FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Doporučovací modely založené na obrázcích
Student:	Martin Pavlíček
Vedoucí:	Ing. Tomáš Řehořek
Studijní program:	Informatika
Studijní obor:	Znalostní inženýrství
Katedra:	Katedra aplikované matematiky
Platnost zadání:	Do konce zimního semestru 2019/20

Pokyny pro vypracování

1) Nastudujte metody pro extrakci různých embeddingů (vektorizovaných reprezentací) z obrázků (např. barevný histogram, ORB, neuronová síť).
2) Seznamte se s problematikou doporučovacích systémů, zaměřte se zejména na algoritmus ItemKnn a metody evaluace offline úspěšnosti v těchto systémech (split/cross-validation, úspěšnostní metriky recall a catalog coverage).
3) Navrhněte doporučovací algoritmus ItemKnn s podporou různých metrik podobnosti.
4) Implementuje několik těchto metrik využívajících vektorizované reprezentace z metod zkoumaných v rámci rešerše.
5) Porovnejte úspěšnost algoritmu s různými podobnostními metrikami na dodaných testovacích datech a diskutujte naměřené výsledky.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Karel Klouda, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 19. února 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Doporučovací modely založené na obrázcích

Martin Pavlíček

Katedra aplikované matematiky
Vedoucí práce: Ing. Tomáš Řehořek

14. května 2018

Poděkování

Děkuji svému vedoucímu, Ing. Tomáši Řehořkovi, za příkladné vedení této práce. Rád bych také poděkoval své rodině za psychickou a morální podporu po celou dobu studií.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 14. května 2018 ……

České vysoké učení technické v Praze
Fakulta informačních technologií

Odkaz na tuto práci

Pavlíček, Martin. *Doporučovací modely založené na obrázcích*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2018.

Abstrakt

Cílem této práce je navrhnout, implementovat a porovnat několik *content-based* doporučovacích modelů, založených na různých metodách predikce obsahové podobnosti doporučovaných položek z jejich obrázků, se zaměřením na doporučování na webu a v prostředí online e-shopu. Tyto modely jsou navrženy jako alternativa k hojně využívaným modelům kolaborativního filtrování, které trpí řadou problémů jako je například *cold-start problem*.

Pro predikci obsahové podobnosti obrázků jsou mimo jiné použity moderní přístupy založené na algoritmu ORB nebo učení umělých neuronových sítí. Úspěšnosti implementovaných modelů jsou následně offline testovány na reálných datasetech zaznamenaných uživatelských interakcí několika významných online e-shopů pomocí technik *recall* a *catalog coverage*.

Model založený na umělé neuronové síti prokázal v offline testování nejlepší úspěšnost z navrhovaných modelů a byl nasazen do online A/B testu proti produkčnímu algoritmu, založeném na kolaborativním filtrování. Na testovaném vzorku 7435 uživatelů prokázal nově navržený model srovnatelnou proklikovost jako produkční algoritmus.

Klíčová slova Doporučovací systémy, personalizované doporučování, content-based doporučovací systém, zpracování obrazu, umělá neuronová síť, ORB, strojové učení

Abstract

The aim of this thesis is to design, implement and compare set of new content-based recommender models, using image processing methods for item similarity extraction with focus on web and e-commerce recommendations. Proposed models are meant as an alternative for a widely used collaborative filtering-based recommender systems, which have set of problems, including cold-start problem.

In this thesis, for image similarity extraction there will be used modern methods like ORB algorithm or artificial neural network. Proposed models will be offline tested in the latter part of this thesis on the recall and catalog coverage metrics on a real world e-shop datasets.

Artificial neural network-based recommender model had the best results in offline tests out of all proposed models and took place in online A/B test against production collaborative filtering-based recommender model. Total number of 7435 users attended this test and proposed model based on artificial neural network showed comparable click through rate as the production collaborative filtering-based model.

Keywords Recommender systems, personalized recommendation, content-based recommendation system, image processing, artificial neural network, ORB, machine learning

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Stejně jako se informační technologie staly nedílnou součástí každodenních činností běžného člověka, tak i množství informací s nimiž se denně musíme vypořádat stále roste se zrychlující se tendencí. Celosvětové webové portály udávají denní datový přírůstek v řádech tisíců záznamů a není tak v lidských silách tyto informace zpracovat ručně. Musíme naučit stroj, aby tyto informace předzpracoval pro nás a nabídl nám výtah jen těch nejužitečnějších informací, které nás opravdu zajímají.

Z těchto důvodů jsou už dnes téměř nezbytnou součástí každého prezenčního média i systémy pro vyhledávání, filtrování a doporučování relevantních informací se zaměřením na konkrétního uživatele. Spolu s množstvím dat rostou i požadavky na výkonnostní (ne)náročnost těchto systémů a kvalitu poskytovaných výsledků. Proto je důležité tyto systémy stále vyvíjet a zdokonalovat.

Nejen právě v posledním jmenovaném, tedy doporučovacích systémech, udělalo v posledních letech velký pokrok odvětví takzvaného machine-learningu, tedy automatizovaného strojového učení na základě předkládaných příkladů. Řešení založená na takovém přístupu poskytují velice dobré výsledky díky přirozeně naučeným pravidlům a generalizacím, které nemusel uměle navrhnout člověk a mají často i menší nároky na výkon. Se zvyšujícími se požadavky na doporučovací systémy by se právě machine-learning mohl stát mainstream technologií odvětví informačních systémů budoucích let.

V této práci spojím ověřené postupy ze světa konvenčních algoritmů a obohatím je o nejnovější poznatky z odvětví machine-learningu. Dohromady tak vznikne doporučovací model založený na robustních základech ověřených algoritmů, obohacený o sílu přirozeně naučených inferenčních pravidel získaných pomocí automatizovaného strojového učení. Doporučovací systémy založené na takovém modelu pak ocení například zákazníci e-shopů, kteří dostanou lepší doporučení a rychleji si tak vyberou kýžené zboží z množství možných alternativ, nebo čtenáři informačních portálů, jimž budou doporučovány relevantnější články a zprávy a dozví se tak přesně ty informace, které je zají-

mají. Na druhé straně, dobrý doporučovací systém může být pro daný e-shop nebo webový portál významnou konkurenční výhodou. Doporučí-li například sytém zákazníkovi správný produkt, zvyšuje se tím šance, že si zákazník produkt koupí v daném e-shopu a nedá přednost konkurenci. Používá-li naopak webový portál nepřímou techniku monetizace pomocí cílené inzerce, může doporučovací systém napomoci udržet návštěvníka na webovém portálu déle. Ten tak zkonzumuje více reklamy a přinese vlastníkům vetší zisk.

Kapitola 1

Cíl práce

Cílem této práce je navrhnout, implementovat a porovnat několik *content-based* doporučovacích modelů, určených k doporučování položek na internetu, se zaměřením zejména na personalizované doporučování v prostředí online e-shopů. Doporučovací modely budou založené na principu *content-based* doporučování nad obrázky položek a pro predikci uživatelovi zaujatosti v položce a následné doporučování těchto položek budou využívat různé metody pro stanovení obsahové podobnosti obrázků.

V první části této práce se teoreticky seznámím s doporučovacími systémy a výzkumem, který už byl v tomto odvětví informačních technologií odveden. Seznámím se s používanými typy doporučovacích modelů a technikami personalizovaného doporučování obecně. Následně prozkoumám několik metod počítačového vidění, vhodných pro extrakci informace o obsahu obrázků, jako jsou barevný histogram, algoritmus ORB nebo umělá neuronová síť.

