Automatická predikce psaného textu

Martin Kotěna

Bakalářská práce
2017

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2016/2017

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Martin Kotěna**
Osobní číslo:	**A13169**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Forma studia:	**prezenční**
Téma práce:	**Automatická predikce psaného textu**
Téma anglicky:	**The Automatic Prediction of Written Texts**

Zásady pro vypracování:

1. **Vypracujte literární rešerši na dané téma.**
2. **Prověřte sběr dat z internetových zdrojů, jejich následná úprava, filtrace a očištění.**
3. **Vypracujte analýzu textu (bigramy, trigramy) a podmíněných pravděpodobností.**
4. **Vytvořte prediktivní model.**
5. **Vytvořte prezentaci dané problematiky.**

Rozsah bakalářské práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. **KUCKARTZ, Udo. Qualitative text analysis: a guide to methods practice & using software. Los Angeles: Sage, c2014, xvii, 173 s. ISBN 978-1-4462-6775-2.**
2. **GÓMEZ RODRÍGUEZ, Carlos. Parsing schemata for practical text analysis. London: Imperial College Press, c2010, xiv, 275 s. Mathematics, computing, language and life: frontiers in mathematical linguistics and language theory. ISBN 978-1-84816-560-1.**
3. **POPPING, Roel. Computer-assisted text analysis. London: Sage Publications, 2000, x, 229 s. New technologies for social research. ISBN 0-7619-5379-5.**
4. **WEISS, Sholom M. Text mining: predictive methods for analyzing unstructured information. New York: Springer, 2005, xii, 237 s. ISBN 978-0-387-34555-0.**
5. **HAN, Jiawei, Micheline KAMBER a Jian PEI. Data mining: concepts and techniques. 3rd ed. Waltham: Elsevier, c2012, xxxv, 703 s. ISBN 978-0-12-381479-1.**
6. **SKALSKÁ, Hana. Data mining a klasifikační modely. Vyd. 1. Hradec Králové: Gaudeamus, 2010, 154 s. ISBN 978-80-7435-088-7.**
7. **WEISS, Sholom M., Nitin INDURKHYA a Tong ZHANG. Fundamentals of predictive text mining. London: Springer, 2010, xiii, 226 s. Texts in computer science. ISBN 978-1-84996-226-1.**
8. **HÁJEK, Martin. Čtenář a stroj: vybrané metody sociálněvědní analýzy textů. Praha: Sociologické nakladatelství (SLON), 2014, 226 s. Studie. ISBN 978-80-7419-161-9.**

Vedoucí bakalářské práce: **doc. Ing. Roman Šenkeřík, Ph.D.**
Ústav informatiky a umělé inteligence

Datum zadání bakalářské práce: **24. února 2017**

Termín odevzdání bakalářské práce: **24. května 2017**

Ve Zlíně dne 24. února 2017

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové/bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne 20.05.2017

.....
podpis diplomanta

ABSTRAKT

Cílem této předložené bakalářské práce je návrh a implementace nástroje pro predikci psaného textu spojeného s vypracováním databázového systému. Teoretická část je věnována obecnému popisu predikce textu, programovacích jazyků a použitého databázového systému. V praktické části práce jsou podrobně rozebrány principy fungování výše zmíněného nástroje. Zároveň je zde popsáno vytvoření databáze se statistickým zpracováním dat dle Bayesovy pravděpodobnosti a vypracování webového rozhraní s uživatelským přístupem. Schopnost tohoto nástroje tedy spočívá nejen v rychlé predikci celých slov i frází, ale také v podložení těchto nabídek předchozí analýzou rozsáhlých textů z různých zdrojů. Součástí bakalářské práce je CD s popisovanou programovou implementací a vypracovanou databází.

Klíčová slova: predikce textu, bigram, trigram, klient-server, databáze

ABSTRACT

The aim of this thesis is to design and implement a tool for text prediction associated with the development of a database system. The theoretical part of this work is devoted to general description of text prediction, programming languages and used database system. The practical part is focused on the principles of the aforementioned instrument. The database system with statistical data processing according to Bayes‘s probability is also described there together with the graphical user interface. The ability of this tool is based not only on quick prediction of full words or phrases, but also on backing up these offers by prior analysis of extensive texts from various sources. This thesis also contains a CD with described program implementation and prepared database.

Keywords: text prediction, bigram, trigram, client-server, database

Za odborné vedení, připomínky a návrhy při vypracování této bakalářské práce bych chtěl poděkovat svému vedoucímu bakalářské práce doc. Ing. Romanu Šenkeříkovi, Ph.D.

Zároveň bych chtěl také poděkovat Ing. Tomáši Urbánkovi za podnětné připomínky k praktické části této práce.

"Talk is cheap. Show me the code."

- Linus Torvalds

Prohlašuji, že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD

Predikce textu je jednou z nejvyužívanějších softwarových aplikací využívaných zejména v mobilních zařízeních. Výrobci těchto zařízení upustili od klasických klávesnic a dlouho využívali 12 tlačítek, kde daný počet stisknutí vytvořil znak. Takovýmto způsobem bylo možné na jedno tlačítko přiřadit teoreticky nekonečné množství znaků.

Nevýhodou tohoto řešení byla skutečnost, že pro vytvoření gramaticky správné věty bylo potřeba velké množství stisknutí, a jedna chyba znamenala tvorbu znaku opět od začátku. Zároveň byla tato metoda i časově náročná.

Další vývoj přišel s dotykovými obrazovkami, které zobrazují digitální klávesnici. Uživatel pak píše poklepáním na zobrazené tlačítko. Jednou z prvních metod bylo „vyťukávání znaků“, přičemž uživatel mohl přepínat mezi druhy klávesnic, které obsahovaly tradiční písmena nebo zavináč, otazník, a další. Napsat slovo ale stále trvalo velmi dlouho.

Z tohoto důvodu vznikl komplexní software schopný predikovat i opravit slovo, které měl uživatel s největší pravděpodobností na mysli. Tato funkce výrazným způsobem urychluje psaní dokumentů, zpráv, či nabízí vhodná hesla pro vyhledávání na internetu.

Problémem při napovídání konkrétního slova je absence programových nastavení. To znamená, že program nedokáže přesně určit uživatelem hledané slovo bez předchozích vstupních dat. Některé programy se dokážou samy učit statistickým zpracováním dřívějších textů. Dalším způsobem je použití předpřipravené databáze vypracované statistickým zhodnocením psaných textů velkého souboru uživatelů.

Cílem této práce je vytvoření implementace pro predikci psaného textu založeného na doplňování frází využívající předpřipravenou databázi slov. Na uživatelský vstup program reaguje vrácením všech slov, která kdy následovala po zadaném vstupu, přičemž je řadí sestupně na základě Bayesovy pravděpodobnosti.

Pro vypracování zadání byly použity programovací jazyky Java pro analýzu a zápis vstupních dat do databáze, JavaScript pro vypracování uživatelského rozhraní, PHP pro komunikaci programu s databází, HTML pro vytvoření a CSS pro úpravu stránky.

I. TEORETICKÁ ČÁST

1 PREDIKCE TEXTU

1.1 Definice

Pojem predikce pochází z latinských slov „prae – před“ a „dicere – říkat“. Význam tohoto slova je tedy předpověď toho, co by mohlo nastat. Ačkoliv by se mohlo na první pohled zdát, že se jedná o nějaký druh nepodloženého věštění nebo odhadování, ve skutečnosti tomu tak není. Predikce textu totiž využívá sofistikované a komplexní sady algoritmů, které na základě statistických teorií dokáží s vysokou pravděpodobností „předpovědět“ slovo nebo frázi, kterou měl uživatel skutečně na mysli. [1]

1.2 Historie predikce textu

Počátky predikce textu jsou spjaty s rozvojem elektroniky, zejména poté s telefony a jejich zabudovanou telefonní klávesnicí. Tyto klávesnice využívaly a stále využívají softwarové algoritmy propojené zejména se zabudovaným slovníkem. Výše zmíněný algoritmus dokáže v tomto slovníku vyhledávat a rychle uživateli nabízet seznam možných slov. Tato slova program vybírá na základě nejvyšší shody mezi uživatelským vstupem a daným nejpodobnějším slovem ve slovníku. Tento typ predikce je znám od roku 1970 a byl patentována v roce 1985. [1]

Druhý nejčastěji používaný systém je systém se zabudovaným slovníkem syntaxí. Tento systém ke své správné funkci využívá statistický výskyt slov. Jinými slovy je založen na pravděpodobnosti výskytu řetězce slov v jazyce nebo v kontextu. Řetězcem mohou tedy být nejen písmena, ale také celá slova. Tento systém tedy dokáže opravit i chybějící interpunkci nebo shodu podmětu s přísudkem. [1]

Oba výše zmíněné typy predikce textu by měly obsahovat určitý druh databáze, ve které se budou slova postupně shromažďovat. Platí ovšem, že čím je databáze obsáhlejší, tím je větší pravděpodobnost správné nabídky. [1]

1.3 Princip funkce

Nejčastěji se s predikcí textu setkáváme při automatickém doplňování textu, což je jedna ze základních funkcí telefonu nebo počítače. Tato funkce tedy na základě určitého vstupu dokáže doplnit zbytek slova nebo fráze. V praxi uživatel této funkce dostane na výběr z několika možností na základě vstupu, což většinou bývá několik prvních znaků slova. Uživatel má možnost si jednu z nabízených možností vybrat, nebo pokračovat v psaní textu sám.

1.4 N-Gramy

Takzvané „N-Gramy“ jsou velmi často používané v konkrétních aplikacích využívající predikci textu. Jedná se o soubor po sobě jdoucích slov v určité frázi, přičemž se použitý program při stanovení „N-Gramu“ pohybuje nejčastěji po jednom slově. [2]

Počet „N-Gramů“ v jakékoli uvažované větě se dá předem vypočítat dle vzorce (1).

$$Y = X - (N - 1)\,, \tag{1}$$

kde: Y – počet N-Gramů v dané větě

X – počet slov v dané větě

N – úroveň N-Gramu (bigram N = 2, atd.)

Nejčastěji se používají bigramy či trigramy z důvodu optimalizace nabídky možností.

1.4.1 Bigram

Bigram je podmnožina „N-Gramů“ a vyznačuje se tím, že je úrovně N = 2. Jinými slovy, bigram je tvořen právě dvěma slovy. [2]

Pro vypracování bigramů byla zadána následující věta:

„Chundelatá ovce skotačí po zeleném travnatém poli.“

Pokud tedy uvažujeme „N-Gram“ úrovně N = 2, jinými slovy tedy bigram, potom budou bigramy vzniklé z této věty následujícího charakteru.

1. Chundelatá ovce
2. ovce skotačí
3. skotačí po
4. po zeleném
5. zeleném travnatém
6. travnatém poli

Z tohoto případu je patrné, že ze zadané věty bylo vytvořeno celkem šest bigramů. Pro vygenerování následujícího bigramu je nutné se pohnout o jedno slovo dopředu.

