUGE

Abstract

Today's world is overloaded with information and this work is trying to help people work with information by creating summaries of this information. During the research has been staging method of producing extracts from large articles. Staging were summarization mehods heuristic and statistical used in the early days of text digitization and modern methods analyzing texts more deeply. The main attention was paid to Luhn summarizer and to method using latent semantic analysis. These methods were also implemented in C# on the Mono platform.

The second part of the thesis deals with the issue summarizing the evaluation of implemented methods. From literature and scientific articles have been staging techniques used for measurement and evaluation of automatically generated summaries. For the actual performance evaluation program was used ROUGE, used for that purpose at conferences and Text Analysis Conference. The evaluation was carried out several experiments with different settings and summaries have been evaluated and freely available sumarizátory.

Keywords

summarization, summary, Luhn summarizer, Latent semantic analysis, evaluation, ROUGE

Obsah

1 Úvod

S počátkem digitalizace textových dokumentů vznikl problém s nedostatečnou kapacitou datových úložišť. Tento problém se začal řešit vytvářením souhrnů dokumentů určených k uložení a katalogizací dokumentů podle nich. Pokud dokument již obsahoval souhrn, nebyl problém ho rychle zařadit. Tyto souhrny vytvářené převážně samotnými autory jsou označované jako abstrakty nebo resumé. Problém chybějících abstraktů se začal řešil automatickou sumarizací, jelikož nebylo možné vytvořit ručně souhrny pro "velké"množství dokumentů. Tyto automaticky vytvořené souhrny bylo pak možné využít pro vyhledávání dokumentů v knihovních terminálech a také se podle nich mohli rozhodnout, kterou knihu přečíst.

Dnes stojíme před opačným problémem. Kapacita datových úložišť je pro potřeby uložení textových dokumentů v podstatě neomezená. Například čtečka elektronických knih Amazon Kindle 3 má kapacitu 4GB a průměrná elektronická kniha má velikost přibližně půl megabytu. To znamená, že do této čtečky lze nahrát až 8000 knih. Takové množství knih není v podstatě možné přečíst a je nutné si podle nějakých informací vybrat jen knihy, které uživatele zajímají. Ve světě internetu je tento problém mnohonásobně větší. Denně vznikají tisíce dokumentů, článků, zpráv a zápisků na blozích. Ze všech těchto dokumentů si uživatelé vybírají jen ty, které považují za důležité. Pro rozhodnutí, který článek je pro nás důležitý, můžeme využít souhrn, jenž nám pomůže indikovat, který článek stojí za přečtení. Za extrémní formu souhrnu můžeme považovat třeba nadpis.

S různými typy souhrnů se setkáváme v podstatě denně. Například při vyhledávání informací pomocí služeb Googlu si vybíráme odkazy, právě na základě souhrnu stránky, který pro nás vyhledávací služba vytváří podle zadaného dotazu. Tento typ souhrnu se označuje jakou souhrn na dotaz. Dalším příkladem souhrnů jsou "headlines"čtené moderátory televizních zpráv. Tyto souhrny představují dva odlišné přístupy k vytváření souhrnů. Jeden vytvářen výpočetní technikou a druhý člověkem. A právě k myšlení člověka se při vytváření souhrnu snaží co nejvíce přiblížit metody automatické sumarizace.

1.1 Automatická sumarizace

Automatická sumarizace je lingvistická disciplína, jejímž cílem je vytváření co nejlepších souhrnů. Souhrn dokumentu nás pak informuje o původním dokumentu a jeho informační hodnotě. Tyto souhrny jsou vytvářeny na základě dvou různých základních principů. Metody automatické sumarizace vytváří buď abstrakty nebo extrakty. V rámci

diplomové práce se budu věnovat převážně metodám vytvářejícím extrakt, jelikož vytváření abstraktů, zvláště pro český jazyk, je velmi komplexní disciplína, která vyžaduje tým odborníků z oblasti syntaxe a morfologie jazyka. Navíc většina světového výzkumu v oblasti vytváření souhrnů se zaměřuje právě na extrakci vět.

1.2 Sumarizace dnes

Díky zvyšujícímu se výkonu hardwaru se dnes sumarizace odklání od statistických metod, které jsou méně náročné na výpočetní výkon a začínají se čím dál více používat metody využívající hlubších lingvistických znalostí. Metody jako jsou například grafová metoda, metoda využívající teorii rétorických struktur nebo latentní sémantická analýza dnes získávají na významu a jsou stále více využívány. Tyto metody zkoumají vazby mezi jednotlivými prvky vět nebo i celými větami a na základě této analýzy vytvářejí souhrny. Principy statistických metod však neupadly v zapomnění, jsou často využívány v rámci jiných metod.

V současné době je vytvořeno mnoho sumarizátorů, některé pro komerční účely a jiné pro vědecké, jejichž smyslem je výzkum nových přístupů k sumarizaci. Příkladem komerčního více dokumentového sumarizátoru může být `<http://www.news-articles.org>`, který vyhledává na stránkách internetových periodik aktuální dění ve světě a zprávy, jenž vyhodnotí jako nejdůležitější zobrazí na svých stránkách. Dalším příkladem podobné služby jsou `<http://news.google.com>`, kde jsou i extrahované tzv. "Top stories", tedy události, o kterých se píše ve světě nejvíce. Na webu lze nalézt také velké množství online textových sumarizátorů, ale málokterý si dokáže poradit s češtinou a jejich výsledky jsou málo kvalitní.

V České republice se oboru automatické sumarizace intenzivně věnují hlavně na Západočeské univerzitě, kde se využívají převážně metody založené na latentní sémantické analýze. Vznikl například multidokumentový sumarizátor založený na LSA [10] nebo projekt ALMUS [23], který vytváří i aktualizační souhrny. Členové ZČU byly také jediní čeští účastníci konference Text Analysis Conference. Tyto konference organizuje od roku 2000 National Institut of Standards and Technology a udávají směr vývoje sumarizací a jejich evaluací. Momentálně je výzkum sumarizace zaměřen na multidokumentovou a aktualizační sumarizaci.

2 Rozdělení souhrnů a metod sumarizací

Před začátkem této kapitoly považuji za nutné definovat některé důležité pojmy, které budou v této a dalších kapitolách použity.

Souhrn - text obsahující důležité informace z rozsáhlého dokumentu
Sumarizace - proces vytvářející souhrn
Sumarizátor - systém realizující alespoň jednu metodu sumarizace
Term - prvek nebo prvky textu označující jeden objekt, činnost, jev, ...

2.1 Členění dle typy souhrnu

Základní možnost, jak rozdělit proces sumarizace, je podle formy jejího výstupu, tedy souhrnu. Podle formy souhrnu se sumarizace dělí na metody vytvářející extrakty a na metody vytvářející abstrakty.

Metody založené na principu **sumarizace generováním** se snaží vytvořit ze vstupního článku abstrakt tak, jak ho známe z různých dokumentů. Autor vlastními slovy popíše, o čem dokument pojednává a výsledný abstrakt pak slouží potenciálním čtenářům k indikaci užitečnosti dokumentu pro jejich potřebu. Metody automatické sumarizace vytvářející abstrakt jsou dnes v plenkách. Bylo navrženo několik metod, které se pokoušejí vytvářet abstrakt, ale tyto abstrakty nedosahují kvality ručně psaných abstraktů a v praxi nejsou moc preferované. Metody vytvářející abstrakt je také velmi těžké implementovat, jelikož jejich implementace zahrnuje komplexní znalosti z oblasti morfologie a syntaxe jazyka. Implementace je také ztížena díky ohebnosti cílového jazyka a abstrakty pro velmi ohebné jazyky, jako je čeština, jsou velmi často nečitelné a špatně podchycují abstrahovanou informaci.

Druhým principem je **sumarizace extrahováním**, tedy vytváření souhrnů na základě extrakce vět z původního textu sumarizovaného dokumentu. Metody vytvářející extrakt jsou dnes velmi oblíbené. Věty extraktu neztrácejí oproti automaticky generovanému abstraktu kontext informací, který je vyjádřený tvarem věty. Extrakty ovšem ztrácejí význam informace, převážně kvůli vytržení věty z jejího kontextu. Například ve větě se nacházející zájmeno *on* může po extrakci věty do výsledného extraktu poukazovat na jiného muže než v původním textu. Pomocí vět v extraktu lze ovšem jednoduše odkazovat do původního dokumentu přímo na větu a odstavec, kde se nacházejí. Toto u abstraktu nelze, jelikož abstrakt obsahuje informace z několika vět složených do jedné

věty. Díky extrahování věty z původního textu také odpadá celý proces umělého vytvoření nové věty.

2.2 Členění dle úrovně analýzy dokumentu

Dalším velmi důležitým rozdělením je rozdělení podle úrovně analýzy původního textu. Takto se dělí sumarizace na sumarizace s povrchním přístupem a na sumarizace s hlubšími přístupy k sumarizovaným informacím.

Metody využívající **povrchní přístupy** k sumarizaci jsou metody využívající zjevné jevy v sumarizovaném textu. Příkladem těchto jevů je například frekvence výskytu termů, specifické termy pro určitou doménu, slova zvyšující význam termů věty nebo pozičně významné termy. Povrchní metody jsou využívány jen k vytváření extraktu, jelikož nedokáží určit vztahy mezi termy dokumentu, ale jen jejich významnost v rámci věty a celku. Tyto metody lze také velmi dobře využít pro sumarizaci na dotaz. Povrchní přístup k sumarizaci využívají například heuristické a statistické metody sumarizace.

Hlubší přístupy k sumarizaci dokáží určit vztahy mezi termy, jejich důležitost a význam. Proto je lze využít nejen pro vytváření extraktu, ale i abstraktu. Metody s hlubšími přístupy využívají lingvistických znalostí o textu, text analyzují a na základě toho text hodnotí. Příkladem takovéto metody je metoda využívající teorie rétorických struktur, která vytváří RS-strom, který zachycuje vztahy mezi jednotlivými částmi textu. RST metodu implementuje například systém popsaný v [17]. Dalšími metodami jsou například grafové metody, které zkoumají relace podobnosti vět, nebo latentní sémantická analýza, která problematiku sumarizace převádí na algebraickou úlohu dekompozice matic.

2.3 Členění dle potřeb uživatelů

Sumarizace lze také rozdělit podle potřeb a zaměření uživatelů. Uživatelé často potřebují, aby souhrny byly generovány dle jejich zaměření nebo dle jejich dotazu. Sumarizace lze podle tohoto rozdělit na sumarizace obecné, na dotaz nebo doménové. **Obecné sumarizace** nevyužívají žádných požadavků uživatele na sumarizaci. Uživatel tyto sumarizace nijak nepřizpůsobuje pro své potřeby a pro všechny uživatele má tento typ sumarizace stejný výsledek. Opakem jsou **sumarizace na dotaz**. Od těchto sumarizací očekává člověk souhrn, který bude obsahovat konkrétní hledané informace. Příkladem takovéto sumarizace je souhrn generovaný vyhledávací službou firmy Google. Souhrn webové stránky, uvedený pod odkazem na ni, zobrazuje vyhledávanou informaci uvnitř

stránky spolu s jejím bezprostředním okolím. Tento vygenerovaný souhrn závisí přímo na dotazu, který uživatel zadal. Třetím typem uživatelské sumarizace je vytváření souhrnu na základě definované **domény**, tedy oblasti, která uživatele zajímá. Pokud by článek obsahoval témata z oblasti politiky, ekonomie a potravinářství, pak by uživatel, který by si zvolil téma ekonomie, získal jiný souhrn než uživatel, kterého zajímá více politika.

2.4 Členění dle rozsahu souhrnu

Důležitý parametr souhrnu je jeho rozsah. Pokud se chce uživatel informovat o problematice článku, potřebuje souhrn s větším rozsahem, než pokud ho zajímá jen téma článku, pak je velký rozsah souhrnu zbytečný. Podle tohoto kritéria je možno rozdělit souhrny na indikativní, informativní a hodnotící. Toto rozdělení je založeno na kompresním poměru sumarizace a bylo poprvé použito v práci [4]. Výpočet kompresního poměru (ang. compression ratio) je vyjádřen jako podíl délky souhrnu ku délce původního dokumentu.

Hodnotící souhrny jsou souhrny, které počítač nedokáže vygenerovat. Jedná se například o recenze, preview a kritiky. Tyto souhrny mají velmi velký rozsah a jsou vytvářeny lidmi, jenž mají odborné znalosti z oblasti, kterou se původní dokument zabývá. Tímto se do hodnotící sumarizace, mimo děje knihy nebo problematiky vědeckého článku, dostanou i zkušenosti, názory a znalosti tvůrce souhrnu. V podstatě můžeme říci, že vznikl úplně nový dokument o jiném dokumentu.