V druhé části práce pak navrhnu rozšíření výše jmenovaných metod a algoritmů počítačového vidění o kroky potřebné k jejich využití pro stanovení obsahové podobnosti dvojice obrázků doporučovaných položek. Takto upravené algoritmy následně využiji k návrhu několika vlastních *content-based* doporučovacích modelů, určených k doporučování položek na webu s hlavním zaměřením na online e-shopy.

Hlavní částí této práce je samotná implementace všech navrhovaných částí a provedení offline evaluace kvality doporučení navrhovaných modelů. Ty budou testovány různými metodami, určenými k offline evaluaci doporučovacích modelů, na datasetech s reálnými daty několika významných e-shopů.

Cílem této práce je ukázat, že *content-based* doporučovací modely založené na doporučování podle obrázků mohou být zajímavou alternativou modelům založeným na kolaborativním filtrování, která netrpí problémy jako je například *cold-start problem*, za předpokladu, že pro stanovení obsahové podobnosti položek používají dostatečně kvalitní prediktor podobnosti obrázků doporučovaných položek, jakým může být například dobře natrénovaná neuronová síť.

Kapitola 2

Analýza

V této kapitole se budu tématy práce zabývat především teoreticky. Nejprve se zaměřím na doporučovací systémy a představím o jaký software se jedná, k čemu slouží a nastíním jejich nejčastější způsoby využití. Popíši základní typy doporučovacích modelů a nastíním jejich výhody a nevýhody.

V druhé části se pak podívám na několik technik počítačového vidění, určených ke zpracování a získání informací z obrazu. Tyto techniky v dalších částech práce rozšířím o další kroky, které mi umožní jejich využití ke stanovení obsahové podobnosti dvojice obrázků. Ty pak využiji ve vlastních *content-based* doporučovacích modelech k predikci podobnosti doporučovaných položek.

2.1 Doporučovací systémy

Než se pustíme dále do zkoumané tématiky, definujme si nejprve výraz „doporučovací systém“ formálně. Mluvíme-li v kontextu informačních technologií o takzvaném doporučovacím systému, myslíme tím software, určený ke shromažďování, zpracování a využití těchto informací za účelem doporučování položek pro specifického uživatele.

Doporučením nejčastěji rozumíme prioritizovaný výběr z velkého množství možných alternativ. Systém tak zpravidla dělá sestavením prioritizovaného seznamu položek, seřazených dle relevance pro cílového uživatele, o kterých se domnívá, že se mu budou líbit nejvíce. Systém tímto šetří uživateli nutnost procházet tyto alternativy ručně [1].

„Položkou“ v tomto případě rozumíme souhrnné pojmenování pro produkt, službu nebo jiný element z množiny doporučovaných kandidátů, ze které doporučovací systém vybírá a *doporučuje* pro uživatele seznam nejrelevantnějších. Může se jednat o produkt v e-shopu, článek v online periodiku, skladbu v online rádiu nebo například cílenou reklamu na internetu [1, 2]. Toto je jen krátký

výčet příkladů, který má za úkol nastínit jak rozsáhlé využití doporučovací systémy mají a kde všude mohou být užitečné.

V této práci budu používat dva velice blízké termíny, „doporučovací systém“ a „doporučovací model“. Přestože jsou si dosti podobné, každý z nich má nepatrně odlišný význam a je nutné je od sebe odlišovat. Doporučovací systém je označení pro komplexní řešení zastřešující získávání, zpracování, filtrování a uskladnění dat s následnou možností dotazování pro doporučení. Doporučovací model pak označuje samotné jádro doporučovacího systému, které se stará o co nejlepší doporučení na základě dostupných a předzpracovaných dat.

Základní doporučovací model můžeme formálně definovat jako funkci:

$$Model : \text{Interakce} \times \text{Položky} \times \text{Uživatel} \rightarrow D$$

Ta na základě vstupu doporučí cílovému uživateli N co možná nejrelevantnějších položek. N je v tomto případě nastavitelná konstanta udávající počet cílových doporučení D, tedy $N = |D|$. Vstupy se mohou pro různé doporučovací modely mírně lišit a sofistikovanější doporučovací modely mohou využít mnohem více parametrů a informací než zde uvádím. Obecně ale můžeme říct, že téměř všechny doporučovací modely v určité formě využívají známé interakce a množinu možných položek k doporučení pro cílového uživatele a proto je tato definice pro potřeby této práce dostačující.

Přestože mají doporučovací systémy jen relativně krátkou historii, jejich komerční nasazení, zejména v prostředí webu, nabylo rychle na popularitě, protože přináší jak uživatelům webových portálů, tak i jejich vlastníkům, řadu výhod a jsou proto intenzivně zkoumaným odvětvím v oblasti informačních technologií [1].

Mezi největší přínosy nasazení doporučovacích systémů podle [1] patří:

- **Zvýšení prodejů** nabízených položek je jednou z primárních motivací vlastníků komerčních webových portálů pro nasazení doporučovacího systému. Správné doporučení hledané položky často vede uživatele přímo ke koupi daného produktu bez nutnosti hledat u konkurence a může tak vést k přímému zvýšení zisků. Prodejem v tomto případě nemusí být nutně myšleno pouze zpeněžení produktu nebo služby. Pod tímto pojmem může být skryto i úspěšné navedení uživatele k přečtení článku na webu, zhlédnutí správně zacílené reklamy nebo jiné aktivitě, generující vlastníkům webového portálu zisk.

- **Diverzifikace prodejů** je taktéž silnou motivací k nasazení doporučovacího modelu. Některé položky mohou být pro uživatele obtížně nalezitelné a bez správné formy propagace tak uživatelé nemusí tyto položky nalézt, přestože by o ně mohli mít zájem. Plošná propagace za pomoci reklamy může být drahá a oproti cílenému doporučení uživatelům o kterých víme, že by mohli mít o dané položky zájem, neúčinná.

- **Zvýšení spokojenosti cílového uživatele** je další z řady benefitů, který může doporučovací systém přinést. Najdou-li uživatelé na webovém portálu obsah který je zajímá, zůstanou zde déle, což vede přímo či nepřímo ke zvýšení zisku a upevnění pozice portálu na trhu.

- **Porozumění uživatelům** a jejich potřebám na daném webu může být další, nepřímo viditelnou, motivací pro provozovatele webového portálu k nasazení doporučovacího systému. Dobře navržený doporučovací systém může jako vedlejší produkt své práce produkovat velké množství analytických informací, které mohou být využity k dalšímu rozšíření sortimentu o žádané položky nebo jinému zlepšení poskytovaných služeb.

Doporučovací modely rozdělujeme podle způsobu doporučování, na základě známých informací, do dvou kategorií.

2.1.1 Kolaborativní filtrování

Nejznámějším a v praxi také nejpoužívanějším přístupem k návrhu doporučovacího modelu je takzvané kolaborativní filtrování (CF) [3], občas označované také jako kolektivní filtrování. Vůdčí společnosti na webovém trhu, jakými jsou například Google [4], Amazon [5] nebo Yahoo! [6], používají ve svých doporučovacích systémech modely založené z větší části právě na kolaborativním filtrování.