1.4.2 Trigram

Trigram je v principu shodný s bigramem, ale na rozdíl od něj obsahuje právě tři slova. [2]

Tento rozdíl se nejlépe dá demonstrovat na té samé větě, která byla použita pro vypracování bigramu.

„Chundelatá ovce skotačí po zeleném travnatém poli.“

1. Chundelatá ovce skotačí
2. ovce skotačí po
3. skotačí po zeleném
4. po zeleném travnatém
5. zeleném travnatém poli

Ze stejně zadané věty bylo vytvořeno celkem pět trigramů, přičemž stále platí podmínka pohybu o jedno slovo dopředu.

1.4.3 Praktické použití N-Gramů

N-Gramy se dají využít v různorodých aplikacích. Své využití nalézají například při přípravách nových jazykových modelů. Největší softwarové firmy jako je například Google a Microsoft se zasloužily o markantní rozvoj těchto nástrojů, které se dnes hojně využívají v různých aplikacích, jako jsou oprava překlepů, dělení slov a sumarizace textu. [2]

1.5 Použití a aplikace predikce textu

Nejčastěji se s predikcí textu setkáváme při zadávání emailových adres, vyhledávání jmen v telefonním seznamu, při vytváření programu v programovacím jazyce, ale také při vyplňování různých formulářů.

Vedle výše zmíněných možností využití predikce textu ji lze také významně zrychlit zadávání textu a ve výsledku tak urychlit komunikaci mezi člověkem a počítačem.

Každá aplikace predikce textu (též známá jako „našeptávač“) nabízí své nesporné výhody. Dá se využít jako pomocník s pravopisem, jako inteligentní vyhledávač na různých serverech či jako pomocník pro dotvoření myšlenky a použití vhodného slova v kontextu.

Všechny tyto aplikace vedou ke snížení počtu úhozů na klávesnici, které jsou potřeba k zadání námi požadovaného textu. Použití predikce tedy zrychluje práci a zvyšuje efektivitu využití času. [2]

2 KLÍČOVÉ VLASTNOSTI SYSTÉMU PRO PREDIKCI TEXTU

Ačkoliv existují různorodé aplikace systému predikce textu, všechny se vyznačují specifickými vlastnostmi, které tyto systémy musí splňovat.

- Rychlost a nenáročnost
- Efektivita
- Snadnost použití
- Všestrannost
- Nezávislost na použitém jazyku

Predikce textu má pouze poradní charakter. Uživatel si sám může zvolit, co mu v dané chvíli lépe vyhovuje. Při psaní volného textu, kdy uživatel ví, co chce napsat, je lepší a rychlejší pokračovat ve vlastním psaní, ale například při vyhledávání na internetu si uživatel rád počká na různé možnosti a z nich si poté vybere, co je pro něj nejlepší.

2.1 Rychlost a nenáročnost

Každý systém predikce textu musí okamžitě reagovat na vstup, přičemž nesmí příliš vyčerpávat omezené výpočetní zdroje telefonů nebo tabletů. Uživatel musí mít k dispozici výsledek co nejrychleji. V opačném případě tato funkce postrádá smysl z hlediska úspory času.

2.2 Efektivita

Rychlost reakce zmíněná výše je bezpředmětná, pokud nám predikce textu není schopna poskytnout relevantní zpětnou vazbu. Toto se dá řešit vypracováním kvalitní zdrojové databáze, která je využívaná predikcí textu. Čím je tato databáze obsáhlejší, tím je predikce textu efektivnější.

2.3 Snadnost použití

Systémy predikce textu musejí být nabízeny v takové formě, aby je koncový uživatel dokázal efektivně využít. Koncoví uživatelé nemusí být pouze lidé, kteří jsou počítačovými techniky. Může se jednat o studenty, starší lidi a jiné běžné uživatele. Z tohoto důvodu musí být rozhraní intuitivní a snadno použitelné. Pokud tomu tak není, uživatelé si obstarají takové programy, které jim vyhovují.

2.4 Všestrannost

Systém neopravuje jen pravopisné chyby, ale také interpunkci, kontext nebo může nabídnout i alternativní slovo k použitému.

2.5 Nezávislost na použitém jazyce

Principy vytvořeného systému musí fungovat v jakémkoliv používaném jazyce a nejlépe nezávislé na platformě, aby nebylo nutné programovat pro každý světový jazyk a elektronické zařízení (telefon, počítač, tablet).

3 SMYSL AUTOMATICKÉHO DOPLŇOVÁNÍ PRO UŽIVATELE

Smysl pro koncového uživatele je zejména v tom, že existuje zpětná vazba mezi prediktivním systémem a uživatelem. To znamená, že se systém dokáže svým způsobem „učit“, čímž poskytuje slova, zkratky nebo slovní spojení, která uživatel často používal a psal. Toto pomáhá zejména autorům článků.

Handicapovaní lidé mohou mít problémy s normální klávesnicí. Pro tyto účely je možné použít psaní pomocí pohybu očí. Hraje zde velkou roli predikce textu, neboť napsání daného slova po jednotlivých písmenech by bylo velmi náročné. [3]

Mezi další použití patří kontrola pravopisu nebo zrychlené psaní, kdy predikce opravuje překlepy a nabízí vhodná slova.

Predikce textu se dá využít jako zjednodušení zápisu. Nejlépe je to vidět na mobilních telefonech s dotykovým displejem, kdy má uživatel dvě varianty. První možností je kombinace automatické opravy s predikcí slov, která nabídne opravu překlepu, pokud uživatel udělá chybu. Druhá možnost je volné psaní s vypnutou predikcí textu. [3]

Predikce textu má využití jak pro osoby, které mají problémy se zadáváním textu, tak i pro běžné uživatele. Mezi hlavní výhody patří kontrola pravopisu, inspirace a zrychlené psaní.

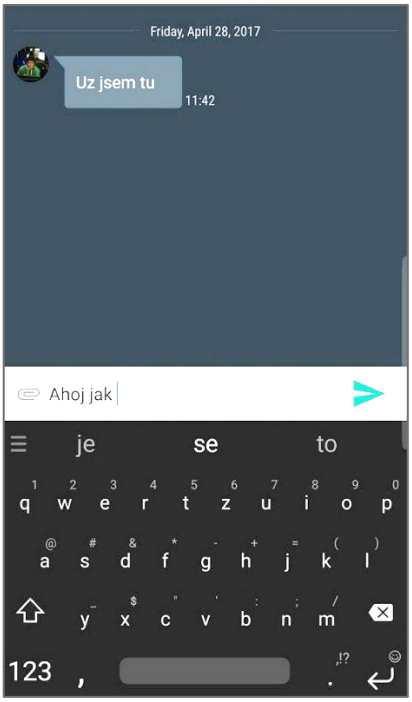
4 EXISTUJÍCÍ ŘEŠENÍ

Pro predikci textu existuje mnoho různých řešení. Nejvíce je využívána v mobilních telefonech.

4.1 Aplikace pro mobilní telefony

4.1.1 SwiftKey Keyboard

Jedná se o prediktivní klávesnici, která se dokáže naučit styl psaní uživatele, včetně oblíbených frází, emotikonů a důležitých slov. Díky tomu se adaptuje jak slovník, tak i prediktivní text na individuální potřeby uživatele. [4]



Obr. 1. SwiftKey Keyboard [4]

4.1.2 Eye-type

Tato softwarová klávesnice slouží zejména pro tělesně handicapované uživatele. Velká tlačítka jsou ovládána očima uživatele. Takový typ aplikace z mobilního telefonu lze vidět na obr. 2. Tento způsob vyžaduje k ovládání velká tlačítka, proto je jich omezený počet. Na každém tlačítku je více písmen. Každé písmeno má svou souřadnici, která napomáhá k vybrání písmene očima uživatele. Systém predikce textu potom předpoví zadávané slovo, které může uživatel potom přijmout nebo si vybrat z alternativní nabídky. [5]

Obr. 2. Eye-Type [5]

4.1.3 Našeptávač Google Instant

Tento našeptávač je produktem světové firmy Google. Tato aplikace předpovídá, co daný uživatel hledá a nabízí možné výsledky, zatímco uživatel píše. V průběhu psaní se nabízené odpovědi mění.

Tento systém výrazně šetří uživatelský čas pro zadání hledaného hesla, nabízí lepší možnosti díky analýze všech uživatelských vstupů, které kdy byly použity, nabízí okamžité výsledky a uživatel vidí možnosti, aniž by musel na cokoli klikat.

Google Instant dokáže měnit nabízené možnosti i dle místa, odkud uživatel vyhledává. Pokud tedy uživatel zadá heslo „hotely“, tato aplikace může nabídnout hotely ve Zlíně nebo v Brně. Zároveň také analyzuje historii vyhledávání uživatele a jako první nabízí výsledky, které uživatel hledal vícekrát. Zároveň je tato aplikace ošetřena, aby nezobrazovala výsledky považované za nevhodné.

4.2 Bayesova statistika

Thomas Bayes byl anglickým duchovním, který žil v 18. století. Dnes je znám hlavně díky formulaci tzv. Bayesovy věty, která se týká podmíněné pravděpodobnosti. Tato metoda se zabývá statistickým zpracováním dat. Popisuje pravděpodobnost, že nastane *jev A*, za současné znalosti podmíněných jevů, které mohou mít na *jev A* vliv. Jinými slovy, pokud existuje vztah mezi váhou a výskytem srdečních onemocnění, použitím Bayesovy metody se dá váha osoby použít k přesnějšímu odhadu pravděpodobnosti, zda jedinci hrozí srdeční onemocnění.

Jedná se tedy o statistiku, která zpřesňuje pravděpodobnost hypotéz dle dalších relevantních jevů. Praktický výpočet se řídí dle Bayesovy věty (2).

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}, \tag{2}$$

kde: P(A|B) – Pravděpodobnost jevu A za podmínky, že nastal *jev B*.

P(B|A) – Pravděpodobnost jevu B za podmínky, že nastal *jev A*.

P(A) – Pravděpodobnost jevu A.

P(B) – Pravděpodobnost jevu B.

Běžná statistika stanovuje pravděpodobnost na základě zpracování jevů, které již v minulosti nastaly. Naproti tomu se Bayesova statistika používá tam, kde není dostatek minulých dat. Z tohoto důvodu se používá zejména tam, kde se pracuje s „nejistými jevy" – testování nových léků, managementu, odhadování jevů. [6]

Pro zobrazení pravděpodobnostních vztahů mezi jednotlivými jevy se využívá pravděpodobnostní model, nebo také jinak Bayesovská síť. Jde vlastně o acyklický orientovaný graf. Každý uzel v tomto grafu odpovídá jedné náhodné veličině. Tento graf obsahuje několik uzlů a hrany mezi uzly potom zobrazují pravděpodobnostní závislost mezi vybranými veličinami.