Souhrny, jenž mají uživatele informovat, zda dokument číst či ne, jsou označovány jako **indikativní**. Jsou to souhrny přinášející uživateli nezbytné minimum informací, podle kterého se rozhoduje, jestli si přečte celý dokument a jestli je jeho téma pro něj důležité. Tyto souhrny mají rozsah maximálně do 10 % rozsahu původního textu dokumentu. Příkladem takovéto sumarizace jsou například již zmíněné headlines, které nás informují o tématu zprávy, ale neseznámí nás s jejími detaily.

Detailnější informace o dokumentu poskytuje souhrn **informativní**. Informativní souhrny mají rozsah od 20 % do 30 % původního textu. Takovýto rozsah už dostačuje k plnému porozumění problematiky, o které dokument pojednává a uživatel by měl po přečtení informativního souhrnu rozumět problematice dokumentu stejně, jako kdyby si přečetl celý dokument.

2.5 Multidokumentová, aktualizační a ASR sumarizace

Dalším zajímavým typem sumarizace je vytváření **aktualizačního souhrnu**. Při vytváření aktualizačního souhrnu je definována množina znalostí uživatele (například seznamem přečtených dokumentů) a souhrn je vytvářen tak, aby množina informací v souhrnu neobsahovala uživateli již známé informace. Můžeme tedy říci, že klasická textová sumarizace je aktualizační sumarizace pro prázdnou množinu znalostí.

Problémy přehlcení informacemi vznikajícími ve světě internetu řeší **multidokumentová sumarizace**. Tato sumarizační disciplína zjednodušuje uživatelům práci vytvářením souhrnů z více článků do jednoho souhrnu. Při inicializaci procesu sumarizace je z dokumentů určených k sumarizaci vytvořen velký korpus. Z tohoto korpusu jsou následně vybírány informace tak, aby každá nově vybraná informace neobsahovala již dříve vybrané informace. Multidokumentová sumarizace je velmi podobná aktualizační sumarizaci, jen množina již známých informací je rozšiřována s každou novou vybranou větou. Následující vzorec (1) popisuje výpočet podobnosti dvou vět.

$$sim(\vec{n_k}, \vec{n_a}) = \frac{\vec{n_k}.\vec{n_a}}{|\vec{n_k}|\,|\vec{n_a}|} \tag{1}$$

Vektory n_a a n_k představují ohodnocení termů vět z množiny již vybraných vět a vět, které mají být ještě sumarizovány. Postupně jsou počítány podobnosti vět a věta, obsahující nejméně informací z množiny již sumarizovaných vět, je do korpusu přidána.

Dnes je možné se setkat ještě s jinou podobou multidokumentové sumarizace. Na množině dokumentů je vypočítané skóre pro každý dokument a nakonec jsou vybrány dokumenty, které přinesou uživateli nejvíce neopakujících se informací a uživateli jsou nakonec zobrazeny jen dokumenty s největší informační hodnotou a neopakujícími se tématy.

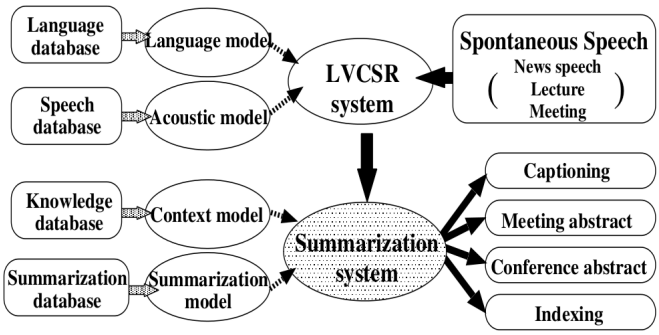
Velmi zajímavou oblastí sumarizace je ASR[1] sumarizace. Tedy sumarizace výsledků systému pro automatické rozpoznávání řeči. Tento typ sumarizace má velmi široké využití. Pomocí ní lze provést zjednodušení dlouhých projevů, získat témata konferencí nebo z diskuzních pořadů vytěžit informace o probíraném tématu.

Proces ASR sumarizace je ovšem ztížen o problematiku automatického rozpoznávání řeči a všech problémů s ní spojenou. Jedná se hlavně o chyby rozpoznání slov. Kvůli vlivu ASR systému byla pro měření výsledků navržena nová evaluační metoda tzv. SumACCY[24]. Z množiny referenčních sumarizací je vybrána sumarizace, která se

[1] Automatic Speech Recognition

nejvíce podobá sumarizaci vytvořené systémem a je vypočítána podobnost těchto dvou sumarizací.

Proces ASR sumarizace také komplikuje tzv. "Cocktail Party Effect", tedy situace, kdy mluví několik mluvčích najednou, která nastává například v diskuzních pořadech. Tento a podobné problémy je ovšem třeba řešit již před začátkem samotného procesu ASR sumarizace a sumarizační systém na tyto jevy, které se běžně při komunikaci lidí objevují, adaptovat.



Obrázek 1: Schéma ASR sumarizačního systému [6]

.

3 Metody sumarizace textu

Již s počátkem digitalizace textu vznikly první sumarizační algoritmy. V této kapitole jsou uvedeny některé významné sumarizační algoritmy, hlavně algoritmy statistické sumarizace, na které je tato práce zaměřena. Popsány jsou první sumarizační metody, které byly optimalizované na hardwarovou nenáročnost, ale i moderní metody, které již nejsou omezeny hardwarovými parametry systému a využívají hlubších analýz textu dokumentu.

3.1 Heuristické metody

Heuristické metody jsou jedny z prvních metod, které byly navrženy. Jeden z prvních algoritmů byl zveřejněn v práci H. P. Luhna [13] v roce 1958. Algoritmus je založen na myšlence, že nejčastěji opakující se termy v textu jsou nejvýznamnější a na základě jejich četnosti lze vytvářet extrakt. Algoritmus nejdříve v jednotlivých větách nalezne termy a jejich četnost a následně věty ohodnotí podle četnosti jejich termů. Věty s největším skóre jsou zahrnuty do souhrnu. Tuto metodu ovšem mírně komplikuje fakt, že nejčastěji vyskytující se slova v jazyce nejsou pro význam věty důležitá. Z tohoto důvodu je vytvořen seznam nejčastěji vyskytujících se slov v jazyce a slova, která obsahuje, jsou z vět odstraněna.

Tato metoda ovšem špatně detekovala očividně významné věty. Věty, které obsahují termy z nadpisu nebo termy zvyšující význam věty (významný, důležitý, výsledek,...) by měly být obsaženy v souhrnu s větší pravděpodobností než ostatní věty. Kombinace těchto znalostí a výpočtu četnosti termů vytvořila první kvalitní metodu automatické sumarizace, které mohla být realizována i na tehdejším hardwaru.

3.2 Statistické metody

Metody řešící nedostatky heuristických metod jsou metody statistické. Tyto metody zavedly do analýzy textu natrénované znalosti o textech, termech a jejich souhrnech. Z hlediska jejich principu existují hlavně dva přístupy. Luhnův sumarizátor realizoval jeden přístup ke statistickým metodám a druhý funguje na základě Bayesovského teorému. Obě tyto metody vyžadují natrénování korpusu, podle kterého budou věty sumarizovaného dokumentu hodnoceny.

3.2.1 Luhnův sumarizátor

Luhnův sumarizátor funguje na základě výpočtu frekvence termu v dokumentu a jeho inverzní dokumentové frekvence v korpusu dokumentů daného jazyka. Výpočet skóre termu je realizován jako součin těchto hodnot.

$$Score(t,d) = tf(t,d) * idf(t,D) = tf(t,d) * log\frac{|D|}{|\{d \in D : t \in d\}|} \tag{2}$$

Vzorec (2) vyjadřuje výpočet skóre termu t v dokumentu d, jehož inverzní dokumentová frekvence byla natrénována na korpusu D. Skóre termu je tedy vypočítáno jako významnost termu v dokumentu, vážena přes jeho natrénovanou inverzní dokumentovou frekvenci. V čím větším počtu dokumentů se term nacházel, tím menší je jeho inverzní dokumentová frekvence a tím má menší význam pro sumarizovaný dokument.

Výsledné skóre věty je pak vypočítáno jako součet všech termů věty [16]. Větu z dokumentu d reprezentuje vektor termů q a výpočet skóre této věty je dán vzorcem (3).

$$Score(q,d) = \sum_{t \in q} tf(t,d) * idf(t,D) \tag{3}$$

Prostup při vytváření souhrnu pomocí Luhnovy sumarizace lze vyjádřit níže uvedeným algoritmem.

1. Načti idf slovník.
2. Vypočti frekvenci pro všechny termy dokumentu.
3. Pro všechny věty dokumentu d vypočti jejich skóre.
4. Do souhrnu zařaď věty s největším skóre.

Tento algoritmus výborně extrahoval nejvýznamnější téma dokumentu, ovšem vedlejší témata byla zanedbána a do souhrnu se nedostala. Proto byl algoritmus rozšířen. Extrahovaným termům v první větě byla nastavena jejich frekvence na nulu a pro výběr další věty bylo přepočítáno skóre všech vět. Tímto se zamezilo opakovanému výběru věty, ve které mělo největší vliv skóre již vybraný termů a již vybrané věty měli skóre nulové, jelikož všechny jejich termy měli nulové skóre. Modifikovaný algoritmus funguje takto:

1. Načti idf slovník.
2. Vypočti frekvenci pro všechny termy dokumentu.

3. Pro všechny věty dokumentu d vypočti její skóre.

4. Do souhrnu zařaď větu s největším skóre.

5. Skóre použitých termů nastav na nulu.

6. Pokud ještě není vybráno požadované množství vět pokračuj bodem 3.

Takto navržený sumarizační algoritmus extrahoval věty na základě povrchních znalostí o dokumentu a jeho hardwarové nároky nebyly nijak přehnané.

3.2.2 Naivní Bayesův klasifikátor

Zcela odlišný přístup ke statistické sumarizaci je využití Bayesovského teorému [11]. Metoda využívající tento teorém vyžaduje trénovací korpus dvojic text-souhrn. Na tomto korpusu jsou spočteny příznaky vět, podle kterých jsou věty klasifikovány. Příznakem mohou být například frekvenčně významné termy, délka věty a jiné důležité jevy. Na korpusu jsou následně vypočítány pravděpodobnosti zařazení vět článků z korpusu do souhrnů. Takto připravený korpus lze využít k určení skóre "věta do souhrnu patří/nepatří"a klasifikovat podle něj věty sumarizovaného dokumentu.

Některé z výše uvedených příznaků mohou být na sobě závislé, ovšem pro potřeby výpočtu pravděpodobnosti zařazení věty do souhrnu je toto zanedbáno a předpokládá se, že jsou jednotlivé příznaky nezávislé [9]. Díky tomu může být použit vzorec (4) pro Bayesův klasifikátor, proto je metoda označována jako naivní Bayesův klasifikátor.

$$P(h|q) = \frac{P(q|h) * P(h)}{P(q)} \tag{4}$$

Vektor q označuje vektor příznaků věty. $P(h|q)$ vyjadřuje skóre věty při výpočtu zařazení věty do souhrnu. Pravděpodobnost $P(q)$ je pravděpodobnost výskytu věty v korpusu text-souhrn, $P(h)$ je poměr počtu vět v souhrnech k počtu všech vět korpusu. Pravděpodobnost $P(q|h)$ vyjadřuje pravděpodobnost, že věta q je zařazena do souhrnu v trénovacím korpusu.

Jelikož je věta q vyjádřena vektorem příznaků, měl by být Bayesovský vzorec (4) upraven na tvar pro jednotlivé prvky vektoru q.

$$P(h|q_1, q_2, ..g_n) = \frac{\prod_{i=1}^{n} P(q_i|h) * P(h)}{\prod_{i=1}^{n} P(q_i)} \tag{5}$$

Jelikož je hodnota pravděpodobnosti velmi malá, nedoporučuje se nechávat vzorec v tomto tvaru, jelikož by mohlo dojít k podtečení datového typu, ale doporučuje se hodnocení vektoru věty provádět podle zlogaritmovaného vzorce (6), který riziko podtečení eliminuje. Navíc můžeme odstranit pravděpodobnost $P(q_i)$, která vzorec nijak neovlivní, protože je její hodnota vždy konstantní. Stejně tak je možné vynechat hodnotu pravděpodobnosti $P(h)$, která vyjadřuje kompresní poměr [11].

$$P(h|q_1, q_2, ..g_n) = \sum_{i=1}^{n} logP(q_i|h) \tag{6}$$

Pro každou větu ze sumarizovaného dokumentu jsou vypočítány pravděpodobnosti podle vzorce (6) a věty, které dosáhnou nejvyšší pravděpodobnosti zařazení do souhrnu, jsou do něj vybrány v pořadí podle nejvyšší dosažené pravděpodobnosti.