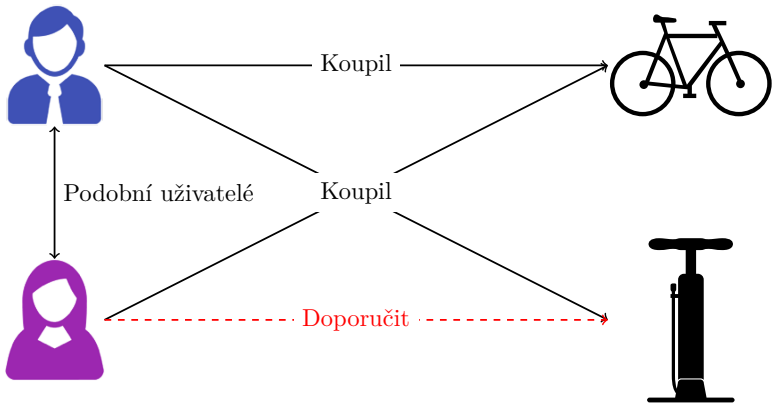
Systémy založené na predikčním modelu kolaborativního filtrování jsou postavené především na shromažďování dat o uživatelích a uskutečněných uživatelských interakcích a vyhodnocování těchto informací. Takovými informacemi může být:

- Uživatelova historie navštívených položek,
- explicitní hodnocení jaké položce uživatel zanechal,
- komentář,
- doba, jakou uživatel strávil prohlížením položky

a mnoho dalších metrik, které mohou být pro stanovení uživatelovi zaujatosti v položce relevantní.

Nasbírané informace následně doporučovací model vyhodnotí a na jejich základě sestaví jedinečný uživatelský profil pro každého uživatele v systému. Při následném doporučování pro konkrétního uživatele se pak model snaží v databázi identifikovat jemu nejpodobnější uživatele a doporučit mu položky, které se líbily jemu nejpodobnějším uživatelům, ale které sám ještě neviděl [3].

Příklad takového doporučení je ilustrován na obrázku 2.1. Na základě zakoupeného kola systém oba uživatele identifikoval sobě podobné (cyklisty).



Obrázek 2.1: Vizualizace doporučení modelu založeném na kolaborativním filtrování

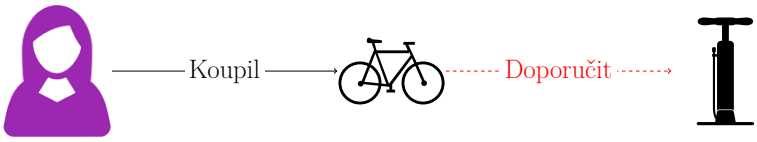
Na základě toho, že si modrý uživatel zakoupil pumpičku může systém dedukovat, že je pumpička pro tuto skupinu uživatelů zajímavým produktem a doporučí ji tedy uživatelce fialové.

Největší výhodou doporučovacích modelů založených na kolaborativním filtrování je, že operují především nad daty o uživatelských interakcích a s informacemi o doporučovaných položkách pracují často jen okrajově, pro zpřesnění výsledků [7]. Máme-li data o historii uživatelů a jejich interakcích, můžeme uživatele mezi sebou jednoduše porovnávat a tvořit doporučení na základě uživatelovi podobnosti s jinými uživateli, nehledě na doménu položek, které doporučuje.

To činí tyto doporučovací systémy z větší části doménově nezávislé. Jeden doporučovací systém, založený na kolaborativním filtrování, tak zpravidla můžeme téměř bez modifikací nasadit na různé webové portály, zabývající se různou tématikou, a očekávat, že bude fungovat stejně dobře, jako na prvním portálu. Z toho je zřejmá nepopiratelná výhoda jednoduchosti nasazení kolaborativního modelu, a potažmo na něm postaveném systému, oproti *content-based* doporučovacím modelům a systémům.

Největší výhoda těchto modelů, je ale také jejich největší slabinou. Model kolaborativního filtrování potřebuje pro kvalitní doporučení dostatek informací o cílových uživatelích, jejich preferencích vůči položkám anebo podobnost s ostatními uživateli. Nemá-li tyto informace k dispozici v dostatečném množství, jsou jeho doporučení většinou velice slabá a nerelevantní pro cílového uživatele. V případě, kdy do systému přijde nový uživatel, o kterém nemáme apriori žádnou informaci z jiného zdroje, nemůžeme takového uživatele efektivně srovnávat s jinými uživateli a hledat jemu podobné. V takovém případě mluvíme o takzvaném *cold-start* problému uživatele [8].

Cold-start problém ve specifické podobě rozeznáváme také u doporučova-



Obrázek 2.2: Vizualizace doporučení modelu založeném na obsahu

ných položek [8]. Z architektury kolaborativních doporučovacích modelů přímo vyplývá, že nemá-li model pro položku žádné zaznamenané interakce, nebude tato položka nikdy doporučena. To je problém, protože obvykle je žádoucí doporučovat uživatelům nové položky. Takovými může být například nové zboží v e-shopu, nové filmy a seriály v online televizi nebo nejnovější zprávy na informačním nebo zpravodajském portálu. Proto, aby se nové položky v první řadě dostaly do oběhu a získali jsme pro ně první interakce, musí je systém doporučovat více méně náhodně různým uživatelům a doufat, že se některému budou líbit. Tím ale na druhé straně zhoršuje celkovou úspěšnost systému, protože tak činí na úkor doporučení navrhovaných doporučovacím modelem.

2.1.2 Content-based doporučovací modely

Naproti kolaborativnímu filtrování stojí *content-based* (CB) doporučovací modely [1]. Pro tento typ modelů neexistuje ustanovené české označení. V této práci je tedy budu nazývat jako doporučovací modely založené na obsahu, což je volný překlad originálního *content-based recommendation model*.

Modely založené na obsahu pracují především s informacemi o doporučovaných položkách. Využívají nejrůznější algoritmy pro stanovení obsahové podobnosti doporučovaných položek a cílovému uživateli doporučují položky, které zapadají do jeho osobního profilu. Ten systém sestaví na základě položek, které už uživatel v minulosti navštívil a na základě hodnocení, které jim zanechal, nebo které systém jinak stanovil na základě uživatelových akcí [9].

Je-li po systému následně požadováno doporučení pro konkrétního uživatele, systém se mu snaží doporučit položky nejpodobnější těm, které už cílový uživatel viděl a kladně hodnotil v minulosti [10]. Na obrázku 2.2 je takové doporučení znázorněno. Systém předem zná vztah mezi pumpičkou a kolem. Pro identifikovaného cyklistu pak tedy přímo doporučuje pumpičku, bez nutnosti další znalosti o tom, kteří uživatelé také v minulosti kupovali pumpičky.

Podobnost doporučovaných položek bývá stanovována na základě uměle extrahovaných atributů cílových položek. Vzhledem k tomu, že významné atributy jsou často specifické pro danou doménu, nebývá jednoduché nasadit zavedený doporučovací systém z jedné domény na jinou. Tímto se *content-based* doporučovací systémy stávají jen obtížně přenositelnými mezi doménami a ztrácejí v tomto ohledu na relativně jednoduše nasaditelné modely, založené na kolaborativním filtrování [9].

Velkou výhodou, kterou mají doporučovací modely založené na obsahu oproti kolaborativnímu filtrování, je jejich tolerance vůči nedostatku informací o cílovém uživateli. Vzhledem k tomu, že tyto modely staví hlavně na datech o položkách, novým uživatelům je model připraven doporučovat relevantní položky už na základě jedné navštívené nebo ohodnocené položky.

Vzhledem k tomu, že doporučovací modely založené na obsahu hodnotí doporučované položky na základě jejich vlastností a nikoli na základě toho, jakým uživatelům se líbily v minulosti, netrpí tyto modely na *cold-start item* problém, jako modely založené na kolaborativním filtrování.