Postup výpočtu je demonstrován na náhodně vybraných hodnotách, které jsou sestaveny do tabulky 1.

KLÍČ	HODNOTA
Ahoj	Jak
Ahoj	Jak
Ahoj	Tome
Ahoj	Petře
Pokus	Jak

Tab. 1. Hodnoty pro Bayese

- P(A) -> P(Jak) = $\frac{3}{5}$ –> pravděpodobnost kolikrát se slovo „jak" objevilo ve sloupci hodnota

- P(B|A) -> P(Ahoj|Jak) = $\frac{2}{4}$ -> Číslo 2 nám značí kolikrát hodnota „jak“ tvoří dvojici s klíčem „ahoj“
- P(B) -> P(Ahoj) = $\frac{8}{20}$ -> Součet všech možností vytvoření dvojicí s klíčem „ahoj“
- P(B) = P(Ahoj|Jak)*P(Jak) + P(Ahoj|Tome)*P(Tome) + P(Ahoj|Petře)*P(Petře)

$$P(Jak|Ahoj) = \frac{P(Ahoj|Jak) * P(Jak)}{P(Ahoj)} = \frac{\frac{2}{4} * \frac{3}{5}}{\frac{2}{4} * \frac{3}{5} + \frac{1}{4} * \frac{1}{5} + \frac{1}{4} * \frac{1}{5}} = \frac{\frac{3}{10}}{\frac{8}{20}} = \mathbf{0,75}$$

$$P(Tome|Ahoj) = \frac{P(Ahoj|Tome) * P(Tome)}{P(Ahoj)} = \frac{\frac{1}{4} * \frac{1}{5}}{\frac{1}{4} * \frac{1}{5} + \frac{2}{4} * \frac{3}{5} + \frac{1}{4} * \frac{1}{5}} = \frac{1}{8} = \mathbf{0,125}$$

$$P(Petře|Ahoj) = \frac{P(Ahoj|Petře) * P(Petře)}{P(Ahoj)} = \frac{\frac{1}{4} * \frac{1}{5}}{\frac{1}{4} * \frac{1}{5} + \frac{1}{4} * \frac{1}{5} + \frac{2}{4} * \frac{3}{5}} = \frac{1}{8} = \mathbf{0,125}$$

Ukázkový příklad:

Jako příklad může posloužit dveřní rám v obchodě, který by se měl rozezvučet, pokud se pokusí projít zloděj se zbožím, které nemá deaktivovaný čip. Systém spustí alarm v 95 % případů, kdy prochází někdo s kradeným zbožím. V 5 % se ale deaktivace čipu z nějakého důvodu nepovede a poplach bude spuštěn i při projití s legálně zakoupeným zbožím. Statisticky jsou 3 % návštěvníků zloději a ostatní chtějí zboží normálně koupit. Otázka nyní zní, jaká je pravděpodobnost, že alarm správně upozorní na kradené zboží. [7]

Výchozí hodnoty:

- P(A) – pravděpodobnost, že je zboží kradené
- P ($\bar{A}$) – pravděpodobnost, že zboží není kradené
- P(A|B) – pravděpodobnost, že je zboží kradené, a tudíž se spustí alarm
- P(B|A) – pravděpodobnost, že se spustí alarm, když je zboží kradené
- P(B|$\bar{A}$) – pravděpodobnost, že se spustí alarm, když zboží není kradené
- a … relativní množství zlodějů
- b … relativní množství poctivě nakupujících
- P(B) – pravděpodobnost, že se spustí alarm

P(B) se vypočítá jako součet šancí na spuštění alarmu při průchodu s kradeným i s nekradeným zbožím:

P(B) = a * P(B|A) + b * P(B|$\bar{A}$) = 0,03 * 0,95 + 0,97 * 0,05 = **0,077**

Nyní dosadíme hodnoty do Bayesovy věty:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)} = \frac{0{,}95 * 0{,}03}{0{,}077} = \frac{0{,}0285}{0{,}077} \approx \mathbf{0,3701}$$

Bylo zjištěno, že v případě spuštěného alarmu je v našem příkladu pouze 37 % šance, že jde skutečně o zloděje.

5 KLIENT-SERVER

Architektura klient – server (Client – server) je jedním z typů architektury informačních systémů. Jedná se o dvouvrstvou síťovou architekturu, která rozlišuje klientskou část, obsahující aplikační logiku a zobrazující určité uživatelské rozhraní, a část serverovou, která obsahuje nějaký druh relační databáze. Tyto dvě části spolu komunikují přes počítačovou síť. [8]

Jinými slovy, jedná se o dva počítačové programy, které spolu komunikují pro zajištění určitých cílů. Klientská část přijímá vstup od uživatele a poté žádá o informace nebo služby svoji druhou část – server. Toto uspořádání má mnoho praktických využití – E-mail, přístup k databázi, firemní a obchodní aplikace, internetové protokoly, atd. Nejpoužívanější aplikací je webový prohlížeč, což je klientská část nainstalovaná na počítači uživatele a o informace nutné ke správnému zobrazení stránky žádá server, na kterém daná stránka běží. (viz Obr. 3) [8, 9]

Komunikace mezi oběma částmi funguje tak, že klient přijme uživatelský vstup, což může být třeba stisknutí tlačítka nebo vyplnění pole, a následně odešle požadavek serverové části. Zde je požadavek zpracován a zpět klientovi je odeslána adekvátní odpověď. Klient pak jedná dle přijatých informací od serveru (zobrazí stránku, výsledky dotazu, …). Dotazy tedy zpracovává serverová část, klient pak přijímá vstupy od uživatele a prezentuje výsledky. (viz Obr. 4) [8]

5.1 Charakteristika klienta

- Aktivní část – reaguje na uživatelský vstup odesíláním požadavků
- Pouze rozhraní – Klient většinou neobsahuje požadované informace

5.2 Charakteristika serveru

- Pasivní část – server pouze reaguje na odeslané požadavky klientem
- Informace – server obsahuje všechny informace, které rozesílá klientům

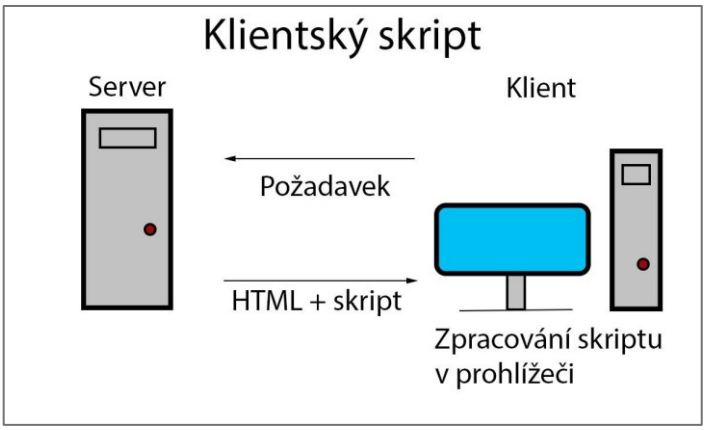
5.3 Výhody a nevýhody

Výše popsaná architektura klient-server má celou řadu výhod a některé nevýhody. Mezi výhody patří skutečnost, že lze určitou aplikaci rozdělit mezi více počítačů, což mimo jiné znamená, že lze vyměňovat nebo opravovat počítače, aniž by to konkrétní uživatelé poznali. Takto lze například provozovat rozsáhlá uložiště dat. [8, 9]

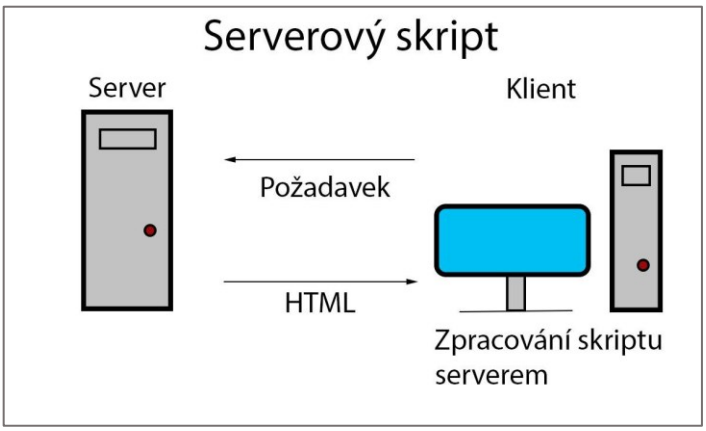
Jelikož všechny potenciálně citlivé informace obsahuje jen server, lze jej efektivněji zabezpečit proti neautorizovaným připojením a pokusům o krádeže dat. Manipulovat s daty mohou pouze oprávnění klienti. [9]

Další výhodou je skutečnost, že data jsou uložena centrálně, často v jedné databázi. Jakákoli změna provedená správcem se poté projeví u všech klientů. Není proto nutné aktualizovat každý klient zvlášť.

Nevýhodou můžou být nedostatečné kapacity počítačové sítě, na které tyto aplikace běží. Rychlost odpovědí serveru je tedy odvozena od propustnosti sítě a výkonu počítače, na kterém běží serverová databáze. Vzhledem k tomu, že všechny informace obsahuje server, doporučuje se databázi pravidelně zálohovat nebo provozovat více paralelních serverů. [8, 9]



Obr. 3. Princip funkce klientského skriptu



Obr. 4. Princip funkce serverového skriptu

6 DATABÁZE

Databáze je prostředí, kde se určitým způsobem ukládají údaje, které jsou pak přístupné pro jiné programy nebo další zpracování. Jako dotazovací jazyk se nejčastěji používá SQL.

Údaje ukládané v databázi mají schopnost o něčem vypovídat a do databáze jsou řazeny dle určitého klíče, třeba podle velikosti nebo chronologicky.