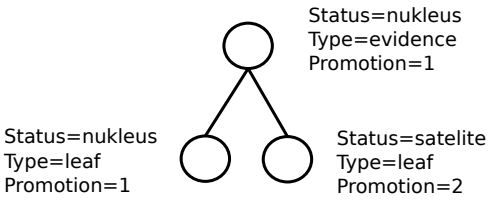
3.3 Moderní přístupy

Moderní přístupy k sumarizaci již nejsou omezeny výkonem hardwaru, tak jako heuristické nebo statistické metody. Díky tomu lze využít hlubší znalosti o dokumentu. V roce 1998 byl prezentován v práci [17] sumarizační systém využívající k sumarizaci teorii rétorických struktur. Dále byly využity znalosti, jako jsou například vzájemné vazby termů, kontext slov a jiné lingvistické znalosti, jejichž výpočet nemohl být dříve realizován. Tyto znalosti jsou využity k vytváření souhrnů, které již dokáží lépe vystihnout informace obsažené v dokumentu. Mezi tyto metody patří například grafové metody, metoda rétorických struktur nebo latentní sémantická analýza.

3.3.1 Teorie rétorických struktur

Teorie rétorických struktur zkoumá skladbu řečového projevu a prostřednictvím rétorických relací zachycuje vazby mezi částmi textu. V práci [17] je popsán sumarizátor, který z jednotlivých částí textu a vztahů mezi nimi vytváří binární strom označovaný jako RS-strom.

Vztahů, které jsou někdy označovány jako role, je používáno celkem 23, viz. [15]. Vztahy mohou být například podmínka, vysvětlení, rozšíření, výsledek, základ, atd.. Podle těchto vztahů mohou uzly nabýt stavu: nukleus, satelit, kombinace nukleů a satelitů a text zvýrazňující jiné části. Za nukleus je považována část textu obsahující nejpodstatnější údaje textu. Satelity jsou části textu nesoucí vedlejší údaje vázané na nuklee. Uzly RS-stromu



Obrázek 2: Ukázka ohodnocení uzlů [2]

jsou ohodnocovány podle jejich rétorické role. Na obrázku 2 je vidět rozdělení nukleu na další nukleus a satelit. Typ uzlu značí jeho rétorickou relaci k vyššímu celku. Promotion značí s kolika dalšími uzly tvoří daný uzel nukleus.

Strom vygenerovaný rétorickým analyzátorem je využit k určení významu částí textu pro celek. Do souhrnu jsou vybírány části textu, které se umístí nejblíže kořenu stromu. Čím rozsáhlejší je požadovaný souhrn, tím více vzdálenější uzly stromu jsou vybírány.

3.3.2 Grafové metody

Velmi zajímavou metodou jsou také metody grafové. Tyto metody využívají pro vytváření souhrnů algoritmy využívané vyhledávacími službami pro hodnocení struktury webu. Například algoritmus PageRank využívá Google pro hodnocení důležitosti webových stránek. Z tohoto algoritmu vznikl algoritmus TextRank, využívaný pro sumarizaci. Algoritmus PageRank hodnotí vrcholy orientovaného grafu G=(V,E) podle stupně sousedních uzlů iteračním výpočtem PR (PageRank).

$$PR(V_i) = \frac{1-d}{N} + d * \sum_{V_j \in In(V_i)} \frac{PR(V_j}{|Out(V_j)|} \tag{7}$$

Vzorec (7) vysvětluje tento výpočet. V první iteraci jsou nastaveny hodnoty PR pro všechny uzly na 1 a během několika iterací je vypočítáno výsledné PR. N je celkový počet vrcholů grafu, d je faktor tlumení nabývajících hodnot 0 až 1 a představuje pravděpodobnost přechodu do sousedního vrcholu. $PR(V_j)$ je PR sousedního vrcholu a $Out(V_j)$ je výstupní stupeň tohoto vrcholu.

Při sumarizaci jsou jako vrcholy grafu považovány jednotlivé věty článku a ohodnocení hrany grafu vyjadřuje vazby mezi sousedními větami. Algoritmus TextRank[18] již nevyužívá orientované grafy a je možné přecházet mezi sousedními větami libovolně. Ohodnocení vazby mezi větami V_i a V_j je vyjádřeno váhou hrany w_{ij}. Výpočet PR pro

věty je prováděn podle modifikovaného vzorce (8).

$$PR(V_i) = \frac{1-d}{N} + d * \sum_{V_j \in In(V_i)} w_{ji} \frac{PR(V_j}{\sum_{V_k \in Out(V_j)} w_{jk}} \tag{8}$$

Určení vazeb mezi větami je provedeno pomocí metrik hodnotící podobnost vět. Lze využít například kosinovou podobnost, překrytí kontextu a jiné metriky, které dokážou určit podobnost vět. Například na Michiganské univerzitě vznikl sumarizátor LexRank[2] využívající kosinovou podobnost vět.

3.3.3 Latentní sémantická analýza

Latentní sémantická analýza převádí problém ohodnocení vět dokumentu na algebraickou úlohu, která dovoluje analyzovat vztahy mezi větami a jejich termy bez nutnosti zásahu člověka. Využití metody LSA pro sumarizaci navrhli Xin Liu a Yihong Gong ve své práci [25]. Inspirovali se latentní sémantickým indexováním využívaným při vyhledávání informací ve velkém datovém korpusu na základě dotazu uživatele.

Sumarizace metodou latentní sémantické analýzy probíhá ve dvou krocích. Prvním je sestavení matice $A = [A_1, A_2, ..., A_n]$, tedy mapování termů dokumentu do jeho vět. Každý sloupcový vektor A_i obsahuje vektor frekvence jednotlivých termů věty i-té věty. Tato frekvence je vážena přes globální frekvenci termu. Možnosti jak vážit termy jsou uvedeny v [3]. Pokud má sumarizovaný dokument m termů a n vět vznikne matice $m \times n$. Tato matice je v dalším kroku rozložena singulární dekompozicí (SVD - singular value dekompozition) na součin matic (9).

Dekompozice dokáže zachytit mapování témat do vět. Tyto vztahy jsou zachyceny v matici V^T, která popisuje mapování termů témat dokumentu do jeho vět. Mapování je zajištěno dekompozicí, která rozděluje původní dokument do lineárně nezávislých vektorů. Tyto vektory vyjadřují základní koncepty dokumentu a věty společně s termy jsou do prostoru těchto vektorů promítány pomocí SVD. Na základě výskytů termů dokáže SVD také detekovat podobné termy. Například termy *lékař* a *doktor*, které se velmi často vyskytují ve společnosti termů *nemocnice*, *medicína* a *nemoc*, budou v prostoru promítnuty velmi blízko u sebe. Takto jsou do vektorového prostoru promítnuty všechny termy dokumentu a je zjištěna důležitost hlavních témat dokumentu podle počtu a vzdálenosti termů k těmto tématům. Věty dokumentu jsou pak ohodnoceny podle toho,

[2] <http://tangra.si.umich.edu/clair/lexrank>

jak jsou jejich termy blízko těmto tématům.

Singulární dekompozice

$$A = U\Sigma V^T \tag{9}$$

Matice U je sloupcově ortonormální[3] matice $m \times n$, která obsahuje levé singulární vektory, matice Σ je čtvercová diagonální matice $n \times n$ obsahující singulární hodnoty v sestupném pořadí a ortonormální matice V^T $n \times n$ obsahuje pravé singulární vektory. Rozměry matice Σ jsou omezeny počtem vlastních čísel matice A^TA, které jsou využity k výpočtu singulárních hodnot a sloupcových vektorů matic U a V^T [8]. Za předpokladu, že vět je vždy méně než termů, které věty obsahují, jsou rozměry matice $\Sigma n \times n$. Počet vlastních čísel matice A^TA definuje i rozměry matice V^T, která je z nich vypočítána.

Výpočtem Euklidovské normy jednotlivých sloupcových vektorů a výběrem vět s největší normou (velikostí) získáme věty, které mají být zařazeny do souhrnu. Takto vybrané věty jsou vybrané podle toho, jak moc věta zachycuje témata článku. Tyto věty ovšem nekorespondují s důležitostí témat článku. Důležitost témat je obsažena v matici Σ. Proto bylo navrženo vylepšení [22], které bere v potaz i důležitost témat. Vylepšený výpočet souhrnu je realizován pomocí vzorce (10).

$$s_r = \sqrt{\sum_{i=1}^{n} v_{ri}^2 * \sigma_i^2} \tag{10}$$

Výsledkem je vektor s, který obsahuje skóre jednotlivých vět a do souhrnu je vybráno potřebné množství vět s největší hodnotou.

Příklad:

Úkol: Vyberte jednu větu, která nejlépe popisuje článek: "The man walked the dog. The man took the dog to the park. The dog went to the park."Věty si označíme a převedeme znaky na malé:

```
v1: the man walked the dog
v2: the man took the dog to the park
v3: the dog went to the park
```

[3]Vektory matice jsou ortogonální a normované

Věty obsahují termy: the, man, walked, the, dog, took, to, park, went. Na základě četnosti jejich výskytu vytvoříme matici A. Pro zjednodušení jsou termy váženy binárně. Pokud věta term obsahuje, je term násoben jedničkou, pokud ne, nulou.

$$A: \begin{array}{c|ccc} & v1 & v2 & v3 \\ \hline the & 2 & 3 & 2 \\ man & 1 & 1 & 0 \\ walked & 1 & 0 & 0 \\ dog & 1 & 1 & 1 \\ took & 0 & 1 & 0 \\ to & 0 & 1 & 1 \\ park & 0 & 1 & 1 \\ went & 0 & 0 & 1 \end{array}$$

Singulárním rozkladem matice A získáme matice U, Σ a V^T. Postup výpočtu SVD je popsán v článku [8]. Matice U není pro výpočet souhrnu potřebná,mann88b takže zde není uvedena.

$$\Sigma = \begin{pmatrix} 5.0325 & 0 & 0 \\ 0 & 1.5745 & 0 \\ 0 & 0 & 1.0930 \end{pmatrix} \qquad V^T = \begin{pmatrix} -0.4572 & 0.7699 & -0.4453 \\ -0.7284 & -0.0368 & 0.6842 \\ -0.5103 & -0.6372 & -0.5776 \end{pmatrix}$$

Podle vzorce (10) je vypočítán vektor s a vybrána věta, která nese nejvíce informací z nejdůležitějších témat dokumentu.

$$s = \begin{array}{ccc} 2.6458 & 3.7417 & 2.8284 \end{array}$$

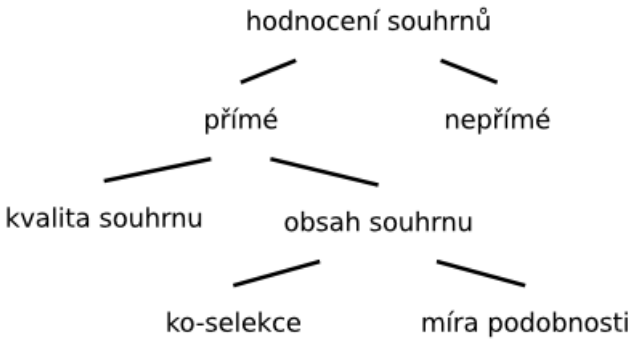
Jelikož máme vybrat jen jednu větu, najdeme největší hodnotu ve vektoru s a její index nám udává kolikátá věta je nejvhodnější pro zařazení do souhrnu. Výsledným souhrnem je tedy věta: The man took the dog to the park.

Latentní sémantickou analýzu je možné využít i pro multidokumentovou sumarizaci. Při provádění multidokumentové sumarizace je nutné zabránit výběru stejných vět z různých dokumentů. K tomu je možné využít například hodnoty kosinu úhlu, které svírá věta vybraná do souhrnu s větami souhrnu. Pokud nejmenší hodnota je větší než experimenty určený práh, je možné větu zařadit do souhrnu. Aby nebyly vybírány věty

velmi podobné již větám v souhrnu je aplikován algoritmus Iterative Residual Rescaling [1], který omezí vliv již vybraných témat na výběr nových (upraví velikost jejich vektorů).

4 Hodnocení sumarizací

Hodnocení nebo-li evaluace sumarizací je důležitou částí vývoje sumarizačního systému. Hodnocení se provádí pomocí souhrnů, které metody generují. Metody vyhodnocení výsledných souhrnů je možné rozdělit do dvou základních skupin, a to na metody přímé a nepřímé [21].



Obrázek 3: Dělení metod hodnocení souhrnů [21].

4.1 Přímé metody

Přímé metody vyhodnocují text souhrnu tak, jak je napsán. Nevyhledávají v něm kontext slov ani jejich význam, ale hodnotí ho podle podobnosti s referenčními souhrny nebo podle analýzy zkušených lingvistů.