Z charakteru modelu je patrné, že doporučuje téměř výhradně položky podobné těm, které už uživatel zná. Z tohoto důvodu jsou doporučovací modely založené na obsahu nevhodné pro prostředí, kde je důležité doporučovat uživatelům rozmanité položky a rozšiřovat jejich obzory.

Dalším problémem modelů založených na obsahu je otázka, jak vlastně stanovit, které atributy dané položky jsou důležité, jakou váhu jim přikládat a jak daný atribut kvantizovat. Tento problém většinou řeší doménový expert. Pokud je doménovým expertem člověk, bývá tento proces časově náročný a vzniká tu taktéž riziko, že atributy odhadne špatně a doporučovací model kvůli tomu nebude fungovat optimálně [9]. Zde *content-based* systémy ztrácí na kolaborativní filtrování v ohledu jednoduchosti a rychlosti nasazení a jedná se často o jeden z hlavních faktorů, kvůli kterému dávají vývojáři přednost modelům založeným na kolaborativním filtrování.

S rapidním nástupem strojového učení se ale otevřela možnost, že by doménovým expertem mohl být i stroj, přesněji strojově trénovaný predikční model. Právě toho bych chtěl v této práci využít a navrhnout vlastní doporučovací model s využitím stroje jako doménového experta na podobnost doporučovaných položek.

2.1.3 Item K-NN

Kolaborativní filtrování poskytuje velice přesná doporučení za předpokladu, že máme dostatek dat o uživatelských interakcích. Na druhé straně se ale jen obtížně vypořádává se situací, kdy máme k dispozici jen řídká data a nejsme tak schopni efektivně identifikovat podobně smýšlející uživatele právě z důvodu nedostatku nasbíraných uživatelských interakcí.

V [11] autor navrhuje řešení problému řídké databáze uživatelských interakcí spojením obou výše popsaných technik, CF a CB doporučování. Toto řešení staví na základní myšlence CB doporučování a doporučení sestavuje na známých hodnoceních položek cílovým uživatelem. Nepotřebuje tak stanovovat vzájemnou podobnost uživatelů k sestavení seznamu nejbližších sousedů cílového uživatele. Díky tomu se efektivně vyhýbá *cold-start user* problému.

Klíčový problém CB doporučovacích modelů, stanovení podobnosti dvojice položek, pak navrhovaný algoritmus řeší pomocí metody založené na kolaborativním filtrování. Zatímco uživatelé do systému přicházejí relativně často

a v našem zájmu je zejména pro nové uživatele doporučovat co nejrelevantnější položky, seznam doporučovaných položek bývá o něco stálejší a položky tak mívají o něco větší množství zaznamenaných interakcí než uživatelé. Při stanovování podobnosti dvojice položek tak algoritmus využívá tohoto předpokladu.

Nejprve pro každou z dvojice porovnávaných položek sestaví vektor, kde i-tý záznam tohoto vektoru odpovídá hodnocení i-tého uživatele v systému. Pokud uživatel ještě položku neviděl, bude příslušná pozice vektoru vyplněna nulou. Podobnost srovnávaných položek je pak rovna podobnosti těchto vektorů, podle některé z vektorových podobností.

V případě například kosinové podobnosti, tak jak ji definuje [11], pak můžeme formálně definovat funkci Sim : Položky $\times$ Položky $\rightarrow \mathbb{R}$, predikující podobnost dvojice položek P_1 a P_2, jako:

$$Sim(P_1, P_2) = Sim(P_2, P_1) = \frac{X_1 \cdot X_2}{|U|}$$

$$\begin{aligned} \text{Uživatelé} &= \{u \in U : (u, P_1) \in I \wedge (u, P_2) \in I\} \\ X_1 &= \{Rating(u, P_1), : u \in \text{Uživatelé}\} \\ X_2 &= \{Rating(u, P_2), : u \in \text{Uživatelé}\} \end{aligned}$$

kde U označuje množinu všech uživatelů v systému, I množinu všech zaznamenaných interakcí a funkce $Rating$: Uživatelé $\times$ Položky $\rightarrow \mathbb{R}$ vrací známé hodnocení položky uživatelem u.

Samotné doporučení pro cílového uživatele pak algoritmus provádí sestavením prioritizovaného seznamu doporučení, seřazeného na základě predikovaných hodnocení položek. Predikci hodnocení pro každou položku algoritmus vypočte jako součet hodnocení všech položek, které cílový uživatel hodnotil v minulosti, vážené podobností s predikovanou položkou.

Pro další zlepšení přesnosti doporučení algoritmu je možné limitovat množství uvažovaných sousedů pro každou položku na pouze K nejpodobnějších sousedů. Tím se v doporučovaném seznamu zmenší počet položek, které jsou takzvaně „podobné všem“, ve prospěch položek nejpodobnějších těm, které už uživatel v minulosti hodnotil kladně.

Formálně tak můžeme utilitní funkci F : Uživatelé $\times$ Položky $\rightarrow \mathbb{R}$, kterou systém použije k predikci hodnocení položky P uživatelem U, definovat jako:

$$F(U, P) = \sum_{(U,P') \in I_U} Rating(U, P') \cdot \begin{cases} Sim(P, P') & P' \in Q_{P(K)} \\ 0 & \text{Jinak} \end{cases}$$

kde $Q_{P(K)}$ označuje množinu K nejpodobnějších položek k položce P a I_U množinu známých interakcí pro uživatele U:

$$I_U = \{(u, p) \in I \ : u = U\}$$

Zatím co tento přístup velice efektivně řeší dříve popsaný *cold-start user* problém kolaborativního filtrování, *cold-start item* problém zde stále zůstává neřešen. Nemá-li systém pro danou položku dostatek zaznamenaných interakcí, nemůže ji efektivně srovnávat s ostatními a stanovovat jejich podobnost, a tím pádem ani efektivně doporučovat. Právě to se pokusím adresovat při návrhu vlastního doporučovacího modelu v další části práce.

2.2 Evaluace doporučovacích modelů

Před implementací jakéhokoli softwaru je důležité si nejprve stanovit cíle a účel, jakému bude daný software sloužit. Programátor by si měl nejprve ujasnit, jak bude daný software fungovat, jak kvantizovat metriky podle kterých bude daný software hodnocen a jak hodnotit míru splnění stanovených cílů. Stejné platí i v případě doporučovacích systémů. Na ty můžeme, kromě typických kritérií pro hodnocení softwaru jakými jsou nároky na výkon, paměť nebo diskový prostor nutný k uložení dat, uplatňovat i kritéria specifická pro doporučovací systémy, zejména pak na kvalitu jejich doporučení.

V případě doporučovacích systémů rozdělujeme evaluační techniky na základě způsobu testování do dvou kategorií, online a offline.

2.2.1 Online evaluace

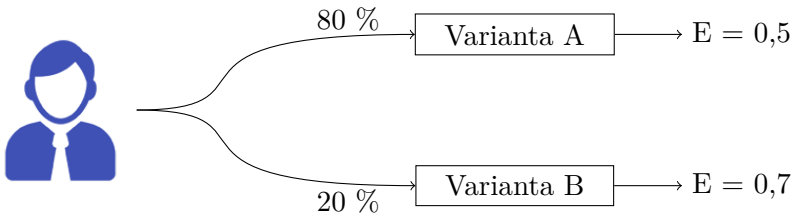
Jako online evaluační techniku označíme jakýkoliv proces validace, měření nebo testování doporučení poskytovaných doporučovacím systémem nebo doporučovacím modelem, které provádíme na živých uživatelých, ať už s jejich vědomím nebo ne. Hlavním poskytovatelem zpětné vazby je přímo uživatel sám nebo implicitní závěry založené na jeho akcích, které vyhodnotí systém [12].