Objekty v databázi jsou nazývány „datové entity“. Každá entita je popsána svým názvem a je jí přiřazena sada atributů. Každý atribut má svůj vlastní datový typ. Pokud by databáze obsahovala entitu Obchody, mohly by jí být přiřazeny následující atributy – ID, Název, Popis, Adresa. [10]

S databází je potřeba rychle a efektivně pracovat. K tomu nám slouží databázový systém, což je softwarové vybavení pro práci s databází. Tento systém musí být schopen pracovat s velkým množstvím dat. K této práci nám slouží různé funkce – Create, Drop, Update, Delete, Select. K tomuto všemu používáme různé programy dostupné zdarma či za poplatek – Oracle, Microsoft SQL Server, MySQL, SQLite, atd. [10]

6.1 SQL Databáze

Databází se rozumí úložiště pro soubory dat se strukturou záznamů, které jsou propojeny pomocí klíčů. Databází tedy může být i jednoduchý soubor s odřádkovaným textem a parsovacím kódem, nicméně profesionální SQL server nabízí řadu výhod:

- Několikanásobný přístup – umožňuje paralelní připojení více uživatelů
- Druhy přístupu – podpora přístupů ze sítě, internetu nebo lokálního počítače
- Bezpečnost – databáze se dá zabezpečit lépe než lokální počítač uživatele
- Rychlost – databázový systém dokáže měnit a modifikovat velké objemy dat, jsou však omezeny výkonem počítače
- SQL jazyk – jazyk pro zadávání příkazů

Databáze obsahuje doprovodné funkce a programy. Tento balíček je označovaný jako systém řízení báze dat (DBMS), který řeší komunikaci mezi aplikačním softwarem a daty fyzicky uloženými v databázi. Tyto systémy nejčastěji nabízejí následující funkce:

- Uchovávání dat i po nečekaném výpadku
- Udržování definované struktury záznamů
- Zabezpečení

- Jazyk pro ovládání databáze
- Funkce – zajišťují práci s databází a reagují na změnu stavu databáze
- Vazby mezi tabulkami
- Indexy – zajištění unikátní hodnoty řádků

SQL server tedy nabízí kompletní řešení ukládání dat v jediném balíčku. Jedná se o nejrozšířenější způsob ukládání dat, který se používá u většiny dnes vytvářených aplikací. Jedná se o profesionální řešení, placené a vynikající pro velmi náročné úkoly. [11, 12]

6.2 MySQL Databáze

MySQL je multiplatformní databáze. Komunikace s ní probíhá pomocí jazyka SQL. Pro svou snadnou implementovatelnost, výkon, a především volně šiřitelný software je v současné době mezi nejpoužívanějšími. Velmi oblíbená a často nasazovaná je kombinace Linux, Apache, MySQL a PHP jako základní software webového serveru. Jedná se o Open-Source, který je používán především na levné věci a pro web hosting. Výkon je horší než u SQL serveru zejména u složitějších dotazů. [10]

6.3 Práce s databázemi

Práce s databázemi je založena na tabulkovém uspořádání dat. Všechny data jsou tedy uložena v konkrétní tabulce a vztahy jsou dány relačními vazbami, Správce databáze obvykle pracuje s nějakou množinou dat pomocí jazyka SQL.

Vazby mezi tabulkami mohou mít různé formy. Nejčastěji se můžeme setkat s těmito:

- 0 – Tabulka může, ale nemusí obsahovat vázaný záznam k tabulce druhé
- 1 – Tabulka musí obsahovat právě jeden vázaný záznam k tabulce druhé
- N – První tabulka obsahuje více vázaných záznamů
- M – Druhá tabulka obsahuje více vázaných záznamů

V praxi se tedy může vyskytovat různé množství vztahů mezi tabulkami, například 1/N, kdy v první tabulce existuje právě jeden vázaný záznam, přičemž v druhé tabulce je odpovídajících záznamů více než jeden. [12]

6.4 Datové typy

Základem každé databáze je tabulka, která obsahuje jednotlivé položky s informacemi. Každý sloupec má přitom přiřazený datový typ. Tento typ určuje druh dat, které lze do daného sloupce uložit a také druh operací, které lze s těmito daty provádět.

Data se do databáze dají uložit jako různé datové typy – Integer, Float, Varchar, Text, Date, Time, Boolean.

- Integer – celá čísla
- Float – desetinná čísla
- Varchar – text, omezen zadanou délkou
- Text – text
- Date – datum
- Time – čas
- Boolean – logické hodnoty – Pravda / Nepravda

6.5 Typy databází podle struktury dat

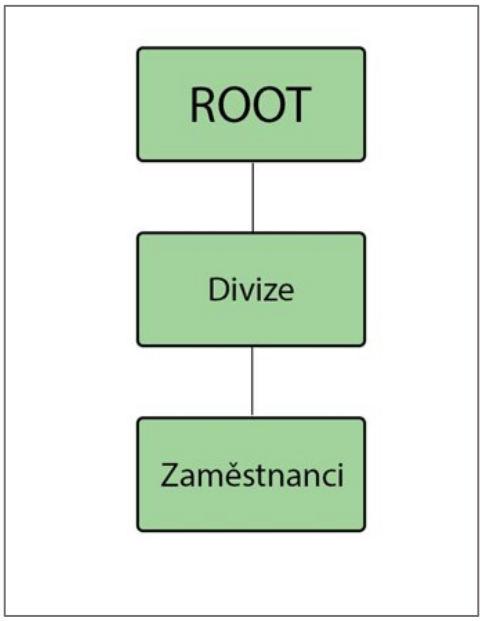
Existují různé typy databází, lišící se zejména modelem uložení dat a datem vzniku, od nejstarších a nejjednodušších plochých databází až po nejmodernější relační databáze. [11]

6.5.1 Plochá (flat) databáze

Tato jednoduchá databáze uchovává data jako obyčejnou sadu záznamů. Příkladem může být soubor typu CSV, který na jednotlivých řádcích uchovává záznamy. Sloupce se poté rozlišují dělícím znakem, což může být třeba středník, tabulátor, atp. Tento typ databáze neobsahuje vazby na jiné záznamy nebo tabulky, s čímž souvisí využitelnost takové databáze – vhodná pro základní uložení dat. [11]

6.5.2 Hierarchická a síťová databáze

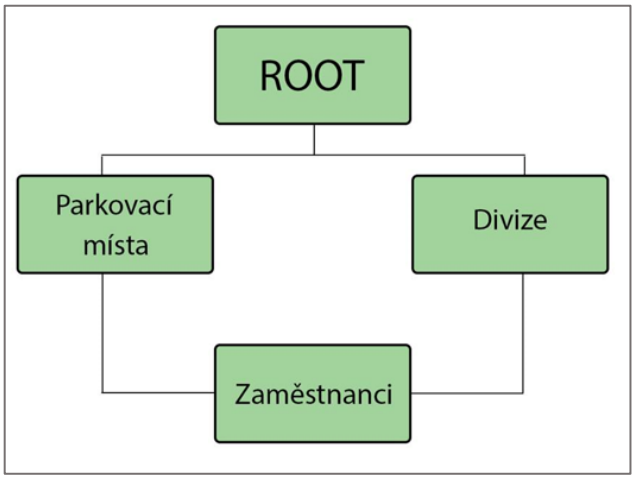
Jedná se o starší typy databází, které se dnes již téměř nepoužívají. Tyto typy měly snahu eliminovat opakující se data v plochých databázích a nepraktickou práci s nimi. Proto tyto databáze dovolují definovat k jednotlivým tabulkám takzvané rodičovské tabulky. Poté tedy platí, že každý záznam v tabulce musí obsahovat odkaz na jeden ze záznamů z hierarchicky vyšší tabulky. [11, 12]



Obr. 5. Ukázka hierarchické databáze

Podle obrázku 5 je tedy zřejmé, že existuje samostatný seznam zaměstnanců a seznam pracovních divizí, přičemž tabulka zaměstnanců je umístěna hierarchicky pod tabulkou pracovních divizí, takže každý zaměstnanec obsahuje odkaz na jednu pracovní divizi. Takováto definice zabírá minimum prostoru a zároveň je třeba pracovní divize definovat pouze jednou a na jediném místě. [11, 12]

Síťové databáze (Obr. 6) fungují velmi podobně, ale mají navíc možnost používat více rodičovských tabulek. Pokud by tedy každý zaměstnanec měl možnost používat parkovací místo, tak by položky z tabulky zaměstnanců měly vazby na tabulku pracovních divizí a na tabulku parkovacích stání. [11, 12]



Obr. 6. Ukázka síťové databáze

6.5.3 Relační databáze

Tyto databáze se dnes nejvíce používají a jako kompletní databázové systémy zcela nahradily předchozí typy. Starší typy databází se však stále používají, zejména v aplikacích, kde není třeba robustního systému s pokročilými funkcemi.

Tato databáze obsahuje data ve formě řádků a sloupců. Řádek zde tedy odpovídá záznamu a sloupce atributům. Každému sloupci je určen datový typ (varchar/text, date, integer, …). Každý atribut tedy uchovává jeden druh informace a jakákoli požadovaná data lze rychle vyhledat. [13]

Relační databáze jsou jednoduché, jsou vhodné pro řízení velkého množství dat a vyhledávání v nich, ale často poskytují malou podporu pro manipulaci. Jsou založeny na jednoduchých tabulkách a vztahy mezi informacemi jsou vyjadřovány porovnáváním uložených hodnot. K jakékoli modifikaci je třeba použít určený jazyk, nejčastěji SQL. [13]

6.5.4 Objektová databáze

Tyto databáze využívají datového modelu, který má objektově orientované aspekty a podporují abstraktní datové typy. Tyto databáze nejsou příliš rozšířeny, protože jsou pomalé. Jinými slovy, tyto databáze jsou založeny na objektech. Objekty jsou struktury kombinující daný kód a data. Objektové databáze tak kombinují prvky objektově orientovaného programování s databázovými schopnostmi. [13]

Objektové databáze umožňují využít objektově orientovaný jazyk aplikace přímo v databázi a není tak potřeba druhého dotazovacího jazyka. Tyto databáze jsou tedy vhodné pro manipulaci s daty.

6.5.5 Souborová databáze

Taková databáze může být obsažena v jednom, ale i v několika souborech operačního systému počítače. Takovou databázi lze snadno dostat na jiný počítat tak, že se překopíruje celý soubor. Tyto databáze však mají několik omezení. Hlavním omezením je dostupnost takové databáze v rámci sítě nebo podpora pro souběžnou práci více uživatelů. Obecně platí, že k práci s těmito databázemi je potřeba mít nějaká práva, jinými slovy k přístupu k souboru je potřeba mít povolení. [10]

6.5.6 Systémová databáze

Tyto databáze slouží jako databázové servery. Výhodou je, že mají dobrou podporu souběžné práce více uživatelů. Na rozdíl od souborových databází jsou robustnější a mají také složitější instalaci. Tyto databáze bývají přístupné pomocí nějakého protokolu. [10]

7 ARCHITEKTURA MODEL-VIEW-CONTROLLER (MVC)

Jde o softwarovou architekturu pro tvorbu uživatelských rozhraní, která se rozděluje na tři nezávislé komponenty, jak lze vidět na Obr. 7. Prvním komponentem je datový model, který se reprezentuje informacemi, které aplikace potřebuje ke své práci. Druhým komponentem je uživatelské rozhraní neboli view (pohled), které přivádí data reprezentována modelem do podoby vhodné k prezentaci uživateli. Třetím a posledním komponentem je řídící logika neboli controller (řadič), který je schopen reagovat na jednotlivé události pocházející od uživatele a je schopen zajišťovat změny v modelu i pohledu. [14]

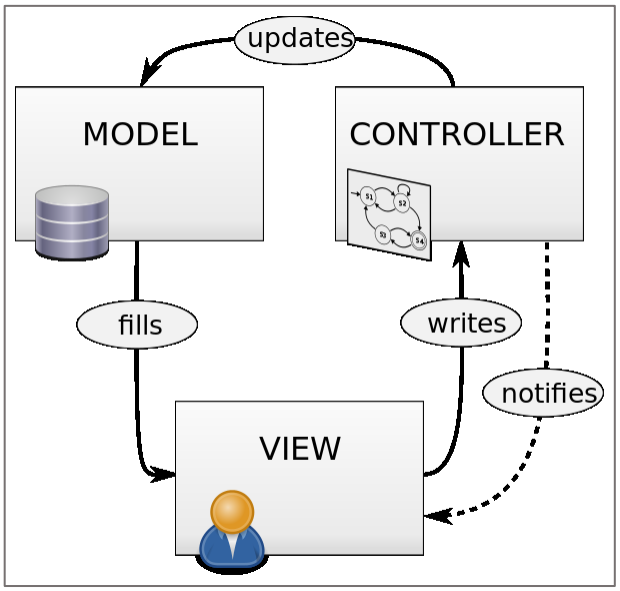
Tato architektura se tradičně používá pro grafické uživatelské rozhraní (GUI) a stala se populární pro navrhování webových aplikací.