4.1.1 Kvalita souhrnu

Metody hodnotící kvalitu textu jsou zaměřeny hlavně na gramatickou správnost textu, redundantnost a srozumitelnost výsledného souhrnu. Je důležité, aby text souhrnu neobsahoval opakující se témata a věty vyjadřující stejné informace. Srozumitelnost souhrnu ovlivňují hlavně reference extrahované z textu. Hlavně extrakce zájmen zvyšuje riziko, že bude toto zájmeno pochopeno ve špatném kontextu a věta v souhrnu bude mít jiný smysl než v textu. Pokud pro nás není důležitá vysoká rychlost, je možné provést analýzu textu a reference se pokusit nahradit jejich skutečným smyslem. Tato substituce je v podstatě jediným faktorem sumarizace, který může ovlivnit gramatickou správnost extrahovaného souhrnu, pokud nebereme v potaz chyby autorů textu. Všechny tyto metody hodnocení souhrnů jsou prováděny zkušenými lingvisty, kteří souhrny ručně hodnotí.

4.1.2 Ko-selekce

Ko-selekční metody vypočítávají na vzniklém souhrnu hodnoty určující míru ko-selekce. Nejvýznamnějšími jsou přesnost P (precision), úplnost R (recall), úspěšnost A (accuracy) a f-skóre F. Pro výpočet těchto hodnot je nutné vytvořit ideální (referenční) souhrny. Tyto souhrny vytváří anotátoři na množině článků, ze kterých jsou vytvořeny i souhrny automatické. Přesnost je vypočítána jako počet vět vybraných systémem a anotátory zároveň dělen počtem vět vybraných systémem. Úplnost je definována jako počet vět vybraných systémem a anotátory zároveň dělen počtem vět vybraných anotátory. Úspěšnost je jako poměr součtu vět vybraných systémem i anotátory zároveň a vět nevybraných systémem ani anotátory k součtu všech možností výběru [5]. Zjednodušení zápisu vzorců je provedeno pomocí kategorizace možných výsledků porovnání souhrnů, viz. tabulka 1.

	vybráno anotátory	nevybráno anotátory
vybráno systémem	TP	FN
nevybráno systémem	FP	TN

Tabulka 1: Tabulka možných výsledků porovnání vět anotátorských a systémových souhrnů.

Vzorce pro výpočet přesnosti, úplnosti a úspěšnosti:

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN} \qquad A = \frac{TP + TN}{TP + TN + FP + FN} \tag{11}$$

Z hodnot přesnosti a úplnosti je vypočítáno f-skóre. Jeho hodnota je definována jako harmonický průměr hodnot úplnosti a přesnosti.

$$F = \frac{2RP}{R + P} \tag{12}$$

Pokud chceme při výpočtu f-skóre upřednostnit úplnost nebo přesnost, využijeme upravený vzorec pro výpočet f-skóre.

$$F = \frac{(1 + \beta^2)RP}{\beta^2 P + R} \tag{13}$$

Proměnná β ovlivňuje, jestli dáváme větší váhu přesnosti ($\beta > 1$), úplnosti ($\beta < 1$) nebo pokládáme obě hodnoty za stejně významné ($\beta = 1$).

4.1.3 Základní míry podobnosti

Nevýhodou ko-selekčních metod je, že při výpočtu se orientují na celé věty souhrnů, takže dvě věty s velmi podobným tématem zapsané odlišně jsou ohodnoceny velmi nízkým hodnocením. Tyto problémy řeší metody zkoumající míru podobnosti vět. Metody nepracují s větami souhrnů jako s celkem, ale využívají slov ve větě pro potřeby hodnocení sumarizačních systémů. Stejně jako u ko-selekčních metod i metody výpočtu míry podobnosti využívají referenční souhrny vytvořené anotátory. Základními hodnotícími technikami jsou kosinová podobnost, překrývání obsahu a nejdelší společná subsekvence [21].

Vektory X a Y jsou vektory vět obsahující slova ze souhrnu anotátorského (x_i) a souhrnu vytvořeného systémem (y_i).

Kosinová podobnost:

$$cos(X, Y) = \frac{\sum_i x_i * y_i}{\sqrt{\sum_i (x_i)^2} * \sqrt{\sum_i (y_i)^2}} \tag{14}$$

Kosinová podobnost vyjadřuje vzdálenost mezi dvěma vektory. Čím větší je kosinová podobnost, tím více si jsou věty podobné.

Překrytí obsahu:

$$overlap(X, Y) = \frac{|X \cap Y|}{|X| + |Y| - |X \cap Y|} \tag{15}$$

Překrytí obsahu (ang. unit overlap) vyjadřuje, kolik mají souhrny společných slov nebo lémat.

Nejdelší společná subsekvence:

$$lcs(X, Y) = \frac{velikost(X) + velikost(Y) - úpravy_{di}(X, Y)}{2} \tag{16}$$

Velikost X a Y vyjadřuje počet prvků těchto dvou řetězců a $úpravy_{di}(X, Y)$ je počet operací vložení (insertion) a mazání (deletion) nutných k úpravě X na Y.

4.1.4 ROUGE

Pro automatické hodnocení sumarizačních systémů byl vytvořen program ROUGE (Recall-Oriented Understudy for Gisting Evaluation)[12]. ROUGE je využíván na konferencích TAC (dříve DUC) k hodnocení sumarizačních systémů. Program dovoluje provádět automatické hodnocení souhrnů na základě metrik míry podobnosti. K tomuto potřebuje anotátory vytvořené referenční souhrny, které využívá k výpočtu hodnot přesnosti a úplnosti. Tento program je dostupný ze stánek projektu[4].

Program ROUGE dokáže hodnotit souhrny v několika režimech. Základním režimem vyhodnocování souhrnů je ROUGE-N. ROUGE-N provádí měření na principu výpočtu identických n-gramů mezi souhrnem vygenerovaným systémem a sadou referenčních souhrnů. Výpočet skóre ROUGE-N mezi referenčním a vygenerovaným souhrnem je prováděn podle vzorce (17).

$$ROUGE - N = \frac{\sum_{C \in RSS} \sum_{gram_n \in C} Počet_{souhlasí}(gram_n)}{\sum_{C \in RSS} \sum_{gram_n \in C} Počet(gram_n)} \tag{17}$$

RSS je sada vět referenčního souhrnu, n značí délku n-gramu, $Počet(gram_n)$ je počet n-gramů v referenčním souhrnu a $Počet_{souhlasí}(gram_n)$ je maximální počet n-gramů, které se vyskytují zároveň v referenčním i hodnoceném souhrnu. Při reálném měření skóre souhrnu je využito více referenčních souhrnů.

$$ROUGE - N_{multi} = argmax_i ROUGE - N(r_i, s) \tag{18}$$

Čím více referenčních souhrnů sdílí stejný n-gram, tím větší skóre bude mít věta, která ho obsahuje. Se vzrůstajícím počtem referenčních souhrnů klesá hodnota ROUGE-N, jelikož roste velikost množiny n-gramů referenčních souhrnů, kterou je dělen počet shodujících se n-gramů.

ROUGE-L a ROUGE-W realizují výpočet nejdelší společné subsekvence - LCS (Longest Common Subsequence). LCS označuje nejdelší společnou subsekvenci dvou vektorů slov. Metoda funguje na myšlence, že delší subsekvence společná pro souhrny je lépe ohodnocena než kratší. Metoda ROUGE-L nevyžaduje, aby subsekvence byla souvislá. To znamená, že dvě sekvence slov obsahující stejnou subsekvenci mají stejné skóre, i když jedna obsahuje subsekvenci spojitou a druhá ne. Toto řeší metoda ROUGE-W, která měří nejdelší spojitou subsekvenci.

[4] <http://berouge.com/default.aspx>

Poslední využívanou metodou je ROUGE-S, která využívá četnosti výskytů skip-bigramů v souhrnech pro vyhodnocení souhrnů. Rozsah skip-bigramu je omezen uživatelem a vyjadřuje kolik unigramů je možné při výpočtu přeskočit.

$$R_{skip2} = \frac{SKIP2(X,Y)}{C^2_{velikost(X)}} \qquad P_{skip2} = \frac{SKIP2(X,Y)}{C^2_{velikost(Y)}} \qquad F_{lcs} = \frac{2R_{skip2}P_{skip2}}{P_{skip2} + R_{skip2}} \tag{19}$$

Kde $SKIP2(X,Y)$ je počet skip-bigramů společných pro referenční větu X a testovanou větu Y. $C^2_{velikost(X)}$ je kombinační číslo vyjadřující počet všech bigramů v referenční větě a $C^2_{velikost(Y)}$ je počet všech skip-bigramů ve větě testované.

Bigramy jsou vytvářeny podle pořadí ve větě, to má za následek, že nelze v testované větě, která obsahuje stejné unigramy jako věta referenční ale v opačném pořadí, nalézt ani jeden stejný bigram. Ovšem je zřejmé, že věty obsahují stejnou informaci a ohodnocení testované věty by nemělo být nulové. Tento nedostatek řeší rozšíření metody ROUGE-S na ROUGE-SU. Tato metoda rozšiřuje ROUGE-S o výpočet společných unigramů.

4.2 Nepřímé metody

Nepřímé metody hodnotí souhrny pomocí různých disciplín z oblasti dolování informací z textu. Nepřistupují k textu souhrnu po částech (slovech nebo větách), tak jako přímé metody, ale analyzují informace v textu obsažené. V angličtině jsou nepřímé metody označované jako "task-based", tedy metody založené na určitých úlohách. Nejvýznamnějšími úlohami jsou kategorizace dokumentů, vyhledávání informací a zodpovídání otázek.

4.2.1 Kategorizace dokumentů

Při hodnocení souhrnů kategorizací je vytvořen korpus anotovaných dokumentů, u kterého jsou pro každý dokument určeny kategorie, do kterých dokument spadá. Testování probíhá tak, že jsou dokumenty kategorizovány na souhrnech těchto dokumentů a následně jsou tyto kategorie porovnány s kategoriemi určenými pro původní text dokumentu. Pokud je souhrn kvalitní náhradou dokumentu, tak se kategorie souhrnu i dokumentu shodují a můžeme prohlásit, že sumarizační metoda, která souhrn vytvořila, dokáže kvalitně sumarizovat informace potřebné pro kategorizaci.

Kategorizaci je možné provádět ručně, ale i automatickými kategorizujícími systémy. Ruční kategorizace ovšem zajišťuje kvalitnější výsledky než kategorizace automatická. Při použití automatické kategorizace je potřeba rozlišovat chyby kategorizace a sumarizace.

4.2.2 Vyhledávání informací

Tato metoda hodnocení souhrnů je založena na předpokladu, že dobrý souhrn umožňuje vyhledat stejné informace jako celý dokument a dotaz položený na dobrý souhrn vrátí stejně kvalitní výsledky jako dotaz položený na celý dokument. Sumarizační systémy jsou hodnoceny na základě relativního poklesu informací při nahrazení plného textu souhrnem tohoto textu. Pro účely měření sumarizačních systémů pomocí vyhledávání informací bylo navrženo několik metod například Kendallovo tau, Spearmanova korelace [20], lineární korelace nebo korelace relevance dat [19].

4.2.3 Zodpovídání dotazů čtenáři

Velmi zajímavá metoda ohodnocení sumarizačního systému je využití lidí odpovídajících na otázky zaměřené na informace obsažené v textu dokumentu [14]. Lidé odpovídali na otázky na základě získaných znalostí a to ve třech fázích. V první fázi odpovídali na otázky bez přečtení článku ani souhrnu. Ve druhé fázi odpovídali na stejné otázky a měli k dispozici automaticky generovaný informativní souhrn. Nakonec odpovídali po přečtení plného znění článku. Získané výsledky byly porovnány a byly zkoumány zlepšení odpovědí na otázky při vzrůstajícím objemu informací.

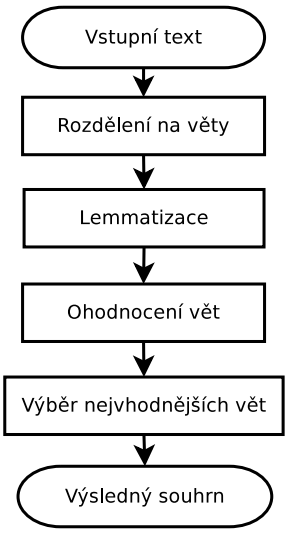
5 Implementace sumarizačních metod

Pro implementaci byly vybrány metody, které nevyžadovaly účast anotátorů ani jiných lingvistů k realizaci metody. Nakonec byla vybrána heuristická a Luhnova metoda jako zástupci statistických metod a sumarizační metoda založená na latentní sémantické analýze, která využívá moderních přístupů k procesu vytváření souhrnů.

Implementace všech metod byla provedena v jazyce C# a platformě Mono, která dovoluje vysokou přenositelnost binárního kódu díky implementaci Mona na systémech Linux, Mac i Windows. Podobnou přenositelnost dovoluje jen jazyk Java, který je ovšem kvůli nutnosti kompilace Java bitekódu při každém spuštění výrazně pomalejší.