Modelovým příkladem online evaluace doporučení je situace, kdy je doporučovací systém požádán o doporučení pro konkrétního uživatele. Poskytnutá doporučení představíme uživateli a sledujeme, zda uživatel navštívil některou z doporučených položek. Pokud ano, můžeme toto doporučení považovat za validní a hodnotit doporučovací model kladně [13].

Online evaluace může poukázat na rezervy v doporučovacím modelu a na základě jejích výsledků můžeme objektivně hodnotit relevanci doporučení poskytovaných modelem. Na základě nasbíraných dat můžeme také doporučovací model vylepšit nebo úplně přepracovat některé jeho části, které se ukázaly být nedostačující.

2.2.1.1 A/B testování

A/B testování je široce používanou online testovací technikou. Jedná se o kontrolovaný experiment, kdy testovací subjekty vystavíme dvojici velice podobných alternativ v jinak naprosto totožném prostředí s cílem nestranného porovnání těchto alternativ. Testovací subjekty, tedy uživatele, nejprve podle



Obrázek 2.3: Vizualizace A/B testu

předem zvoleného poměru rozdělíme do skupin. Podle příslušnosti do skupiny pak každému uživateli předložíme testovanou alternativu, připravenou pro jeho skupinu. V průběhu testu pak sledujeme, která z testovaných alternativ dosáhne lepších výsledků v cílových parametrech [13].

Chceme-li do produkce nasadit nový doporučovací model, zpravidla nás více než absolutní čísla, zajímá srovnání se stávajícím řešením. Je nový přístup lepší než stávající řešení? O kolik, a ve kterých případech? A/B test je velice vhodný právě pro tento případ.

Chceme-li nový doporučovací model porovnat se stávajícím řešením, nasadíme do testu jako alternativu A stávající řešení. Za alternativu B pak nasadíme nové řešení a dle cílových parametrů sbíráme data. Motivace za nasazením nového řešení mohou být různé. Od zvýšení proklikovosti doporučovaných položek, přes zvýšení návštěvnosti méně navštěvovaných položek cílového portálu, až po zvýšení prodeje některých produktů v online e-shopu. Je důležité tato kritéria znát předem a při samotném testování sbírat všechna relevantní data.

Další výhodnou vlastností A/B testu nového řešení je, že máme pod kontrolou míru nebezpečí. Při nasazování nového doporučovacího modelu riskujeme, že bude neotestované řešení uživatelům poskytovat nerelevantní doporučení a přijdeme tím například o část zisku. Míru rizika tak můžeme kontrolovat nastavením poměru testovaných uživatelů, kterým budeme v testu ukazovat doporučení nového řešení. Máme-li v nový model jen malou míru důvěry, zvolíme k testu jen například 5 % uživatelů. Naopak, jsme-li si novým řešením jistí, můžeme zvolit 50 % nebo i více.

Na obrázku 2.3 je znázorněna procedura A/B testu. Testovací poměr zde byl zvolen 4:1. Z obrázku je vidět, že testovaná alternativa B podávala v cílovém parametru, oproti alternativě A, lepší výsledky o celých 28 %. Pomocí statistických nástrojů následně rozhodneme, na kolik je tento výsledek průkazný vzhledem k velikosti testovacího vzorku uživatelů a poměru jejich rozdělení.

Později v této práci využiji A/B test k otestování nejlepšího z navržených doporučovacích modelů. Model bude měřen na proklikovost doporučovaných položek, která zde bude interpretována jako relevance doporučených položek. Poměr proklikovosti pro testovaný model definujeme přirozeně jako počet doporučení, na které uživatel klikl, vůči celkovému počtu doporučení, která byla

uživateli předložena [14]. Doporučením je zde myšlena celá sada doporučených položek, nikoli jedna doporučená položka v sadě.

2.2.2 Offline evaluace

Jako offline evaluační techniku označíme jakýkoli proces validace, měření nebo testování doporučení poskytovaných doporučovacím systémem nebo doporučovacím modelem, které provádíme na předem nasbíraných datech. Offline evaluace se nikdy neúčastní živí uživatelé, ať už vědomě nebo nevědomě, a výsledky jsou vždy vyhodnocovány nad historickými daty, nikoli na sběru uživatelova explicitního nebo implicitního hodnocení doporučení, stanoveného na základě vyhodnocování uživatelova chování [13].

Offline evaluace je rychlou a hlavně levnou alternativou k online evaluaci. Nasazujeme-li do produkce nový doporučovací systém, existuje riziko, že pro část uživatelů bude neotestovaný systém doporučovat naprosto nevalidní položky. Tím riskujeme nevratnou ztrátu těchto uživatelů, ztrátu na zisku nebo i újmu na reputaci webového portálu. Z těchto důvodů je důležité doporučovací systém nejprve otestovat v bezpečném prostředí, než ho nasadíme do produkce.

Online test je také nutné provádět po nějaký časový úsek, aby se ho účastnilo dostatečné množství uživatelů a jeho výsledky byly reprezentativní a průkazné. Čím více různých modelů nebo nastavení jednoho modelu chceme otestovat, tím delší dobu celkový test zabere [12]. Offline testy jsou naproti tomu velice rychlou alternativou, protože nemusíme čekat, až se uživatelé dostaví na test. Data už máme připravená a jediným faktorem, určujícím rychlost provedení testů, je tak nám dostupný výpočetní výkon. Můžeme proto v kratším čase otestovat vetší množství alternativních modelů a jejich parametrů a vytipovat si tak nejvhodnější kandidáty pro online test.

2.2.2.1 Split a cross validace

Split validace je statistickou metodou pro offline predikci úspěšnosti, nejen doporučovacích modelů, na předem nasbíraných datech.

Offline split-evaluaci provádíme tak, že známá data v námi zvoleném poměru rozdělíme na dvě disjunktní podmnožiny, trénovací a testovací skupinu. Data do těchto skupin můžeme rozdělovat na různých úrovních. Na úrovni uživatelů, doporučovaných položek nebo i jednotlivých známých interakcí. V každém případě je nutné, aby nebyly jednotlivé skupiny nevyvážené z hlediska obsahu a byla každá reprezentativním vzorkem celkového obrazu dat. V opačném případě může být model v testovací fázi neprávem penalizován, přestože se správně naučí informace z trénovací sady.

Trénovací množina obsahuje známé výsledky a model na nich, dle jeho možností, učíme. V této fázi nám jde především o to, dosáhnout co nejlepší generalizace informací obsažených v testovací sadě a vyvarování se obou ex-

trémů, jak přeučení, tak i nedoučení modelu. Kvalitu doporučení testovaného modelu průběžně kontrolujeme pomocí testovací množiny.

Základním předpokladem pro validitu naměřených výsledků, které budou reprezentovat reálnou kvalitu modelu, je striktní oddělování dat na kterých je model trénován, a dat, na kterých bude následně testován. Jinými slovy, nesmí se nikdy stát, že by model v testovací části „viděl“ záznamy, na kterých bude následně testován.