7.1.1 Výhody

- Souběžný vývoj – více vývojářů může pracovat současně na modelu, řadiči i pohledech
- Vysoká soudržnost – MVC umožňuje logické seskupení souvisejících činností na řadiči dohromady. To umožňuje vysokou znovu použitelnost jednotlivých komponent, robustnost, spolehlivost a stabilitu.
- Nízká vazba – Díky separaci privilegií do tří komponent je jednoduché modifikovat jednotlivé součásti MVC modelu

7.1.2 Nevýhody

- Hledání kódu – rámcová navigace může být složitá, protože zavádí nové vrstvy abstrakce a vyžaduje, aby se uživatelé přizpůsobili kritériím rozkladu MVC
- Výrazná křivka učení – znalost o více technologiích se stává normou. Vývojáři, kteří používají MVC, musí mít zkušenosti s několika technologiemi.



Obr. 7. Architektura MVC [15]

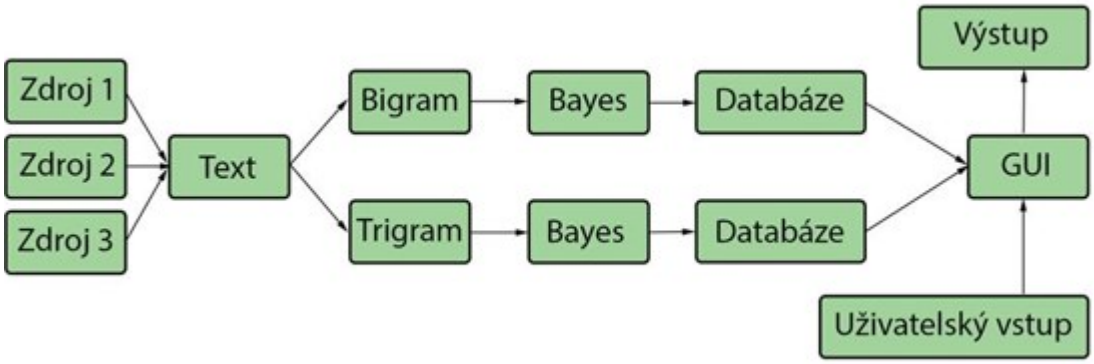
II. PRAKTICKÁ ČÁST

8 NÁVRH

Smyslem navrhovaného řešení prezentovaného v této práci je propojení predikce textu s předpřipravenou databází. Program tedy bude na základě uživatelského vstupu (slovo) nabízet seznam možných následujících slov seřazených podle Bayesovy pravděpodobnosti.

Databáze bude vypracována sběrem dat (textu) z vybraných internetových zdrojů. Tato data musí být očištěna od speciálních znaků a nevětných útvarů. Na takto očištěná data by měl být použit N-Gram, který bude zodpovědný za vytvoření vazeb mezi slovy. Z těchto N-Gramů se vypočítá Bayesova pravděpodobnost, která určí pravděpodobnost následnosti slov v analyzovaných textech.

Pro potřeby projektu bude také vytvořeno uživatelské rozhraní, které bude ulehčovat a zrychlovat práci s programem.



Obr. 8. Schéma návrhu

Z výše popsaného návrhu z Obr. 8 je zřejmé, že k řešení bude potřeba více programovacích jazyků. Pro sběr dat z internetových zdrojů, následné očištění a ukládání do databáze bude použit programovací jazyk Java. Jako databázový server se využívá MySQL-5.7.11. Pro přístup do databáze bude použita aplikace PhPMyAdmin. Ke komunikaci mezi uživatelským rozhraním a databází bude sloužit PhPFramework CodeIgniter. Uživatelské rozhraní bude vypracováno pomocí HTML stránky s využitím JavaScriptu.

8.1 Programovací jazyk JAVA

Počátky tohoto programovacího jazyka spadají do první poloviny devadesátých let minulého století. Pojem „Java“ má dva významy. První význam – Java jako programovací jazyk. Používá se pro zápis programů a jeho nedílnou součástí, jako programovacího jazyka, je tzv. aplikační programové rozhraní. Zjednodušeně řečeno jde o kód, který máme k dispozici při

používání tohoto jazyka. Druhý význam – Java jako platforma, která slouží pro spouštění programů. [16]

Java patří mezi interpretované jazyky. Místo strojového kódu tedy vytváří mezikód nezávislý na architektuře počítače, proto pak program funguje na libovolném zařízení. Java se tedy používá pro multiplatformní a přenositelné aplikace. Java využívá automatický garbage collector, který se stará o průběžné uvolňování paměti. Programátor musí pouze rušit odkazy na objekty.

Tento jazyk byl využit pro rychlé spouštění a stabilní zpracování dat z webových odkazů pro vytvoření databáze. Jedna z velkých výhod je snadný přístup do MySQL a kompatibilita se serverovou částí.

8.2 MySQL

Jde o relační databázový systém, který vlastní společnost Oracle, ale je šířen jako Open Source, což znamená volně dostupný. Každá databáze v MySQL je tvořena nejméně jednou, ale spíše více tabulkami. Tyto tabulky mají řádky a sloupce. V každém řádku se nachází právě jeden záznam. Sloupce mají své jméno a zároveň svůj datový typ. Práce s databází se provádí pomocí dotazů, které vycházejí z programovacího jazyka SQL. [10]

8.3 PHP

Původní význam zkratky PHP byl Personal Home Page (Osobní domovská stránka). Postupně docházelo ke změnám a nyní tato zkratka znamená Hypertext Preprocessor. Jedná se o skriptovací programovací jazyk, který pracuje na straně serveru a slouží pro programování dynamických internetových stránek. Pomocí tohoto programovacího jazyka se dokážou měnit data na jednotlivých webových stránkách. Pochopit tento jazyk pro uživatele není nijak náročné, protože lze vystačit jen se základy, což je ukládání, mazání a úprava jednotlivých dat. [10, 17]

8.4 HTML

Jedná se o webové stránky nebo jinak také HTML dokumenty. Tyto dokumenty jsou obyčejné textové soubory. Takový soubor obsahuje samotný text, ale i speciální značky, které určují význam jednotlivých částí textu. [18, 19]

Byl využit pro tvorbu uživatelského a administrativního rozhraní.

8.5 JavaScript

Programovací jazyk pro WWW stránky, který je často přímo vkládaný do HTML kódu. Na rozdíl od programovacího jazyka PHP se tento jazyk spouští až po stažení WWW stránky z internetu. Z tohoto důvodu nemůže tento jazyk pracovat se soubory, aby tím neohrozil soukromí uživatele. [18]

JavaScript byl použit pro našeptávání slov v uživatelském rozhraní a spouštění scriptů vázaných na určité klávesy.

9 IMPLEMENTACE

Klient server je aplikace, kde serverová část zajišťuje sběr dat z internetových zdrojů. Aplikace je napsána v programovacím jazyce Java a je nezávislá na použité platformě. Serverová část této aplikace využívá databázový server MySQL. Z tohoto databázového serveru si program zajišťuje svá nastavení a využívá ho také pro ukládání „naparsovaných" dat.

Aplikace zajišťuje sběr dat z internetových zdrojů a může ji spustit každý uživatel sám. Serverová část aplikace pro sběr dat z internetových zdrojů si stáhne své nastavení z databázového serveru. Na tomto databázovém serveru se nachází uložené internetové zdroje, které bude postupně procházet. Dále je v této aplikaci nastaveno vnoření na hodnotu 0, aby nedocházelo k přílišnému zanoření, aby byly využitelné všechna nalezená data. Prochází postupně celý HTML dokument stránky, kde získáváme všechen text z jednotlivých HTML elementů. Texty jsou serverovou částí následně zpracovány. Nalezené texty, které jsou delší než 4 slova, jsou označené jako věty (viz Obr. 9).

```
if(ddata.split("[\\s\\xA0]+").length > 4)
```

Obr. 9. Podmínka pro oddělení vět od šumu

Věty jsou dále zpracovávány tak, že jsou rozděleny na dvojice a zároveň na trojice. Dvojice v textu označujeme jako „bigram" a trojice „trigram" (viz Obr. 10 a 11).

```
slova.add(new Slova(b.get(i), b.get(i+1)))
```

Obr. 10. Vytvoření bigramu

```
slova.add(new SlovaTrigram(b.get(i), b.get(i+1), b.get(i+2)))
```

Obr. 11. Vytvoření trigramu

Jednotlivá slova jak u bigramu, tak trigramu musí být očištěna od speciálních znaků. Speciálními znaky rozumíme například dvojtečka, vykřičník, otazník a tak dále (viz Obr. 12).

```
String upraveno = ddata.replaceAll("([:%-?!.,…])", "").toLowerCase().trim()
```

Obr. 12. Odebrání speciálních znaků

Po očištění od speciálních znaků se tyto bigramy nebo trigramy uloží do databáze do konkrétní tabulky. Adresy jednotlivých navštívených webových stránek jsou uloženy do lokálního souboru (nebo databáze), aby nedocházelo k duplikacím dat na navštívených stránkách.

Příklad:

„Zatím je tabulka neuvěřitelně vyrovnaná a pomalu začíná jít do tuhého.“

Podle části programu (Obr. 9) se tento text rozdělí podle mezer na jednotlivá slova a zároveň se vypočítá počet slov. Pokud má text více než 4 slova, tak je považován za větu. Dostaneme [„Zatím“,“je“,“tabulka“,“neuvěřitelně“,“vyrovnaná“,“a“,“pomalu“,“začíná“,“jít“,“do“,“tuhého“], délka odpovídá 11 > 4. V dalším kroku (Obr. 12) dojde k očištění od speciálních znaků. Následně dojde (Obr. 10) k vytvoření bigramu: [„Zatím“,“je“], [“je“,“tabulka“], [„tabulka“,“neuvěřitelně“] a tak dále, poté následné vložení do databáze.