5.1 Předzpracování vstupních dat

Před začátkem vlastní sumarizace je potřeba vstupní text předzpracovat. Text vstupuje do procesu sumarizace jako jeden dlouhý řetězec znaků. Tento řetězec je ovšem nutné rozdělit na jednotlivé věty. Tyto věty (pole řetězců) je teoreticky možné využít pro vytvoření souhrnu, ovšem kvůli ohebnosti jazyků je nutné rozdělené věty ještě lemmatizovat. Postup vytvoření souhrnů je zobrazen na obrázku 4.



Obrázek 4: Schéma postupu vytvoření souhrnu

Třída Preparation

Pro přípravu dat byla vytvořena třída *Preparation*, která řeší problematiku dělení vět a jejich lemmatizaci. Třída obsahuje celkem tři statické metody. První z nich je metoda **Raw2Sentences(string text)**. Tato metoda rozdělí vstupní řetězec na jednotlivé věty a

vrací pole řetězců. Při dělení vět bylo nutné řešit problémy, jako jsou zkratky jmen, datum, pořadové číslovky a jiné problémy, které ztěžují detekci konce věty.

Další dvě metody lemmatizují věty. Jsou to metody **Lemmatisation(string[] sents)** a **GetLemma(string line)**. Metoda *Lemmatisation* slouží k lemmatizaci vstupního pole vět a vrací pole lemmatizovaných vět. Metoda *GetLemma* slouží k lemmatizaci jedné "věty"a je využívaná pro lemmatizaci nadpisů a klíčových slov. K vlastní lemmatizaci je využit lemmatizátor, který byl vytvořen na Karlově univerzitě v Ústavu formální a aplikované lingvistiky a je dostupný na stránce ústavu[5]. Kvůli časové náročnosti inicializace lemmatizátoru je metoda *Lemmatisation* implementovaná tak, že věty, které má lemmatizovat, spojí přes speciální sekvenci znaků a nakonec volá metodu *GetLemma*, jenž věty vrací již lemmatizované. Následně jsou věty opět rozděleny pomocí vložené speciální sekvence znaků.

5.2 Interface metod

Pro sjednocení ovládání bylo na navrženo rozhraní obsahující metody, které musí každá sumarizační metoda implementovat.

```
interface SummarizationInterface
{
        void CreateSummary();
        void CreateSummary(string text);
        string[] GetSummaryByPercentOfText(uint percent);
        string[] GetSummaryByCountOfSentences(uint count);
}
```

Kód 1: Interface sumarizační metody

Metoda *CreateSummary* provádí přípravu algoritmu sumarizace na jeho činnost (lemmatizuje text, počítá frekvenci termů,...). Vstupní text, klíčová slova a ostatní potřebná nastavení jsou předány instanci metody v konstruktoru. Ovšem některé metody při vytvoření jejich instance načítají korpus, který potřebují pro vytvoření souhrnu, což výrazně prodlužuje dobu běhu sumarizátoru. Tento problém je vyřešen přetížením metody CreateSummary. Metoda *CreateSummary(string text)* umožňuje využít již načtený korpus pro vytvoření nového souhrnu.

[5] <http://ufal.mff.cuni.cz/pdt/Morphology_and_Tagging/Morphology/>

Metody *GetSummaryByPercentOfText* a *GetSummaryByCountOfSentences* vrací požadovaný počet vět souhrnu. Tento souhrn je reprezentován polem řetězců, které je setříděné podle vypočítané hodnoty věty.

Abstraktní metoda SummarizationMethod

Pro obecnější využití implementovaných metod byla napsána abstraktní třída *SummarizationMethod*. Využití abstraktní třídy je znázorněno v kódu 2.

```
SummarizationMethod summary=null;
switch(metoda)
{
        case "heuristic":
                summary=new Heuristic(text,title,lang);
                break;
        case "lsa":
                summary=new LSA(text,title,lang,useCZ_IDF);
                break;
        default:
                summary=new Luhn(text,title);
                break;
}
summary.CreateSummary();
string[] sum = summary.GetSummaryByCountOfSentences(4);
```

Kód 2: Příklad použití abstraktní metody SummarizationMethod

Kód 2 je využíván sumarizačním serverem (viz. kapitola 5.6, který využívá knihovnu Summarization, jenž obsahuje všechny implementované sumarizační algoritmy a třídu Preparation. V proměnné *metoda* je uložen název uživatelem vybrané metody a podle ní je určeno, která metoda je ve skutečnosti využita pro vytvoření souhrnu.

5.3 Heuristická metoda

Heuristická metoda byla implementována jako první metoda, na které byly odzkoušeny metody třídy Preparation. Tato metoda vyžaduje tzv. StopList, který obsahuje seznam nejčastějších termů jazyka sumarizovaného textu. StopList pro český jazyk byl získán zlemmatizováním unigramů obsažených v článcích získaných od firmy NEWTON

Media a následným výběrem nejčetnějších 311 lémat. Pro anglický jazyk byl získán ze stránek projektu Proteus New Yorské univerzity[6]. Po odstranění termů ze StopListu proběhne výpočet četnosti termů a úprava četnosti termů uvedených v klíčových slovech, nadpisu a doméně. Na základě získaných četností termů jsou ohodnoceny věty vstupního textu a vytvořen souhrn.

5.4 Luhnův sumarizátor

Luhnův sumarizátor vyžaduje pro výpočet slovník inverzní dokumentové frekvence termů jazyka. Tento slovník byl vytvořen z již dříve zmíněných článků. Kvůli ohebnosti českého jazyka bylo nutné vypočítat četnost dokumentů obsahující lemmatizovaný term. Pro tyto účely byly vytvořeny nástroje na výpočet inverzní dokumentové frekvence lemmatizovaných termů. Původně se jednalo o jeden program, který prováděl všechny výpočty najednou, ale jeho běh trval moc dlouho (řádově týdny). Proto byly vytvořeny dva oddělené programy.

První program převedl vstupní xml soubor s články na speciální soubor, který byl pak lemmatizován. Program vytvořil 4 vlákna, která prováděla lemmatizaci čtyř vstupních souborů najednou a výsledek uložila do nového xml souboru s lemmatizovanými články. Celkem bylo lemmatizováno 2 228 021 článků.

Druhý program vytváří vlastní slovník, který slouží k výpočtu inverzní dokumentové frekvence. Program vytváří slovník termů, do kterého přidává termy z lemmatizovaných článků, a vypočítává četnost článků obsahující tyto termy. Program byl vytvořen ve dvou verzích. První verze vyžadovala předem vytvořený slovník slov, jejichž četnost měla být vypočítána. Použitý slovník byl vytvořen lemmatizováním slovníku unigramů. Obsahuje přibližně 180 tisíc nejčastějších českých lemmat. Druhá verze vytvářela slovník dynamicky na základě nalezených lemmat v článcích. Ovšem tento slovník byl zbytečně velký a obsahoval i překlepy autorů článků. Slovník obsahoval přes dva a půl milionu termů a zabíral v paměti 34 MB, což by značně prodlužovalo spouštění sumarizačního programu. Proto byl slovník upraven omezením počtu termů na základě jejich četnosti a to tak, že nejmenší přípustná četnost byla nastavena na padesát

[6] <http://nlp.cs.nyu.edu/GMA_files/resources/>

výskytů. Takto omezený slovník obsahuje cca. 129 tisíc termů a zabírá již jen 1,7 MB.

```
pocet_dokumentu:2228021
v       2072350
s       1983432
a       1975942
být     1968714
```

Kód 3: První 4 řádky natrénovaného slovníku

Díky omezení slovníku a specifickým termům, které slovník neobsahuje, vzniká problém, jak vypočítat inverzní dokumentovou frekvenci pro neznámé termy. Tento problém byl vyřešen úpravou výpočtu hodnoty pro neznámé slovo. Úprava vychází z předpokladu, že slovo, které slovník neobsahuje, je velmi specifické a tím pádem i důležité. Proto byl nulový výskyt termu v trénovacím korpusu nahrazen jedním výskytem. Výpočet je znázorněn v kódu 4.

```
double val=0;
foreach(string word in sentence.words)
{
        try
        {
                val+=tf[word]*idf[word];
        }
        catch (KeyNotFoundException)
        {
                val+=tf[word]*Math.Log(pocetDokumentu);
        }
}
sentence.Score=val;
```

Kód 4: Výpočet skóre věty

5.5 Latentní sémantická analýza

Celý algoritmus vytvoření souhrnu pomocí latentní sémantické analýzy lze rozdělit na tři části:

1. vytvoření matice termů a vět

2. výpočet dekompozice matice

3. výpočet skóre vět

Sloupečky matice A, jež je použitá jako vstupní matice dekompozice, obsahují jednotlivé věty sumarizovaného článku. Řádky matice obsahují termy článku. Hodnota $a_{t,v}$ v matici reprezentuje počet výskytů termu t ve větě v. Tato hodnota je ještě vážena přes globální frekvenci termu [3]. Kompletní matice A je předána dále do dekompozice.

Byly implementovány dvě verze výpočtu matice A. První možnost, jako vážit hodnoty v matici A, je určena jen pro český jazyk, využívá slovník IDF vytvořený pro Luhnův sumarizátor. Term je vážen jeho inverzní dokumentovou frekvencí. Před výpočtem četnosti termu je dokument lemmatizován, aby byly sjednoceny všechny tvary jednoho slova do jednoho termu. Druhá možnost je vážení termů na základě globální frekvence v sumarizovaném dokumentu. Tuto možnost je nutné použít pro jiný než český jazyk, jelikož dokument neprochází procesem lemmatizace, který funguje jen pro český jazyk. Jelikož není k dispozici IDF slovník, který omezuje vliv nejčastějších termů jazyka, je nutné odstranit nejčastější termy. K tomu mám slouží StopList, který je využíván heuristickou metodou.

Pro výpočet dekompozice matice je využita open source knihovna *ALGLIB* dostupná zdarma pro výzkum z webových stránek projektu[7]. Knihovna umožňuje provádět široké spektrum matematických operací, včetně singulární dekompozici matice, a to velmi efektivně.

```
alglib.rmatrixsvd(A, m, n, vypoctiU, vypoctiVT, pridavnaPamet,
                  out S, out U, out VT);
```

Kód 5: Volání metody provádějící singulární dekompozici

Parametr m je počet řádků matice A a n je počet sloupečků matice A. Parametry vypoctiU, vypoctiVT a pridavnaPamet ovlivňují rychlost výpočtu a podobu výsledných matic S, U a VT. Parametr pridavnaPamet je doporučeno nastavit na hodnotu 2. Při tomto nastavení algoritmus potřebuje navíc m*min(m,n) reálných čísel, ale dosahuje maximálního výkonu. Parametry vypoctiU a vypoctiVT ovlivňují obsah matic U a VT. Parametr lze nastavit na hodnoty 0 až 2, kdy při hodnotě 0 není matice vypočítána, při hodnotě 1 je vypočítáno jen prvních min(m,n) sloupečků(matice U) nebo řádků (matice VT) a při hodnotě 2 je vypočtena celá matice. Metoda vrací ve vektoru S singulární hodnoty, v matici

[7] <http://www.alglib.net/>

U levé singulární vektory a v matici VT pravé singulární vektory. Jelikož pro sumarizaci není potřeba matice U, není tato matice vypočítána a v matici VT je vypočteno jen prvních n potřebných vektorů. Tímto je dekompozice výrazně urychlena.

Matice VT a vektor S jsou použity pro výpočet výsledného skóre vět dokumentu podle vzorce (10). Výsledkem výpočtu je vektor ohodnocení vět. Do konečného souhrnu je vybrán uživatelem zadaný počet vět s nejlepším ohodnocením.

5.6 Vytvořené implementace

Všechny tyto metody a třída pro pre-processing dat jsou obsaženy v **knihovně Summarization.dll**. Pokud potřebujeme v nějakém projektu využít sumarizaci, stačí přidat knihovnu do referencí a pomocí direktivy *using Summarization;* připojit k projektu její jmenný prostor. Použití knihovny je znázorněno výše v kódu 2.

Knihovna Summarization byla využita pro následnou realizaci všech sumarizátorů a pro potřeby evaluace implementovaných metod byla využita v rámci lokálního sumarizátoru, který sumarizoval články určené jako testovací množina.