Rozdělení datasetu na trénovací a testovací sadu, tak aby každá věrně reprezentovala celkový obraz dat, má svá úskalí. Nejsme-li schopni tuto podmínku zaručit, riskujeme že naměřené výsledky nebudou odpovídat skutečné kvalitě modelu. Tento problém se snaží řešit metody cross validace. Ty se od split validace liší tím, že celý dataset rozdělí na K exkluzivních sad. Proces trénování a testování se pak opakuje v K iteracích, kde model vždy trénujeme na $K-1$ sadách a testujeme na jedné zbývající sadě. V každé iteraci je testovací sada jiná, než ve všech ostatních iteracích. Celková úspěšnost modelu je pak stanovena jako průměr všech K iterací testu [15]. Tímto způsobem metody cross validace efektivně eliminují možnost nevyváženého rozdělení trénovací a testovací podmnožiny.

Jednou z nejznámějších metod cross validace je takzvaný K-Fold. Používáme-li tuto metodu, rozdělíme testovací dataset do K podobně velkých podmnožin a model podrobíme testování podle popisu cross validace výše [15]. Vizualizace K-Fold cross validace je na obrázku 2.4.

Další hojně používanou metodou cross validace je metoda *Leave-one-out*. Tato metoda rozdělení testovacího datasetu je speciálním případem metody K-Fold, kde K položíme rovno počtů položek v testovacím datasetu [16]. Vzhledem k velkému množství iterací, které je nutné pro kompletní test projít, se tato metoda používá především na menších datasetech, zatímco K-Fold je upřednostňován na datasetech o větším objemu dat.

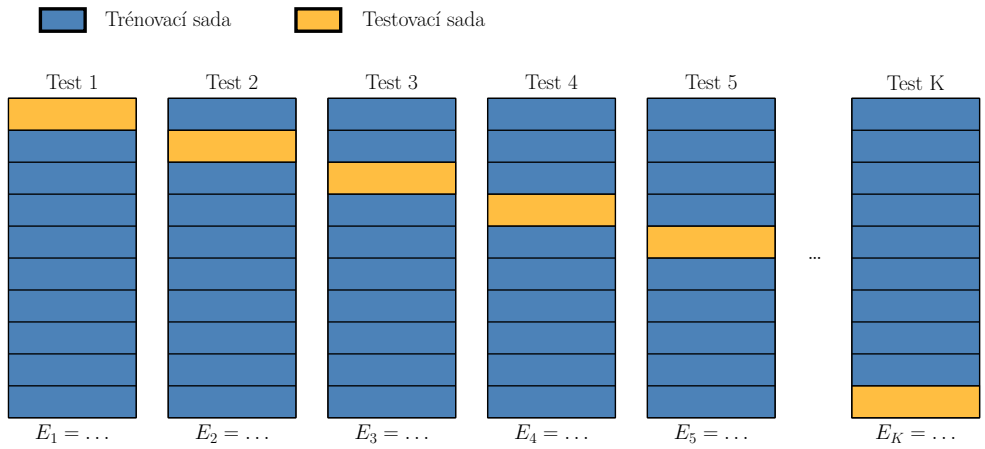
Při offline testování je důležité mít na paměti, že data nad kterými jsou modely testovány nemusí být úplně čistá a musíme počítat s tím, že výsledky offline testů nemusí stoprocentně odpovídat skutečné kvalitě testovaných modelů. Proto je nutné výsledky offline testů brát s rezervou a pokud je to možné, ověřovat jejich výsledky online testy.

2.2.2.2 Recall

Recall je uznávanou metodou pro offline evaluaci relevance doporučení poskytovaných doporučovacími modely [17]. Zakládá se na jednoduchém předpokladu, že pokud modelu „skryjeme“ jednu nebo více známých interakcí položek, které uživatel hodnotil kladně, model by měl být schopný tuto interakci reprodukovat a doporučit „schované položky“.

V této práci použiji metodiku testování *recallu* tak, jak ji představuje [18].

Při testování *recallu* modelu nejčastěji pomocí některé ze split nebo cross-validačních metod rozdělíme testovací dataset na trénovací a testovací pod-



Obrázek 2.4: Vizualizace K-Fold cross validace

množiny. Testovaný doporučovací model nejprve natrénujeme na trénovací podmnožině. Pro každého uživatele v testovací podmnožině pak *recall* získáme tak, že metodou *leave-one-out* vždy „schováme“ jednu kladně hodnocenou položku ze seznamu známých interakcí pro daného uživatele. Tuto položku nazveme testovací. Model pak pro cílového uživatele necháme doporučit N položek a sledujeme, zda se testovaná položka objeví v doporučených položkách. Formálně tento proces popisuje následující vzorec:

$$R_u = \frac{\sum_{(u,p)\in I_u} \begin{cases} 1 & (u,p) \in Model(I \setminus (u,p), P, u) \\ 0 & \text{Jinak} \end{cases}}{|I_u|}$$

N je v tomto případě volitelná konstanta, kterou volíme na základě cílového použití testovaného modelu. Nejčastěji bývá na přítomnost testované položky testována první, prvních 5 a prvních 10 doporučených položek. Není ale neobvyklé, testovat *recall* i na jiném N. Tento parametr je volitelný a je nejlepší ho volit s ohledem na výsledné použití testovaného modelu.

Celkový *recall* pro testovaný dataset pak získáme jako průměr *recallu* jednotlivých uživatelů:

$$Recall = \frac{\sum_{u\in U} R_u}{|U|}$$

Za povšimnutí stojí normalizace výsledného *recallu* napříč uživateli. Tímto se zajistí, že *recall* všech uživatelů v datasetu bude uvažován se stejnou důležitostí a nestane se tak, že by jeden uživatel s velkým množstvím ohodnocených položek ovlivnil celé měření. To je speciálně důležité, vyskytují-li se v datasetu robotičtí uživatelé, takzvaní crawleři.

2.2.2.3 Catalog coverage

V návaznosti na *cold-start item* problém, jehož řešením se budu v rámci navrhovaných modelů zabývat, jsem zvolil jako sekundární metodu pro jejich offline evaluaci metodu *catalog coverage*. V [19] autor poukazuje na to, že je důležité se při evaluaci doporučovacích modelů nezaměřovat pouze na *recall*, ale přihlížet také k sekundárním ukazatelům kvality doporučení, jako je například *catalog coverage* nebo *prediction coverage*.

Mírně zde předběhnu a prozradím, že mnou navrhované doporučovací modely nebudou mít žádné požadavky na doporučované položky a *prediction coverage* bude vždy 100 %. Z tohoto důvodu nemá význam se v této práci *prediction coverage* nadále zabývat a navrhované modely budu evaluovat jen na základě *prediction coverage*.

Volně řečeno, metrika *catalog coverage* udává, jaký poměr doporučovaných položek uživatelé efektivně uvidí, v poměru k celkovému počtu položek v datasetu. Na jedné straně je dobré, když doporučovací model doporučuje relevantně, na straně druhé je ale také důležité, aby model nedoporučoval stále to samé.

Formálně pak *catalog coverage* pro doporučovací model *Model* na testovacím datasetu s uživateli U, zaznamenanými interakcemi I a doporučovanými položkami P definujeme jako:

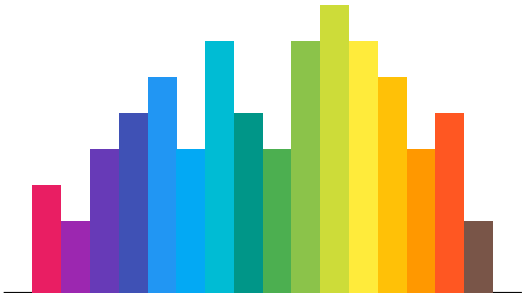
$$CatalogCoverage = \frac{\left| \bigcup_{u \in U} Model(I, P, u) \right|}{|P|}$$

2.3 Metody pro zpracování obrazu

Počítačové vidění, rozpoznávání obrazu a práce s obrazem je velice atraktivním a žádaným odvětvím výzkumu v oblasti informačních technologií. Má taktéž velice široké uplatnění v produkčních aplikacích. Počínaje od domácích úklidových robotů, přes automatizované řízení provozu na rušných dopravních uzlech, v závislosti na aktuální dopravní situaci, až po rozpoznávání tváří na hraničních přechodech a letištích pro identifikaci hledaných osob. Toto je jen několik příkladů praktického využití technik pro zpracování obrazu.