Při tvoření trigramu postupujeme ze začátku jako u bigramu až po očištění od speciálních znaků. Následně dojde (Obr. 11) k vytvoření trigramu: [„Zatím“,“je“,“tabulka“], [“je“,“tabulka“,“neuvěřitelně“], [“tabulka“,“neuvěřitelně“,“vyrovnaná“] a tak dále, poté následné vložení do databáze (viz Obr. 13).

(A)

Zatím	je
je	tabulka
tabulka	neuvěřitelně
neuvěřitelně	vyrovnaná
vyrovnaná	a
a	pomalu
pomalu	začíná
začíná	jít
jít	do
do	tuhého

(B)

Zatím	je	tabulka
je	tabulka	neuvěřitelně
tabulka	neuvěřitelně	vyrovnaná
neuvěřitelně	vyrovnaná	a
vyrovnaná	a	pomalu
a	pomalu	začíná
pomalu	začíná	jít
začíná	jít	do
jít	do	tuhého

Obr. 13. Bigram (A) a Trigram (B) v databázi.

9.1 Implementace databázového systému

Databázový systém se skládá celkem z pěti tabulek. Každá tabulka v databázovém systému plní svoji funkci. Pro funkcionalitu tohoto systému by stačilo použít 3 tabulky, ale nebylo by rozlišeno, kdy se jedná o bigram a kdy o trigram. Proto jsou tabulky navrženy pro každý

účel zvlášť. Tyto tabulky jsou pojmenovány následovně – adresy, cache, cache2, ddata, ddata2.

- adresy – Obsahuje internetové odkazy, které budou prohledány.
- cache – Obsahuje už předpřipravené bigramy pro uživatelské rozhraní aplikace.
- cache2 - Obsahuje už předpřipravené trigramy pro uživatelské rozhraní aplikace.
- ddata – Obsahuje bigramy, které budou dále serverovou částí zpracovávány.
- ddata2 - Obsahuje trigramy, které budou dále serverovou částí zpracovávány.

9.1.1 Tabulka adresy (adresy)

Tato tabulka obsahuje 2 sloupce, jak je vidět na Obr. 14. První sloupec (id) obsahuje datový typ integer. Druhý sloupec obsahuje text (viz Obr. 15), který nám udává internetové zdroje, které bude procházet pro sběr dat.

id	adresa
1	https://www.seznam.cz
2	https://www.novinky.cz
3	http://www.zing.cz
4	http://www.pravo.cz
5	https://www.sport.cz

Obr. 14. Tabulka adresy

#	Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Další
1	**id**	int(11)			Ne	*Žádná*	AUTO_INCREMENT
2	**adresa**	text	utf8_general_ci		Ano	*NULL*	

Obr. 15. Struktura adresy

9.1.2 Tabulka bigramů (ddata)

Tato tabulka obsahuje celkem 3 sloupce, jak je vidět na Obr. 16. První sloupec (id) obsahuje položku typu integer. Druhý sloupec (klic) obsahuje první slovo bigramu. Jeho datový typ je tedy text. Třetí a poslední sloupec (hodnota) obsahuje druhé slovo bigramu. Strukturu celé tabulky lze vidět na Obr. 17. Tato tabulka nemá žádné relace s ostatními tabulkami a je využívána k plnění očištěnými daty z internetových zdrojů.

id	klic	hodnota
1	seznam	najdu
2	najdu	tam
3	tam	co
4	co	neznám
5	nejslavnější	mrakodrap

Obr. 16. Tabulka ddata

#	Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Další
1	**id**	int(11)			Ne	*Žádná*	AUTO_INCREMENT
2	**klic**	text	utf8_general_ci		Ne	*Žádná*	
3	**hodnota**	text	utf8_general_ci		Ne	*Žádná*	

Obr. 17. Struktura ddata

9.1.3 Tabulka trigramů (ddata2)

Tato tabulka je strukturně velice podobná s tabulkou „ddata“ (viz Obr. 18). Rozdíl je v počtu sloupců pro uložení trigramů, který obsahuje tři slova namísto dvou jako bigram. Struktura této tabulky je vidět na Obr. 19. Tato tabulka nemá opět žádné relace s ostatními tabulkami. Serverovou částí je tato tabulka využívaná k plnění očištěnými daty z internetových zdrojů.

id	klic	hodnota1	hodnota2
1	seznam	najdu	tam
2	najdu	tam	co
3	tam	co	neznám
4	nejslavnější	mrakodrap	světa
5	mrakodrap	světa	je

Obr. 18. Tabulka ddata2

#	Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí	Další
1	**id**	int(11)			Ne	*Žádná*	AUTO_INCREMENT
2	**klic**	text	utf8_general_ci		Ne	*Žádná*	
3	**hodnota1**	text	utf8_general_ci		Ne	*Žádná*	
4	**hodnota2**	text	utf8_general_ci		Ne	*Žádná*	

Obr. 19. Struktura ddata2

9.1.4 Tabulka cache (cache)

Tato tabulka obsahuje 2 sloupce (Obr. 20). První sloupec (klic) obsahuje textovou položku, která označuje první slovo bigramu. Druhý sloupec (hodnoty) obsahuje datový typ longtext (jak lze vidět na Obr. 21), který využívá objektový zápis JSON.

Položka v této databázi se tedy skládá ze tří samostatných vstupů, které jsou dále využívány programem. První část je určité slovo, například „podat“. K tomuto slovu jsou přiřazeny dvě číselné hodnoty. První hodnota říká, kolikrát se hodnota „na“ objevila ve spojení s klíčem „podat“ a dále vynásobená pravděpodobností výskytu hodnoty „na“ ve všech hodnotách. Druhá hodnota je součet všech pravděpodobností (normalizační konstanta). Obě tyto čísla jsou vstupními hodnotami pro výpočet Bayesovy pravděpodobnosti, která bude podrobně rozebrána níže.

klic	hodnoty
novinkycz	[["–",5.877034358047e-5,0.0095818264014467],["je",...
–	[["nejčtenější",4.8685491723466e-6,0.0060662122687...
nejčtenější	[["zprávy",0.00025316455696203,0.00025316455696203...
zprávy	[["na",0.011139240506329,0.012974683544304],["podl...
na	[["českém",7.1113639596075e-7,0.0014201393827336],...
českém	[["internetu",0.0010126582278481,0.001012658227848...
sobotka	[["jsem",0.0020203524447754,0.0056068503350707],["...
jsem	[["připraven",0.0018097844680123,0.002853232979815...
připraven	[["podat",0.0027848101265823,0.0027903137039075],[...
podat	[["na",0.021309851403412,0.021315354980737],["nemo...
prezidenta	[["kompetenční",0.0042932489451477,0.0043196202531...
kompetenční	[["žalobu",0.0040711597673623,0.0040882654806705],...

Obr. 20. Tabulka cache

#	Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí
1	**klic**	varchar(50)	utf8_general_ci		Ne	*Žádná*
2	**hodnoty**	longtext	utf8_general_ci		Ne	*Žádná*

Obr. 21. Struktura cache

9.1.5 Tabulka cache2 (cache2)

Tato tabulka obsahuje 2 sloupce (viz Obr. 22). První i druhý sloupec obsahuje datový typ longtext, který využívá objektový zápis JSON (Obr. 23). První sloupec na rozdíl od cache obsahuje 2 slova. Ve sloupci hodnota je třetí slovo trigramu a dvěma číselnými hodnotami. Význam těchto čísel je stejný jako v kapitole 9.1.4.

klic	hodnoty
["novinkycz","–"]	[["nejčtenější",0.00013498920086393,0.000134989200...
["–","nejčtenější"]	[["zprávy",0.00013498920086393,0.00013498920086393...
["nejčtenější","zprávy"]	[["na",0.023083153347732,0.023083153347732]]
["zprávy","na"]	[["českém",0.00013498920086393,0.00013498920086393...
["na","českém"]	[["internetu",0.0010799136069114,0.001079913606911...

Obr. 22. Tabulka cache2

#	Název	Typ	Porovnávání	Vlastnosti	Nulový	Výchozí
1	**klic**	varchar(50)	utf8_general_ci		Ne	*Žádná*
2	**hodnoty**	longtext	utf8_general_ci		Ne	*Žádná*

Obr. 23. Struktura cache2

9.2 Bayesova statistika

Program uživateli nabízí seznam slov řazený dle Bayesovy statistiky. V databázi jsou zpracovány počty výskytu, jak je popsáno v části 9.1.4. Databáze tedy poskytuje tato dvě čísla, ze kterých program počítá pravděpodobnost dle následujících konkrétních vzorců, které jsou založeny na rovnici (2) v části 4.2:

Pro bigramy platí (3), že:

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}, \tag{3}$$

kde *jev A* je predikované slovo a *jev B* je dané slovo (klic).

- P (A|B) – Zjišťuje pravděpodobnost, že nastal *jev A* za předpokladu že nastal *jev B*. V anglické literatuře označován jako posterior.
- P (A) – Pravděpodobnost výskytu *jevu A*. V anglické literatuře označováno jako prior.
- P (B|A) – Zjišťuje pravděpodobnost, že nastal *jev B* za předpokladu, že nastal *jev A*
- P (B) – Pravděpodobnost výskytu *jevu B*, někdy také označován jako normalizační konstanta

Vybrání jednoho klíče a hodnot ke klíči z databáze dat a následné vypočítání Bayesova teorému:

KLÍČ	HODNOTA	P(B|A) [-]	P(A) [-]	P(B) [-]	P(A|B) [%]
novinkycz	-	0,0027	0,0016	0,01122	**0,04**
	je	0,3977	0,0125	0,01122	**44,47**
	jsou	0,1988	0,0031	0,01122	**5,51**
	bez	0,1988	0,0013	0,01122	**2,47**
	a	0,1988	0,0268	0,01122	**47,51**
	měla	0,0027	0,0003	0,01122	**0,01**

Tab. 2. Vzorová ukázka na slovo „novinkycz“

$$P(a|novinkycz) = \frac{P(novinkycz|a) * P(a)}{P(novinkycz)} = \frac{0{,}1988 * 0{,}0268}{0{,}01122} = \mathbf{0,4751}$$

Pro trigramy platí (4), že:

$$P(A|B1, B2) = \frac{P(B1, B2|A) * P(A)}{P(B1, B2)}, \tag{4}$$

kde *jev A* je predikované slovo a *jev B1* a *B2* jsou dané slova (klice). Musí nastat obě v daném pořadí.

```
public function countWords($slovo, $pole)
{
    $pocet = 0;
    $poleTemp = [];
    $r = $this->radky($slovo,$pole);
    $f = $this->removeDuplicates($r);
    foreach ($f as $value)
    {
           foreach ($r as $value1)
            {
              if($value == $value1)
                {
                    $pocet = $pocet + 1;
                }
          }

          array_push($poleTemp, [$value,($pocet/$this->totalCount($slovo,$pole))*($this->PA($slovo,$pole)),0]);
          $pocet = 0;
      }

    return $this->normalize($poleTemp);
}
```