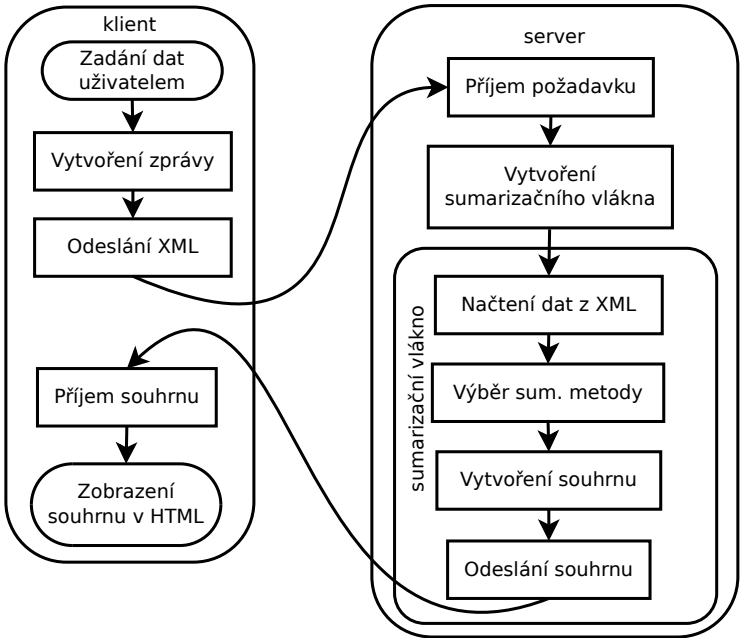
Na webové stránce `<http://nashida.ite.tul.cz>` se nachází **online sumarizátor** využívající knihovnu Summarization. Tato stránka vznikla jen jako prezentační prostředek pro knihovnu a neklade si za cíle využití moderních webových technologií. Sumarizátor funguje na principu klient-server. Klient je napsán v jazyce php a komunikuje se serverem (napsaný v C#) pomocí jednoduché xml zprávy.

```
<?xml version="1.0" encoding="UTF-8"?>
<data>
        <title>title</title>
        <text>text</text>
        <lang>cz</lang>
        <method>LSA</method>
        <percent>25</percent>
</data>
```

Kód 6: Struktura xml zprávy

Komunikace mezi klientem a serverem probíhá na bázi socketů. Pokud klient zašle na server, na port, na kterém server přijímá požadavky, xml zprávu, je na serveru spuštěno sumarizační vlákno, které vytvoří souhrn a je po odeslání souhrnu ukončeno. Vrácen je

souhrn vygenerovaný jako html kód, který je možné přímo zobrazit uživateli. Celý proces vytvoření souhrnu serverem je znázorněn na obrázku 5.



Obrázek 5: Schéma činnosti online sumarizátoru

V příloze A je uveden článek a ukázka jeho souhrnů vytvořených pomocí implementovaných metod. Jedná se o jeden z článků využitý k vyhodnocení implementovaných metod.

6 Vyhodnocení implementovaných metod

Pro vyhodnocení nebo-li evaluaci implementovaných metod byl použit nástroj ROUGE, který byl blíže popsán v kapitole 4.1.4.

6.1 Zadání experimentů

V zadání diplomové práce byly navrženy tři základní experimenty:

1. Vyhodnotit kvalitu vytvořeného sumarizátoru.
2. Porovnat vytvořený sumarizátor se těmi dostupnými na Internetu.
3. Analyzovat souhrny přeložené do neohebného jazyka.

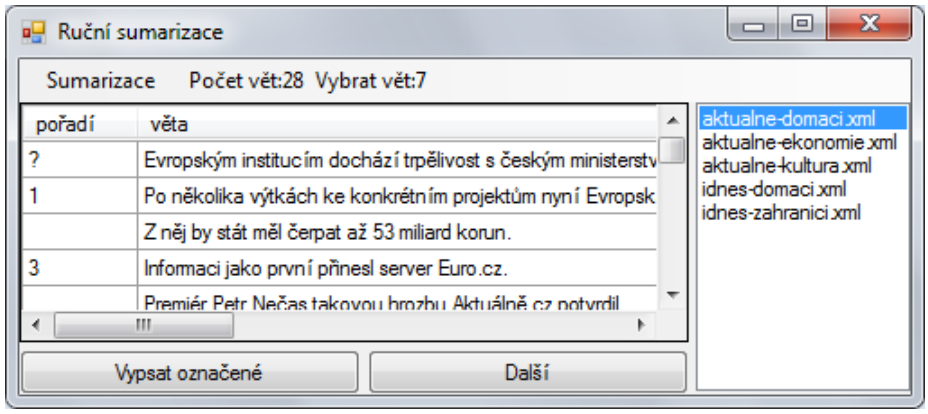
K těmto experimentům byl přidán ještě jeden. Porovnání metody latentní sémantické analýzy využívající pro vážení termů inverzní dokumentovou frekvenci termu a LSA používající globální frekvenci termu v dokumentu. Pro realizaci těchto experimentů bylo potřeba připravit referenční souhrny a vygenerovat, popřípadě ručně upravit, souhrny získané od sumarizačních systémů.

6.2 Vytvoření evaluačního korpusu

Před začátkem vlastní evaluace sumarizačních metod, bylo nutné vytvořit množinu souhrnů dokumentů, která slouží jako reference pro evaluaci metod. Jelikož je vytvoření množiny těchto referenčních souhrnů časově náročné bylo vybráno 25 článků z oblasti domácí a zahraniční politiky, kultury a ekonomie. Články byly získány ze serverů aktualne.cz a novinky.cz a byly předloženy anotátorům.

Anotátorům bylo řečeno, aby z článků vybrali vždy 25 procent vět, u kterých se jim zdá, že nejlépe popisují daný článek a tyto věty ještě očíslovali podle priority. Více instrukcí anotátoři neobdrželi, aby nebyli ovlivněni. Z tohoto důvodu byly z článků odstraněny i nadpisy. Nejdůležitější byly názory anotátorů na význam jednotlivých vět pro celý článek.

Pro usnadnění práce anotátorů byl vytvořen program, který jim celý proces ruční sumarizace usnadnil. Tento program pomocí knihovny Summarization a třídy Preparation rozdělil text článku na jednotlivé věty a zobrazil je do tabulky. Anotátoři, pak do tabulky vyplňovali pořadí k prvním 25 procentům nejvýznamnějších vět článku.



Obrázek 6: Screenshot programu pro ruční sumarizaci

Sumarizace všech článků trvala minimálně hodinu a půl nepřetržité náročné práce. Nakonec se přihlásilo 11 lidí, kteří byli ochotní články zdarma anotovat. Souhrny vytvořené těmito anotátory, pak vytvořily množinu referenčních souhrnů.

Referenční souhrny anotátorů se samozřejmě kompletně nepřekrývají. Každý anotátor vybíral věty podle svých preferencí, znalostí a zálib. V tabulce 2 je uvedena ukázka, překrývání vět anotátorů u jednoho z referenčních článků.

Tabulka 2: Ukázka překrytí referenčních vět článku z přílohy A.

uživatelů	vybraná věta
11	Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda...
8	Důkladné restaurační práce pak odhalily, že obraz vznikl...
7	Obraz vznikl hned ve stejné době jako originál...

Tabulka uvádí kolik uživatelů do svého souhrnu zahrnulo určitou větu. Kompletní tabulka překrytí vět u tohoto článku je uvedena v příloze B. Soubory s překrytím vět u ostatních článků se nacházejí na přiloženém DVD.

6.3 Příprava dat

I když evaluační balík ROUGE po stažení obsahuje dokumenty, které popisují jeho výpočetní mechanismy, neexistuje v podstatě žádný dokument ani online článek zabývající se přípravou dat pro evaluaci. Jediný zdroj informací, jak připravit data, byl nalezen na blogu K. Ganesan [7]. Na blogu je popsána struktura konfiguračního xml souboru a formát

souhrnů nutný pro běh programu. Podle těchto informací byla připravena data od anotátorů a sumarizačních systémů.

Aby bylo dosaženo výsledků, které lépe korespondují s reálným obsahem vět, byly referenční i systémové souhrny lemmatizovány. Například máme-li věty:

v1="Celou situaci řeší už několik dní tým analytiků."

v2="Na vyřešení celé situace mají analytici již jen tři dny."

a využíváme-li pouze nelemmatizované unigramy, pak tyto věty nemají společný žádný unigram. Věty v1 a v2 po zlemmatizování mají tvar:

l1="Celý situace řešit už několik den tým analytik."

l2="Na vyřešení celý situace mít analytik již jen tři den."

a obsahují již 4 společné unigramy, což je rozhodně lepší než v předchozím případě. Pokud by věty v1 a l1 byly referenční pro dva oddělené experimenty, pak využitím vzorce (17) získáme skóre pro věty v2 a l2 vygenerované sumarizátorem:

$$ROUGE-1(v2) = \frac{počet\ společných\ unigramů(v1, v2)}{počet\ unigramů(v1)} = \frac{0}{8} = 0$$

$$ROUGE-1(l2) = \frac{počet\ společných\ unigramů(l1, l2)}{počet\ unigramů(l1)} = \frac{4}{8} = 0.5$$

6.4 Výsledky experimentů

Evaluační úloha byla spuštěna s těmito parametry:

```
./ROUGE-1.5.5.pl -e data -2 -4 -U -n 4 -w 1.2 -a settings.xml
```

Kód 7: Nastavení evaluační úlohy pro ROUGE

ROUGE je nastaven, aby spočítal skóre ROUGE-L, ROUGE-W, ROUGE-S, ROUGE-SU a skóre pro 1-gramy až 4-gramy. Parametr -e říká, kde má ROUGE hledat data WordNetu, parametry -2 -4 -U nastaví výpočet skip-bigramů od délce maximálně 4 s rozšířením SU. Díky parametru -n 4 je nastaven maximální rozsah n-gramů na 4, -w 1.2 určuje váhu s jakou je počítána souvislá subsekvence a parametr -a říká, že mají být testovány všechny sumarizační systémy uvedené v konfiguračním souboru settings.xml.

6.4.1 Evaluace vytvořeného sumarizátoru

První evaluační úlohou bylo vyhodnotit implementaci vybraných sumarizačních metod. Označení R, P a F odpovídá úplnosti, přesnosti a f-skóre viz. kapitola 4.1.2.

Tabulka 3: Výsledky evaluace implementovaných metod v procentech

	ROUGE-1			ROUGE-2			ROUGE-W			ROUGE-SU		
	R	P	F	R	P	F	R	P	F	R	P	F
Heur.	65,0	60,8	62,7	51,3	48,0	49,4	23,5	43,2	30,4	36,8	33,1	34,5
Heur.+téma	61,6	61,5	61,3	48,1	47,9	47,8	22,4	43,8	29,5	34,7	35,1	34,3
Luhn	72,2	60,8	65,9	58,7	49,4	53,6	25,9	42,8	32,2	44,1	32,0	36,9
Luhn+téma	71,3	61,0	65,6	57,9	49,5	53,3	25,8	43,3	32,2	44,5	33,3	37,9
LSA	75,2	60,9	67,2	62,4	50,5	55,8	27,4	43,5	33,6	47,4	31,8	38,0
LSA+téma	75,1	60,9	67,1	62,3	50,5	55,7	27,4	43.6	33,6	47,3	31,8	37,9

Experiment sloužil k porovnání jednotlivých implementovaných sumarizačních metod a znázornění zlepšení souhrnu vygenerovaného těmito metodami. Experiment prokázal, že u sumarizace jednoho dokumentu je zlepšení mezi heuristickou a Luhnovou metodou značné. Toto zlepšení je způsobeno vážením termů. Oproti tomu využití latentní sémantické analýzy již takové zlepšení nepřináší. To znamená, že využití výpočetně složitější metody LSA nemá smysl, pokud máme k dispozici natrénovaný IDF slovník. Ovšem metoda LSA dokáže vytvářet souhrny i bez tohoto slovníku a dokonce i pro více-jazykové dokumenty. Stačí mít jen StopListy pro jazyky sumarizovaného článku,

Experiment také prokázal, že vytváření tématicky orientovaných souhrnů nepřináší žádné zlepšení a obecné souhrny odpovídají referenčním souhrnům stejně dobře jako souhrny, které jsou vytvořeny dle klíčových slov. Generování tématicky zaměřených souhrnů je provedeno zdvojnásobením frekvence klíčových termů článku.

6.4.2 Porovnání s online sumarizátory

Při hledání dostupných online sumarizátorů jsem narazil na problém kompatibility systémů s českým jazykem. Žádný online sumarizátor nedokázal zpracovat český článek přímo. Všechny sumarizátory měly problém s dělením vět souvislého textu. Nakonec byly nalezeny dva, které dokázaly zpracovat český text po rozdělení na věty a neměly problémy s českými znaky. Jednalo se o sumarizátory SMMRY[8] a N4T[9]. Oba sumarizátory fungují na statistickém principu, ale druhý poskytuje velmi dobré nastavení sumarizačního procesu. Posledním testovaným sumarizátorem byl Open Text Summarizer - OTS, který je dostupný

[8] <http://smmry.com/>

[9] <http://www.tools4noobs.com/summarize/>

v linuxových systémech z repozitáře nebo jinak je dostupný ze stránek SourceForge[10]. Podporuje vytváření souhrnů pro 37 jazyků včetně češtiny.