V průběhu let bylo vymyšleno a navrženo mnoho přístupů a způsobů určených ke zpracování obrazu, plnících různé úkoly. Podle výsledného použití extrahované informace mohou být některé techniky vhodnější než jiné, protože každá z technik klade důraz na jiné parametry, atributy a prvky v obraze.

V této části popíši několik technik, zabývajících se právě zpracováním obrazu, vhodných k extrakci informace obsažené v obrázcích. V následující kapitole pak navrhnu jejich rozšíření a způsoby, jak tyto techniky použít ve vlastních *content-based* doporučovacích systémech.



Obrázek 2.5: Vizualizace barevného histogramu

2.3.1 Barevný histogram

Barevný histogram je jednou z prvních zkoumaných a představených technik, zaměřených na zpracování a získání informace z obrazu vůbec. První zmínky o něm můžeme nalézt už v 70. letech 20. stol. [20]

Mezi jeho největší výhody patří jednoduchost a relativní výkonnostní nenáročnost. I přesto ale prokázal vysokou míru kvality extrahované informace při naivním extrakci obsahu obrázků do vektoru nízké dimenze. Vizualizace takového vektoru je zachycena na obrázku 2.5.

V [21] autor popisuje způsoby pro efektivní využití barevných histogramů k rychlému a stabilnímu odhadu podobnosti předmětů na různých obrázcích. Úspěšně zde používá barevný histogram k rychlému odhadu obsahu obrázku a ukazuje, že i takto jednoduchá technika může mít velice kvalitní výsledky a že jednoduchost a rychlost může být výhodou.

Z těchto důvodů jsem si barevný histogram vybral jako první techniku pro zpracování obrazu pro svůj doporučovací model.

2.3.2 ORB

Algoritmus „Oriented FAST and Rotated BRIEF“, zkráceně ORB, je pokročilou technikou určenou ke zpracování obrazu. Prvně byla představena v roce 2011 [22]. Jedná se tedy o poměrně mladou technologii s moderním přístupem k problému. Algoritmus ORB staví a značně zlepšuje původní myšlenky úspěšných algoritmů FAST a BRIEF a svými vlastnostmi dominuje dobře známé matadory odvětví počítačového vidění a zpracování obrazu jakými jsou algoritmy SIFT a SURF, jak po stránce výkonu a rychlosti, tak zejména po stránce kvality a stability zpracování obrazu [22].

Algoritmus ORB se skládá ze dvou hlavních kroků. V prvním kroku algoritmus detekuje významné body na cílovém obrázku. Často jsou takovými body místa s vysokou koncentrací kontur. Samotné vyhodnocení a označení bodů je ale implementační detail algoritmu. Výstupem po tomto kroku je vektor významných bodů, spolu s jejich parametry. Algoritmus pro detekci bodů

Obrázek 2.6: Vizualizace párování bodů algoritmem ORB na obrázcích šatů

je stabilní, takže pro dva stejné obrázky vrátí vždy stejné body se stejnými parametry.

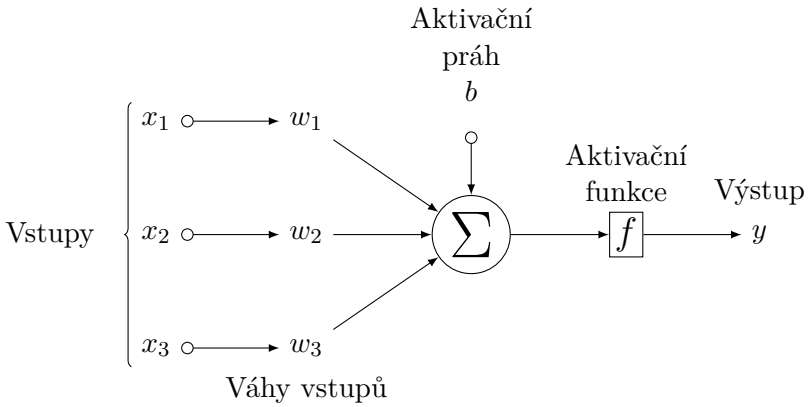
V druhém kroku je na řadě párování podobných bodů, extrahovaných z různých obrázků. Příklad takového párování bodů je vidět na obrázku 2.6, kde můžeme vidět dva podobné, přesto různé obrázky dámských šatů. Úkolem algoritmu je obrázky prozkoumat a najít a namapovat na sebe podobná místa [22].

Původně byl algoritmus navržen k detekci a párování stejných předmětů na různých obrázcích, ale v této práci navrhnu způsob jak ho využít i ke stanovení obsahové podobnosti dvojice obrázků.

2.3.3 Umělá neuronová síť

Přestože teoretický koncept umělých neuronových sítí je nám dlouho známý, prvně byl představen již v roce 1943 [23], praktického využití se jim dostalo až v posledních letech. Mimo jiné za to může zejména jejich relativně velká náročnost na výpočetní výkon, potřebný ve fázi trénování modelu. S masivním nástupem nárůstu výkonu v oblasti hardwarově akcelerovaného počítání na grafických procesorech, se ale tento problém značně zmenšil a umělé neuronové sítě se začaly nasazovat v nejrůznějších aplikacích, zaměřených na zpracování dat, obrazu, a jsou hojně využívány v nejrůznějších odvětvích umělé inteligence [24].

Základní teoretickou stavební jednotkou umělé neuronové sítě je jeden neuron. Ten můžeme vnímat jako autonomní buňku, která má N vstupů, N konfigurovatelných vah a právě jeden výstup. Podle vah na vstupech neuronu,



Obrázek 2.7: Příklad neuronu se třemi vstupy

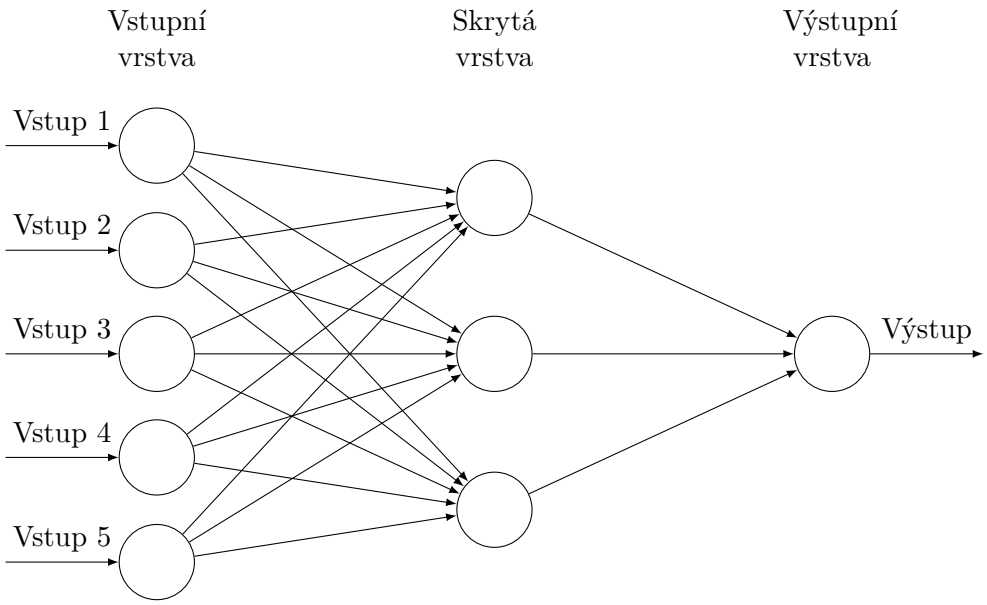
specifických pro každý neuron, je neuron různě citlivý na podněty, signály, přicházející na jeho vstupy. Podle intenzity signálů na vstupech a aktivační funkce pak neuron sám začne emitovat signál specifické intenzity na svém výstupu [25]. Pro ilustraci je příklad jednoho neuronu se třemi vstupy zobrazen na obrázku 2.7.