Obr. 24. Funkce countWords

Nejprve se zjistí všechna slova, která s daným slovem (klíčem) souvisí. Tato slova se vyfiltrují pomocí funkce „removeDuplicates“ tak, aby byla získána množina souvisejících slov. Pro každé slovo se vypočte počet jeho výskytů za předpokladu, že je znám klíč (viz Obr. 24).

```
public function totalCount($slovo, $pole)
    {
        $count = 0;
        foreach ($pole as $value)
        {
            if($value['klic'] == $slovo)
            {
                $count = $count + 1;
            }
        }
        return $count;
    }
```

Obr. 25. Funkce totalCount

Funkce „totalCount“ (viz Obr. 25) spočítá počet výskytů daného klíče. Následně se pomocí funkce „countWords“ (viz Obr. 24) dělí počet výskytů počtem získaných z funkce „totalCount“. Dohromady dají P (B|A).

```
private function PA($a, $pole)
{
    $count = 0;
    foreach ($pole as $value)
    {
        if($value['hodnota'] == $a)
        {
            $count = $count + 1;
        }
    }
    return $count/count($pole);
}
```

Obr. 26. Funkce PA

Funkce „PA“ (viz Obr. 26) slouží ke zjištění pravděpodobnosti predikovaného slova z celkové množiny dat.

```php
private function normalize($pole)
{
    $sum = 0;
    $tempPole = [];

foreach($pole as $value){

    $sum = $sum + $value[1];
}

foreach($pole as $value){
    array_push($tempPole,[$value[0],$value[1],$sum]);
}
```

Obr. 27. Funkce normalize

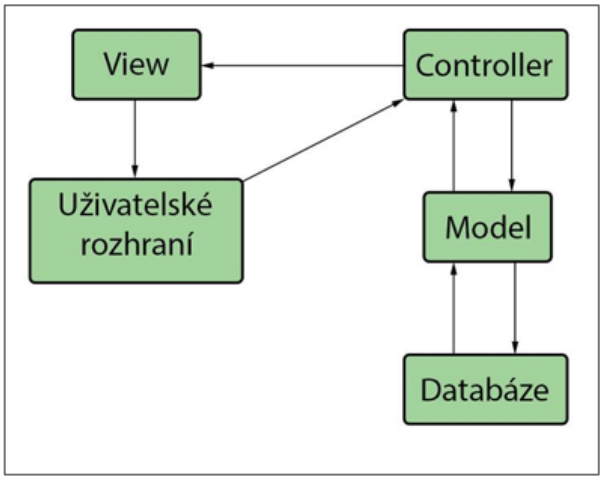
Funkce „normalize“ (viz Obr. 27) poté vypočítá normalizační konstantu – neboli P(B). Tato funkce zajistí, že pravděpodobnost daného klíče všech slov se sečte do 1 a tím se vytvoří tzv. posteriorní distribuce neboli P (A|B).

9.2.1 Aktualizace dat

Aktualizace dat probíhá tak, že získaná posteriorní pravděpodobnost (P(A|B)) je použita jako nová hodnota P(A). Tato hodnota je použita pro následující výpočty. Jde o vlastnost, která vede k získávání nových informací ze starších. Tato skutečnost vede k učení se aplikace. Bigramy nebo trigramy, které se vyskytují v textu častěji než jiné, budou Bayesovým teorémem preferovanějšími.

9.3 Codeigniter

Jde o open source a PhPFrameworkový program, který spadá pod architekturu MVC (Model-view-controller). Na Obr. 28 lze vidět vazby mezi jednotlivými částmi.



Obr. 28. Diagram MVC

Model zajišťuje komunikaci s databází. View obsahuje prezentaci obsahu pro uživatele. Controller řídí model, view a celou aplikaci.

9.3.1 Model

Model z architektury MVC, který byl použit v této práci, se skládá ze čtyř PHP souborů. Jednotlivé soubory jsou pojmenovány Cache.php, Cache2.php, DData.php, DData2.php.

Cache.php

Používá se pro komunikaci s databázovou vrstvou. Obsahuje model pro spojení se s tabulkou cache v databázi predikce, bez kterého nefunguje spojení databáze s PHP skriptem. PHP skript obsahuje funkce, jako jsou „insert_Data“, „update_Data“, „get_Data“ a pracuje s bigramy.

- insert_Data – Funkce přebírá klíč a hodnoty, které vkládá do tabulky v databázi
- update_Data – Funkce přebírá klíč a hodnoty. Podle klíče aktualizuje patřičné hodnoty v tabulce
- get_Data – Funkce, která získává veškerá data z tabulky

Cache2.php

Funguje na podobném principu jako soubor Cache.php, jenom obsahuje funkce pro práci s trigramy.

DData.php

Tento soubor byl použit pro získání veškerých bigramů, které byly vloženy do databáze serverovou částí aplikace.

DData2.php

Funguje na podobném principu jako soubor DData.php, jenom se jedná o funkci, která pracuje s trigramy.

9.3.2 Controller

Controller řídí model, view a celou aplikaci, která je prezentována uživateli. Obsahuje několik funkcí, které mohou být spuštěny. Mezi tyto funkce patří „index“ a „recalculate“. Funkce „index“ prezentuje hlavní stránku aplikace a předává data z databáze pro JavaSkriptovou část stránky. Tato data jsou pak použita pro predikci textu. Funkce „recalculate“ slouží

k aktualizaci a vložení nových dat do databáze, ale aby tato funkce mohla být spuštěna, je nutné vybrat, která tabulka v databázi se bude přepočítávat a je nutné také zadat heslo administrátora, aby se vyhnulo neoprávněnému spuštění. Získává data z tabulek databáze – jak z ddata, tak i cache. Spustí funkci insertOrUpdate, která pracuje s bigramy. Tato funkce rozhodne o tom, která data budou potřeba aktualizovat a která naopak nově vytvořit. Tento postup platí i pro trigramy, jen využívá tabulek – ddata2 a cache2 – a funkci insertOrUpdateTrigram.

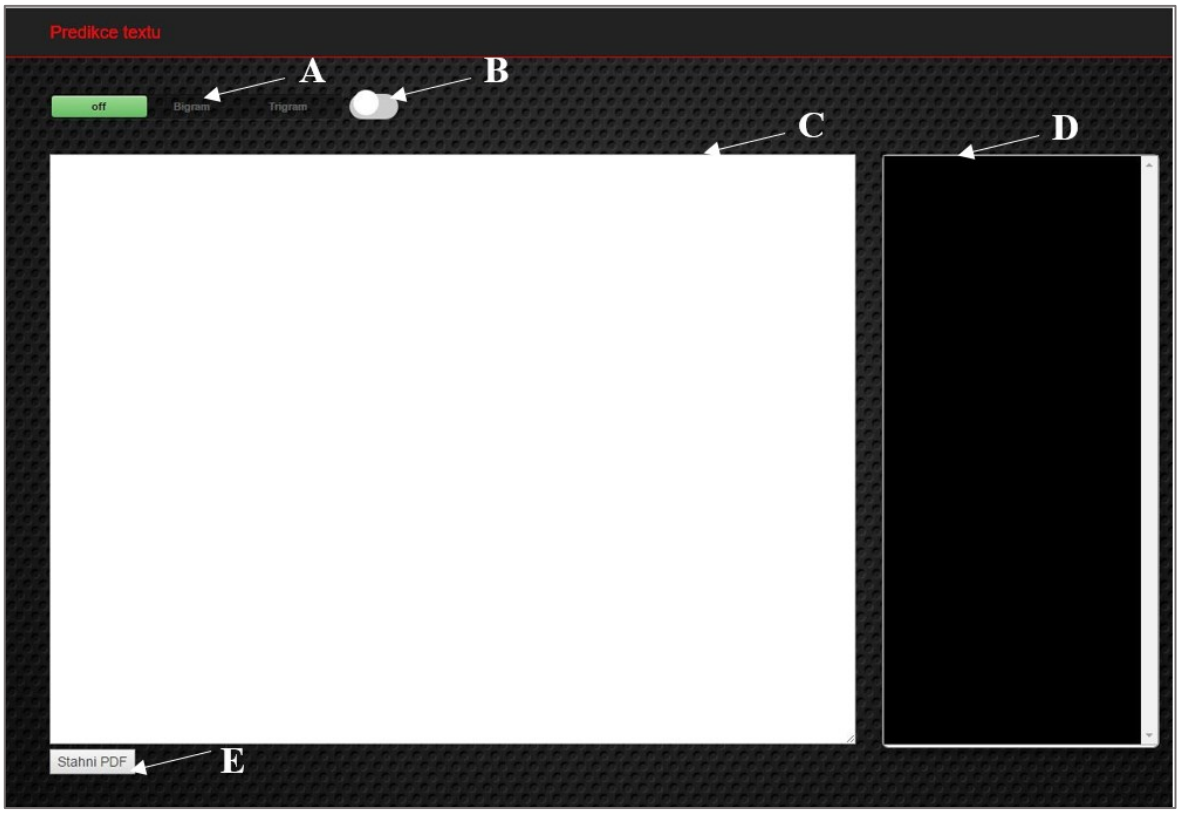
9.4 Webové rozhraní

Webové rozhraní se skládá ze dvou HTML stránek - hlavní stránky (viz Obr. 29) a stránky Recalculate. Hlavní stránka slouží pro běžné uživatele, zatímco stránka Recalculate je pro správce programu.

9.4.1 Hlavní stránka

Tato stránka obsahuje následující prvky:

- Tlačítka pro výběr druhu predikce (A)
- Posuvné tlačítko pro změnu grafického rozhraní (B)
- Textové pole pro zadávání textů (C)
- Tabulka predikovaných slov (D)
- Tlačítko pro export textu do pdf dokumentu (E)



Obr. 29. Celkový pohled na uživatelské rozhraní

Tlačítka pro výběr typu predikce (prvek A)

Posuvník slouží pro vybrání typu predikce textu (viz Obr. 30) a obsahuje tyto prvky:

- Off – vypnutá predikce, žádné slovo se nenabízí na slovo predikované
- Bigram – zapnutá predikce, predikuje nejpravděpodobnější slova na základě předem zadaného slova
- Trigram – zapnutá predikce, predikuje nejpravděpodobnější nadcházející slovo po předem zadané dvojici slov

Tyto volby je možné měnit i v průběhu psaní.