Tabulka 4: Výsledky evaluace volně dostupných metod a implementované Lsa

	ROUGE-1			ROUGE-2			ROUGE-W			ROUGE-SU		
	R	P	F	R	P	F	R	P	F	R	P	F
OTS	56,6	62,6	58,5	42,1	47,8	44,6	19,9	44,2	27,3	30,8	40,4	34,7
SMMRY	56,6	54,4	54,8	39,9	38,0	38,5	19,7	36,8	25,4	32,6	30,1	30,1
T4N	73,3	57,2	64,1	59,9	46,7	52,3	26,9	41,3	32,5	46,2	28,9	35,1
LSA	75,2	60,9	67,2	62,4	50,5	55,8	27,4	43,5	33,6	47,4	31,8	38,0

Jelikož většina testovaným sumarizátorů nebyla vytvářena pro český jazyk, dopadly testované systémy OTS a SMMRY relativně špatně (hůře než implementované metody). Oproti tomu sumarizátor T4No dopadl v porovnání implementovanými metodami dobře. Dosáhl podobného skóre jako Luhnova metoda nebo LSA. Jelikož společnost provozující sumarizátor, tento sumarizátor i prodává, tak jediné co se mi povedlo najít o principu sumarizace, byla tato žertovná zpráva: "Using some alien technology combined with the latest computers from NASA and some dwarves that read all the text your write, we manage to output an exact summary for any text given!".

6.4.3 Porovnání se souhrny neohebných jazyků

Pro potřeby tohoto experimentu bylo 20 procent evaluačního korpusu přeloženo do anglického jazyka a následně byly pomocí metod, které dovolují sumarizovat článek v jiném než anglické jazyce, vygenerovány souhrny. Jelikož referenční články byly napsány v českém jazyce, byly anglické souhrny přeloženy zpět do češtiny, dohledáním sumarizované věty v původním článku.

Jelikož Luhnova sumarizační metoda vyžaduje natrénování slovníku inverzní dokumentové frekvence termů, nebyla do tohoto experimentu zahrnuta.

Tento experiment byl původně navržen proto, aby byly porovnány ohebné (čeština) a neohebné (angličtina) jazyky. Ovšem v průběhu vývoje sumarizačního systému, byl do něj implementován lemmatizátor, který rozdíly odstranil a ohebný jazyk nakonec díky lemmatizaci termů dopadl lépe. Horší výsledky pro lemmatizovaný souhrny heuristické metody, jsou způsobené právě lemmatizací, která sjednotila nevýznamné termy o stejném

[10] `<http://sourceforge.net/projects/libots/>`

Tabulka 5: Porovnání výsledků pro anglický a český jazyk v procentech

	ROUGE-1			ROUGE-2			ROUGE-W			ROUGE-SU		
	R	P	F	R	P	F	R	P	F	R	P	F
Heur. slova	70,3	65,6	67,8	57,1	53,4	55,1	25,2	45,5	32,4	41,4	36,8	38,4
Heur. lemma	67,7	66,3	66,9	53,7	52,4	53,0	24,3	46,1	31,8	39,5	38,3	38,8
Heur ang.	73,4	67,4	70,2	61,3	56,3	58,6	26,3	46,9	33,7	44,9	38,4	41,2
LSA slova	76,3	64,5	69,9	64,9	54,8	59,4	27,8	45,6	34,6	46,6	33,6	39,0
LSA lemma	79,3	65,8	71,9	67,0	55,6	60,8	28,6	45,9	35,2	52,0	36,4	42,7
LSA angl.	69,4	60,4	64,4	54,9	47,6	50,9	24,6	46,9	30,9	40,1	30,4	34,3

lemmatu do jednoho termu s velkou četností. Tyto nevýznamné sjednocené termy, které nepokryl StopList, převážily ostatní termy a ovlivnily souhrny.

Při posuzování výsledků je ovšem mít na paměti, že anglické termy nebyly lemmatizovány. Toto je důvod výrazně horších výsledků pro anglickou LSA.

6.4.4 Globální vs. inverzní dokumentová frekvence

Během studia latentní sémantické analýzy jsem narazil na různé možnosti vážení termů v matici určené k dekompozici. Z těchto možností byly vybrány dvě: frekvence termu v dokumentu a inverzní dokumentová frekvence.

Tabulka 6: Globální frekvence vs. inverzní dokumentová frekvence

	ROUGE-1			ROUGE-2			ROUGE-W			ROUGE-SU		
	R	P	F	R	P	F	R	P	F	R	P	F
LSA-gf	75,2	60,8	67,2	62,4	50,5	55,8	27,4	43,5	33,6	47,4	31,8	38,0
LSA-idf	69,4	60,0	64,2	56,6	48,9	52,4	25,2	42,8	31,6	41,9	31,7	35,9

Výsledky experimentu jsou překvapivé. Předpoklad byl, že vážení termů hodnotou natrénovanou na velkém korpusu a obecnější znalost o termu (použit slovník vytvořený pro Luhnův sumarizátor), dopadne lépe než vážení termu na základě jeho frekvence v sumarizovaném dokumentu. Toto se ovšem nepotvrdilo, dvě metody vážení se v podstatě neliší, jen metoda nevyužívající IDF slovník měla o trochu lepší úplnost.

7 Závěr

Podle zadání práce bylo provedeno nastudování problematiky automatické sumarizace textových dokumentů převážně ze zahraniční literatury a vědeckých článků z konferencí. Nastudovány byly principy sumarizačních metod z počátků sumarizace v 60. letech až po moderní metody využívané dnes. Z těchto metod byly implementovány vybrané metody a byla z nich vytvořena knihovna umožňující provádět automatickou sumarizaci dokumentů.

Při práci na evaluaci metody bylo vyzkoušeno několik online sumarizátorů, ovšem žádný nedokázal sumarizovat český text, který mu byl předložen, aniž by byl proveden ručně preprocesing dat a některé sumarizátory nedokázaly zpracovat český text ani potom. Hlavním problémem bylo špatné zalamování textu na konci vět. Online sumarizátory nedokázaly správně rozdělit text na věty.

Problém preprocesingu textu řeší třída Preparation, která vstupní text dělí na věty, a pro český text ho dokáže i lemmatizovat. Data zpracovaná pomocí metod této třídy jsou následně sumarizována jednou z implementovaných metod.

7.1 Implementované metody

Implementovány byly celkem tři sumarizační metody. Jako první byla vytvořena metoda heuristická, která využívá nejpovrchnější znalosti o sumarizovaném textu. Touto znalostí je frekvence termů. Při vývoji této metody byla zároveň testována a vylepšována třída Preparation. Tato metoda je také jedna ze dvou metod lehce rozšiřitelných o možnosti sumarizace jiného než českého jazyka. Stačí vytvořit StopList pro požadovaný jazyk a pro neohebné jazyky metoda již více nevyžaduje. StopList obsahuje nejčastůjší termy jazyka, které jsou ze sumarizovaného textu odstraněny. Pro ohebné jazyky vyžaduje tato metoda navíc lemmatizátor, který ze slov textu vytvoří jejich základní tvar.

Další implementovanou metodou byla metoda vážící termy textu jejich inverzní dokumentovou frekvencí - IDF. Pro výpočet IDF bylo využito přes 2 miliony článků, na kterých byla zjištěna IDF pro každý term. Celkem byly vytvořeny dva slovníky s IDF termů. První slovník byl vytvořen na základě vstupního slovníku termů a byla k těmto termům jen spočtena jejich IDF. Druhým slovníkem byl slovník vytvořený dynamicky podle termů nalezených v článcích. Jelikož byly vytvořeny slovníky jen pro český jazyk, dokáže tato metoda kvalitně generovat jen české souhrny.

Poslední implementovanou metodou byla metoda využívající Latentní sémantické analýzy. Tato metoda převádí problém sumarizace textu na numerickou úlohu dekompozice matice. Jedná se o metodu, která vznikla inspirací z latentního sémantického indexování využívaného při vyhledávání dat v mohutných databázích. Implementovaná metoda využívá dvou způsobů vážení termů v dekomponované matici. Prvním způsobem je vážení termu přes jeho globální frekvenci v dokumentu. Tento způsob vážení lze využít pro jakýkoliv jazyk textu. Druhým je využití slovníku inverzní dokumentové frekvence.

7.2 Poznatky z experimentů

Pro potřeby evaluace systému byla vytvořena databáze referenčních souhrnů od anotátorů a sada nástrojů pro její vytvoření a úpravu. Tyto referenční souhrny byly využity k vyhodnocení experimentů.

Z experimentů vyplynulo několik důležitých poznatků o implementovaných metodách. Nejdůležitějším důsledkem je zjištění, že pro sumarizaci textu jednoho dokumentu je dostačující Luhnův sumarizátor, který dosahuje stejných výsledků jako metoda LSA, viz tabulka 3. Ovšem Luhnův sumarizátor vyžaduje natrénovaný slovník IDF. Také bylo zjištěno, že generovat tématicky zaměřené souhrny pro články objevující se běžně na zpravodajských serverech, nemá smysl, jelikož obsahují málo vět (průměrně 40) a tématické souhrny se v podstatě shodují s obecnými.

Prověřen byl vliv ohebnosti jazyka na výsledek sumarizace. Bylo zjištěno, že lemmatizace slov má na výsledek souhrnu malý vliv, ovšem výrazně přispívá k zjednodušení sumarizovaného textu a termy textu lépe odpovídají jejich významu a vlivu na článek. Například u LSA zmenšuje počet dimenzí, nebo u statistických metod zmenšuje slovník a více vyskloňovaných termů jednoho základu sjednocuje. Toto zefektivňuje proces výpočtu skóre vět.

Dále bylo vyzkoušeno několik volně dostupných sumarizátorů. Z nich byly vybrány 3 schopné zpracovat český text, ale jen jeden dosáhl kvality implementovaných sumarizačních metod. Byl to sumarizátor[11] provozující firma DreamHost Web Hosting.

Poslední provedený experiment prokázal, že není nutné pro sumarizaci využívat žádnou statisticky zjištěnou znalost o jazyku. Sumarizace využívající globální frekvenci termu v dokumentu k vážení termů má stejně dobré výsledky jako sumarizace, která využívá inverzní dokumentovou frekvenci.

[11] <http://www.tools4noobs.com/summarize>

7.3 Možné uplatnění

Sumarizační systém je možné využít k vytváření souhrnů dokumentů, které nevyžadují žádnou předchozí znalost problematiky. Také je možné provádět extrakci jejich hlavních témat pomocí referencí na věty, které tato témata nejlépe podchycují.

Dalším možným využitím je generování indikativních souhrnů pro potřeby RSS kanálů. Některé zpravodajské a odborné servery přidávají ke každému záznamu v RSS kanálu abstrakt textu článku. Většina serverů toto ovšem nedělá. Buď přidávají do RSS kanálu celý text článku, což ale zbytečně zvyšuje objem xml souboru, který musejí uživatelé stahovat, nebo mají u každého článku jen nadpis, což zase uživateli nedostatečně přiblíží obsah článku.

Celkem vzato byl vytvořen sumarizační systém, který lze použít v libovolném projektu, jenž vyžaduje provádění komprese informací.

7.4 Náměty k rozšíření práce

Během evaluace byla objevena jedna chyba při dělení textu na věty. Chyba byla způsobena nedostatečným slovníkem zkratek. Bylo by vhodné vytvořit rozsáhlejší slovník běžných zkratek (atd., resp., prof., ...) a celkově prostudovat hlouběji problematiku detekce začátku a konce vět. Toto téma by mohl řešit např. bakalářský nebo magisterský projekt.

Diplomovou práci by bylo možné rozšířit o problematiku multidokumentové a aktualizační sumarizace. Tyto dva typy sumarizace byly v práci již zmíněny a jejich výzkum je v dnešní době velmi aktuální. Díky velmi kvalitnímu hlasovému rozpoznávači, který je k dispozici na Ústavu informačních technologií a elektroniky by bylo možné také realizovat ASR sumarizátor. Ovšem realizaci tohoto typu sumarizátoru komplikuje problematika tečkování - tedy označení začátků a konců vět vložením tečky na konec věty. Toto je velmi obtížná úloha a až její vyřešení umožní realizaci ASR sumarizátoru.

8 Literatura

[1] ANDO, R. K. – LEE, L. Iterative Residual Rescaling: An Analysis and Generalization of LSI. In *Proceedings of the 24th SIGIR*, s. 154–162, 2001.

[2] AZMI, A. M. – AL-THANYYAN, S. *A text summarizer for Arabic* [online]. *Comput. Speech Lang.* 26, 4, s. 260–273. August 2012. ISSN 0885-2308. doi: 10.1016/j.csl.2012.01.002. Dostupné z: <http://dx.doi.org/10.1016/j.csl.2012.01.002>.

[3] BERRY, M. W. – BROWNE, M. *Understanding Search Engines*. Philadephia, PA : Society for Industrial and Applied Mathematics, 2005. doi: DOI:10.1137/1.9780898718164. Dostupné z: <http://dx.doi.org/10.1137/1.9780898718164>. ISBN 978-08-987-1816-4.

[4] BORKO, H. – BERNIER, C. *Abstracting concepts and methods*. Library and information science. : Academic Press, 1975. Dostupné z: <http://books.google.cz/books?id=PeawAAAAIAAJ>. ISBN 978-0-121-18650-0.

[5] DORR, B. et al. Extrinsic Evaluation of Automatic Metrics for Summarization. Technical report, University of Maryland, College Park and BBN Technologies, July 2004. Dostupné z: <http://lampsrv02.umiacs.umd.edu/pubs/TechReports/LAMP_115/LAMP_115.pdf>.

[6] FURUI, S. Advances in automatic speech summarization. In *Proceedings of the 7th European Conference on Speech Communication and Technology*, s. 1771–1774, 2001.

[7] GANESAN, K. *Basics of Setting up ROUGE Toolkit for Evaluation of Summarization Tasks* [online]. 2010. Dostupné z: <http://kavita-ganesan.com/rouge-howto>.

[8] GARCIA, E. *Singular Value Decomposition (SVD) A Fast Track Tutorial* [online]. *Compute*. s. 5. 2006. Dostupné z: <http://www.cs.fit.edu/~dmitra/SciComp/Resources/singular-value-decomposition-fast-track-tutorial.pdf>.

[9] KUPIEC, J. – PEDERSEN, J. – CHEN, F. A trainable document summarizer. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '95, s. 68–73, New York, NY, USA,

1995. ACM. doi: 10.1145/215206.215333. Dostupné z: <http://doi.acm.org/10.1145/215206.215333>. ISBN 0-89791-714-6.

[10] KŘIŠŤAN, M. Multidokumentový sumarizátor textů. Master's thesis, Západočeská univerzita v Plzni, 2007.

[11] LEWIS, D. Naive (Bayes) at forty: The independence assumption in information retrieval. In NéDELLEC, C. – ROUVEIROL, C. (Ed.) *Machine Learning: ECML-98*, 1398 / *Lecture Notes in Computer Science*, s. 4–15. Chemnitz, DE: Springer Berlin / Heidelberg, 1998. Dostupné z: <http://dx.doi.org/10.1007/BFb0026666>. ISBN 978-3-540-64417-0, 10.1007/BFb0026666.

[12] LIN, C.-Y. ROUGE: A Package for Automatic Evaluation of summaries. In *Proceedings ACL workshop on Text Summarization Branches Out*, s. 10, 2004. Dostupné z: <http://research.microsoft.com/~cyl/download/papers/WAS2004.pdf>.

[13] LUHN, H. P. *The automatic creation of literature abstracts* [online]. *IBM J. Res. Dev.* 2, 2, s. 159–165. April 1958. ISSN 0018-8646. doi: 10.1147/rd.22.0159. Dostupné z: <http://dx.doi.org/10.1147/rd.22.0159>.

[14] MANI, I. – MAYBURY, M. *Advances in Automatic Text Summarization*. Cambridge, MA, USA : Mit Press, 1999. Dostupné z: <http://books.google.cz/books?id=YtUZQaKDmzEC>. ISBN 9780262133593.

[15] MANN, W. C. – THOMPSON, S. A. *Rhetorical structure theory: Toward a functional theory of text organization* [online]. *Text*. 8, 3, s. 243–281. 1988.

[16] MANNING, C. D. – RAGHAVAN, P. – SCHTZE, H. *Introduction to Information Retrieval*. New York, NY, USA : Cambridge University Press, 2008. Dostupné z: <http://nlp.stanford.edu/IR-book/>. ISBN 0521865719, 9780521865715.

[17] MARCU, D. From Discourse Structures to Text Summaries. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, s. 82–88, 1997.

[18] MIHALCEA, R. – TARAU, P. TextRank: Bringing Order into Texts. In *Conference on Empirical Methods in Natural Language Processing*, Barcelona, Spain, 2004. Dostupné z: <http://acl.ldc.upenn.edu/acl2004/emnlp/pdf/Mihalcea.pdf>.

[19] RADEV, D. R. et al. Evaluation challenges in large-scale document summarization. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, s. 375–382, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics. doi: 10.3115/1075096.1075144. Dostupné z: <http://dx.doi.org/10.3115/1075096.1075144>.

[20] SIEGEL, S. – CASTELLAN, N. *Nonparametric statistics for the behavioral sciences*. McGraw-Hill international editions. Statistics series. New York : McGraw-Hill, 1988. Dostupné z: <http://books.google.cz/books?id=bq3uAAAAMAAJ>. ISBN 9780070573574.

[21] STEINBERGER, J. – JEŽEK, K. *Evaluation measures for Text summarization* [online]. *Computing and Informatics*. 28, 2, s. 251–275. 2009a. ISSN 1335-9150. Dostupné z: <http://www.cai.sk/ojs/index.php/cai/article/viewFile/37/24>.

[22] STEINBERGER, J. – JEŽEK, K. Text summarization and singular value decomposition. In *Proceedings of the Third international conference on Advances in Information Systems*, ADVIS'04, s. 245–254, Berlin, Heidelberg, 2004. Springer-Verlag. doi: 10.1007/978-3-540-30198-1_25. Dostupné z: <http://dx.doi.org/10.1007/978-3-540-30198-1_25>. ISBN 978-3-540-23478-4.

[23] STEINBERGER, J. – JEŽEK, K. Update summarization based on novel topic distribution. In *Proceedings of the 9th ACM symposium on Document engineering*, DocEng '09, s. 205–213, New York, NY, USA, 2009b. ACM. doi: 10.1145/1600193.1600239. Dostupné z: <http://doi.acm.org/10.1145/1600193.1600239>. ISBN 978-1-60558-575-8.

[24] TUR, G. – DE MORI, R. *Spoken Language Understanding: Systems for Extracting Semantic Information from Speech*. : John Wiley & Sons, 2011. Dostupné z: <http://books.google.cz/books?id=RDLyT2FythgC>. ISBN 978-1-119-99394-0.

[25] YIHONG, G. – XIN, L. Generic text summarization using relevance measure and latent semantic analysis. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2001.

A Ukázka souhrnů

Text článku:

Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda da Vinciho objevilo ve svých sbírkách muzeum Prado ve španělské metropoli Madridu. Obraz vznikl hned ve stejné době jako originál a namaloval jej v Leonardově ateliéru jeden z jeho žáků. Na obraze je portrét velmi připomínající Giocondu, jež vznikla v letech 1503-06; krajina v pozadí je na něm však pokryta jakousi patinou času, tvoří ji šedavá skaliska. Autor byl vždy považován za neznámého, vědělo se pouze, že dílo vzniklo zhruba v první čtvrtině 16. století. Prado o obrazu nechtělo oficiálně mluvit dřív, než ho detailně prozkoumá. Důkladné restaurační práce pak odhalily, že obraz vznikl v samotné Leonardově dílně - a pravděpodobně ve stejném období, kdy italský mistr pracoval na originálu. Kopie byla skryta pod přemalbou, takže se dlouho mělo za to, že vznikla až dlouho po Leonardově smrti. Odborníci ale nevylučují, že da Vinciho žák "dokumentoval"Giocondu přímo ve stejnou dobu, kdy ji mistr maloval. A tato "souběžná"kopie je nyní v majetku madridské galerie. Podle magazínu The Art Newspaper existuje hodně kopií ze 16. a 17. století, ale tento objev "úžasně změní naše porozumění nejslavnějšímu obrazu světa". Originál visí v Louvre a jeho zkoumání je příliš velkým rizikem, neboť je velmi křehký a pokrytý několika vrstvami popraskaného laku. Další výzkum kopie by mohl odhalit například detaily toscánské krajiny; z obrazu se také zdá, že dívka, jež stála modelem, je mladší, než se původně myslelo. Může jí prý být jen lehce přes dvacet - což se při pohledu na originál nezdá. Podle vedoucího restaurátora v Pradu Gabriela Finaldiho dáma na originálu vypadá starší kvůli zašpinění obrazu. "Když jsou malby zašpiněné, osoby mají tendenci vypadat starší,"vysvětlil. Stav uchování madridské Mony Lisy je každopádně mnohem lepší nežli originálu, který přitahuje davy. Badatelům to umožní studovat lépe originální obraz a mohou se pokusit dešifrovat i další záhady, jež ji stále obklopují. Tato kopie Giocondy se dostala do španělské královské sbírky v roce 1966. "Jde jednoznačně o dvojče originálu. Jedna zřejmě byla svědkem zrození druhé,"říká Miguel Falomir, šéf pradského oddělení francouzského a italského malířství (do roku 1700). Madridská Mona Lisa bude vystavena v Louvru od 21. února v rámci výstavy o Leonardově Santa Aně.

Heuristická metoda:

Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda da Vinciho objevilo ve svých sbírkách muzeum Prado ve španělské metropoli Madridu.

Další výzkum kopie by mohl odhalit například detaily toscánské krajiny; z obrazu se také zdá, že dívka, jež stála modelem, je mladší, než se původně myslelo.

Stav uchování madridské Mony Lisy je každopádně mnohem lepší nežli originálu, který přitahuje davy.
Madridská Mona Lisa bude vystavena v Louvru od 21. února v rámci výstavy o Leonardově Santa Aně.
Podle magazínu The Art Newspaper existuje hodně kopií ze 16. a 17. století, ale tento objev "úžasně změní naše porozumění nejslavnějšímu obrazu světa".

Luhnův sumarizátor:

Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda da Vinciho objevilo ve svých sbírkách muzeum Prado ve španělské metropoli Madridu.
Důkladné restaurační práce pak odhalily, že obraz vznikl v samotné Leonardově dílně - a pravděpodobně ve stejném období, kdy italský mistr pracoval na originálu.
Na obraze je portrét velmi připomínající Giocondu, jež vznikla v letech 1503-06; krajina v pozadí je na něm však pokryta jakousi patinou času, tvoří ji šedavá skaliska.
Originál visí v Louvre a jeho zkoumání je příliš velkým rizikem, neboť je velmi křehký a pokrytý několika vrstvami popraskaného laku.
Stav uchování madridské Mony Lisy je každopádně mnohem lepší nežli originálu, který přitahuje davy.

Latentní sémantická analýza:

Badatelům to umožní studovat lépe originální obraz a mohou se pokusit dešifrovat i další záhady, jež ji stále obklopují.
Na obraze je portrét velmi připomínající Giocondu, jež vznikla v letech 1503-06; krajina v pozadí je na něm však pokryta jakousi patinou času, tvoří ji šedavá skaliska.
Důkladné restaurační práce pak odhalily, že obraz vznikl v samotné Leonardově dílně - a pravděpodobně ve stejném období, kdy italský mistr pracoval na originálu.
Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda da Vinciho objevilo ve svých sbírkách muzeum Prado ve španělské metropoli Madridu.
Obraz vznikl hned ve stejné době jako originál a namaloval jej v Leonardově ateliéru jeden z jeho žáků.

B Překrytí referencí

Tabulka 7: Překrytí referenčních vět článku z přílohy A.

uživatelů	vybraná věta
11	Podle všeho nejstarší známou kopii slavné Mony Lisy od Leonarda da Vinciho objevilo ve svých sbírkách muzeum Prado ve španělské metropoli Madridu.
8	Důkladné restaurační práce pak odhalily, že obraz vznikl v samotné Leonardově dílně - a pravděpodobně ve stejném období, kdy italský mistr pracoval na originálu.
7	Obraz vznikl hned ve stejné době jako originál a namaloval jej v Leonardově ateliéru jeden z jeho žáků.
6	Originál visí v Louvre a jeho zkoumání je příliš velkým rizikem, neboť je velmi křehký a pokrytý několika vrstvami popraskaného laku.
4	Další výzkum kopie by mohl odhalit například detaily toscánské krajiny; z obrazu se také zdá, že dívka, jež stála modelem, je mladší, než se původně myslelo.
4	Na obraze je portrét velmi připomínající Giocondu, jež vznikla v letech 1503-06; krajina v pozadí je na něm však pokryta jakousi patinou času, tvoří ji šedavá skaliska.
4	Madridská Mona Lisa bude vystavena v Louvru od 21. února v rámci výstavy o Leonardově Santa Aně.
3	Podle magazínu The Art Newspaper existuje hodně kopií ze 16. a 17. století, ale tento objev "úžasně změní naše porozumění nejslavnějšímu obrazu světa".
3	Odborníci ale nevylučují, že da Vinciho žák "dokumentoval"Giocondu přímo ve stejnou dobu, kdy ji mistr maloval.
2	Kopie byla skryta pod přemalbou, takže se dlouho mělo za to, že vznikla až dlouho po Leonardově smrti.
2	Autor byl vždy považován za neznámého, vědělo se pouze, že dílo vzniklo zhruba v první čtvrtině 16. století.
1	Stav uchování madridské Mony Lisy je každopádně mnohem lepší nežli originálu, který přitahuje davy.