Obr. 30. Posuvník pro výběr typu predikce

Posuvné tlačítko pro změnu grafického rozhraní (prvek B)

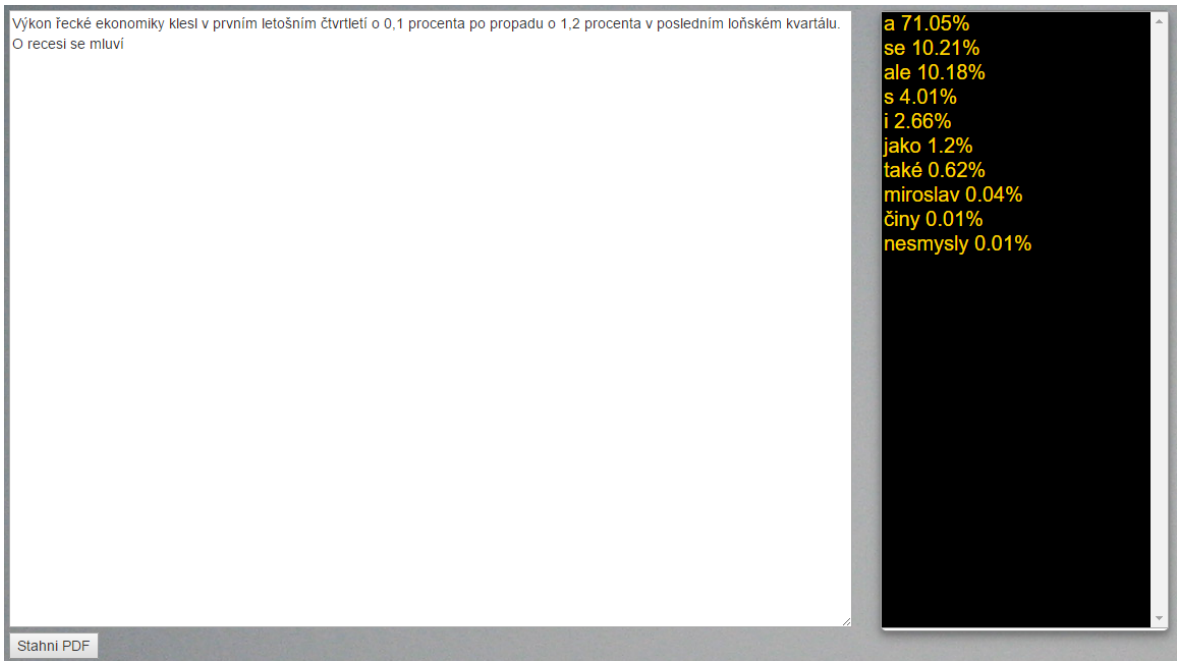
Slouží uživateli pro změnu vzhledu stránky ze světlého stylu na tmavý a obráceně, což je vhodné zejména pro úlevu očím při denním/nočním psaní.

Textové pole pro zadávání textů (prvek C)

Jedná se o hlavní uživatelský element sloužící pro zadávání textů. Tabulátor slouží pro přepnutí mezi hlavním textovým polem a tabulkou predikovaných slov. Pokud je během psaní zapnuta predikce, klávesa mezerník slouží pro spuštění algoritmu pro nápovědu následujícího slova.

Tabulka predikovaných slov (prvek D)

Je druhým uživatelským elementem a slouží pro výběr následujícího predikovaného slova. Klávesa tabulátor slouží k opětovnému přesunu do textového pole. Klávesou enter se vybrané slovo v tabulce vloží do psaného textu v textovém poli. U jednotlivých slov je spočtena Bayesova podmíněná pravděpodobnost a slova jsou seřazena od nejvyšší pravděpodobnosti. (viz Obr. 31)



Obr. 31. Ukázka textového pole (C) a tabulky predikovaných slov (D)

Export PDF (prvek E)

Po stisknutí tohoto tlačítka se veškerý text z textového pole exportuje do souboru PDF, který je pak možno uložit na disk.

9.4.2 Stránka Recalculate

Tato stránka obsahuje celkem 3 elementy:

- Tlačítko pro výběr volby přepočítávání dat v tabulkách cache a cache2
- Textové pole pro zadání hesla
- Tlačítko Submit pro odeslání požadavku na přepočítávání

Tato stránka je pouze pro oprávněné uživatele (Administrátory).

ZÁVĚR

Cílem této práce bylo vytvořit implementaci predikce textu spojenou s pokročilou analýzou vět a frází (bigramy a trigramy, které jsou podrobně rozebrány v kapitolách 1.4.1 a 1.4.2) a s využitím Bayesova teorému, diskutované v kapitolách 4.2 a 9.2. Součástí řešení, prezentovaného v této práci, bylo vypracování obsáhlé databáze slov. K tomu bylo třeba analyzovat texty z různých internetových zdrojů. Následně tyto data upravit, filtrovat a očistit, jak je popsáno v kapitole 9. Na základě výše zmíněného byl vytvořen prediktivní model spojený s webovou prezentací celé implementace (kapitola 9.4).

Program je určen pro všechny kategorie uživatelů a není potřeba předchozích znalostí žádného programovacího jazyka. Webové rozhraní je intuitivní a přehledné. Zároveň je připraveno i s administrativním přístupem, což umožňuje změnu nastavení programu pro cílené použití (například rozsáhlé úpravy využívaných databází pro bigramy, trigramy nebo sběr dat). Jinými slovy, tyto databáze jsou nezávislé na prezentované implementaci, a program tak může sloužit například jako „našeptávač“ v celé řadě odvětví.

Teoretická část této práce podrobně rozebírá principy predikce textu a základní přístupy ke sběru dat. Dále se soustředí na analýzu textu s příklady použití (bigram, trigram). Jsou zde také uvedeny praktické aplikace predikce textu.

V praktické části je popsán samotný vývoj programu. Jsou zde podrobně popsány postupy čištění vstupních dat, způsoby uložení slov v databázích, jejich komunikace s vlastním programem predikce textu a prezentace formou webového rozhraní.

Prezentována implementace predikce textu s rozsáhlou analýzou vstupních dat může sloužit jako učební nástroj pro pochopení základních principů analýz textu a sběru vstupních dat nebo jako pomocný nástroj pro psaní textu. Tato práce může být dále rozvinuta možností analýzy textu ze souboru uživatele, což povede k našeptávání na míru. Další možností je propojení se slovníkem za účelem opravy překlepů při psaní textu.

SEZNAM POUŽITÉ LITERATURY

[1] WEISS, Sholom M., Nitin. INDURKHYA a Tong ZHANG. *Fundamentals of predictive text mining*. ISBN 978-1-84996-225-4.

[2] What are N-Grams. *Text-analytics101* [online]. [cit. 2017-05-16]. Dostupné z: http://text-analytics101.rxnlp.com/2014/11/what-are-n-grams.html

[3] Jak mohou technologie pomoct s psaním. *I-sen* [online]. [cit. 2017-05-16]. Dostupné z: http://www.i-sen.cz/clanky/specialni-potreby/jak-mohou-technologie-pomoct-s-psanim-klavesnice

[4] SwiftKey Keyboard. *Google Play* [online]. [cit. 2017-05-16]. Dostupné z: https://play.google.com/store/apps/details?id=com.touchtype.swiftkey&hl=cs

[5] Eye Type. *Google Play* [online]. [cit. 2017-05-16]. Dostupné z: https://play.google.com/store/apps/details?id=org.opencv.eyetypevasko&hl=cs

[6] VOMLEL, Jiří. *Dvě aplikace Bayesových sítí* [online]. [cit. 2017-05-16]. Dostupné z: http://staff.utia.cas.cz/vomlel/sbornik-vomlel.pdf?q=vomlel/sbornik-vomlel.pdf

[7] Bayesianismus: Bayesova věta a bayesovská statistika. *Encyklopedie lingvistiky* [online]. [cit. 2017-05-17]. Dostupné z: http://oltk.upol.cz/encyklopedie/index.php5/Bayesianismus:_Bayesova_v%C4%9Bta_a_bayesovsk%C3%A1_statistika

[8] Programování stránek. *Jak psát web* [online]. [cit. 2017-05-16]. Dostupné z: https://www.jakpsatweb.cz/programovani.html

[9] PAVLÍČKOVÁ, Jarmila a Luboš PAVLÍČEK. *Vývoj klient/server aplikací v Javě*. Praha: Oeconomica, 2004. ISBN 80-245-0791-9.

[10] GILMORE, W. J. *Velká kniha PHP 5 a MySQL: kompendium znalostí pro začátečníky i profesionály*. Nové, 3. vyd. Přeložil Jan POKORNÝ. Brno: Zoner Press, 2011. Encyklopedie Zoner Press. ISBN 978-80-7413-163-9.

[11] JECHA, Tomáš. *Teorie databází* [online]. 2009 [cit. 2017-05-16]. Dostupné z: http://www.dotnetportal.cz/clanek/60/Lehky-uvod-teorie-databazi

[12] DUBEN, Stanislav. *Databáze* [online]. 2007 [cit. 2017-05-16]. Dostupné z: http://duben.org/zaklady-sql/zaklady-jazyka-sql-a-databazi-i-dil

[13] Relační vs. objektově-relační vs. objektové databáze. *Masarykova univerzita* [online]. [cit. 2017-05-16]. Dostupné z: https://www.fi.muni.cz/~xbatko/oracle/compare.html

[14] Model-View-Controller. *Vojtěch Hordějčuk* [online]. [cit. 2017-05-16]. Dostupné z: http://voho.eu/wiki/model-view-controller/

[15] SURREL, Grégoire. Diagram MVC. In: *Wikimedia* [online]. [cit. 2017-05-16]. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/b/b4/MVC_Diagram_%28Model-View-Controller%29.svg

[16] HEROUT, Pavel. *Učebnice jazyka Java*. 5., rozš. vyd. České Budějovice: Kopp, 2010. ISBN 978-80-7232-398-2.

[17] MACH, Jakub. *PHP pro úplné začátečníky*. 2., přeprac. a rozš. vyd. Brno: Computer Press, 2006. Bestseller (Computer Press). ISBN 80-251-1248-9.

[18] DOMES, Martin. *Tvorba internetových stránek pomocí HTML, CSS a JavaScriptu*. Kralice na Hané: Computer Media, 2005. ISBN 80-86686-39-6.

[19] WEMPEN, Faithe. *HTML a CSS: krok za krokem*. Brno: Computer Press, 2007. Krok za krokem (Computer Press). ISBN 978-80-251-1505-3.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

PHP	HyperText Preprocessor – skriptovací programovací jazyk
HTML	HyperText Markup Language – název značkovacího jazyka v informatice pro tvorbu webových stránek
CSS	Cascading Style Sheets – kaskádový styl v informatice pro popis způsobu zobrazení elementů na stránkách napsaných například v jazyce HTML
MySQL	Systém řízení báze dat – multiplatformní databáze
SQL	Jeden z deklarativních jazyků v informatice
MVC	Model-View-Controller

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

SEZNAM PŘÍLOH