Spojením neuronů do vrstev neuronové sítě tak, že napojíme výstupy jednoho nebo více neuronů z předchozí vrstvy na vstupy neuronů ve vrstvě následující, vznikne základní model umělé dopředné neuronové sítě. Příklad takové hustě propojené dopředné neuronové sítě o třech vrstvách a devíti neuronech je vizualizován na obrázku 2.8.

Správné váhy pro vstupy neuronu umělé neuronové sítě získáme takzvaným procesem učení. Tento proces spočívá v opakovaném předkládání série příkladů vstupů a k nim správných výstupů a zaznamenávání chyb, kterých se síť v průběhu testování dopustila. Chyba na výstupu je pak vyhodnocena učícím algoritmem a váhy na vstupech neuronů jsou upraveny tak, aby se zvýšila přesnost výstupů. Tyto kroky opakujeme tak dlouho, dokud není chybovost výstupu sítě menší než je požadovaná úroveň [25].

Tento proces je náročný jak na množství příkladů v trénovací sadě, tak i na jejich diverzitu. Neuronová síť je jako každý jiný učící model náchylná k takzvanému přeučení. Proto musí být trénovací dataset dostatečně bohatý i na diverzitu předkládaných příkladů, tak aby k jejímu přeučení nedocházelo [25].

Trénovací proces je zpravidla složený z mnoha iterací a proto je náročný i z pohledu potřebného výpočetního výkonu. Čím hlubší je navržená síť a čím více má neuronů, tím složitější operace dokáže provádět a tím složitější problémy dokáže řešit. S počtem neuronů ale také roste složitost trénování takové sítě a při návrhu je tedy nutné správně odhadnout potřebnou komplexitu sítě pro daný úkol [26].



Obrázek 2.8: Příklad dopředné umělé neuronové sítě se třemi vrstvami

Kapitola 3

Návrh

V této kapitole navrhnu rozšíření metod pro zpracování obrazu, představených v předchozí kapitole, tak aby byly použitelné pro stanovení obsahové podobnosti dvojic obrázků doporučovaných položek. Zároveň také navrhnu zapojení těchto metod do upraveného doporučovacího algoritmu Item K-NN.

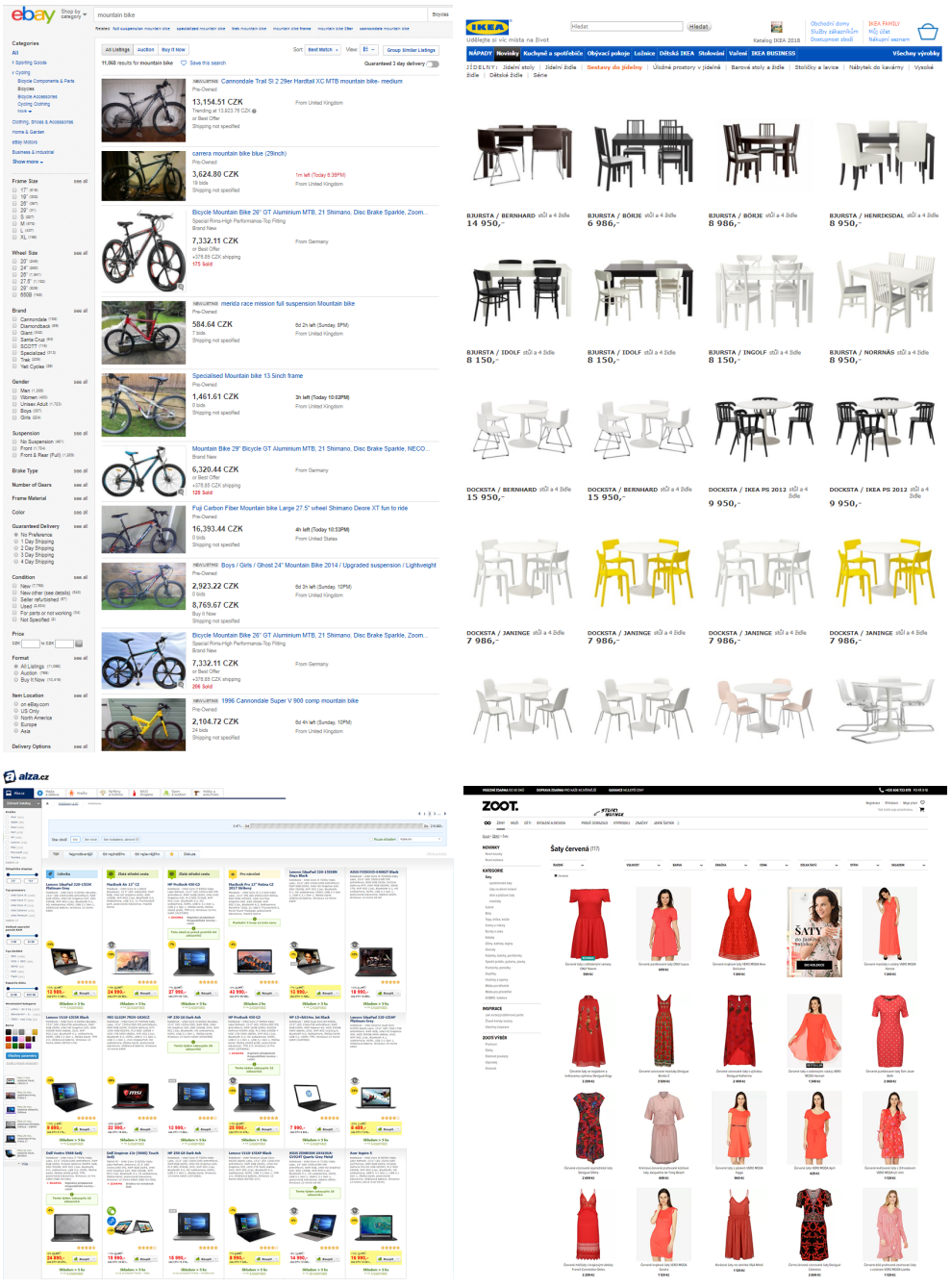
3.1 Content-based doporučovací model

Podívejme se nejprve na několik screenshotů významných českých i zahraničních e-shopů a aukčních portálů na obrázku 3.1.

Přestože každý z nich je zaměřený na trochu jiný sortiment zboží, jedno mají všechny společné. Každý produkt, který na své webové prezentaci propagují, má obrázek, který zabírá nezanedbatelnou část prostoru vyhrazeného pro daný produkt. Z toho můžeme usoudit, že vizuální prezentace produktu hraje pro případného zákazníka při výběru produktu důležitou roli, napříč odvětvími. To ostatně podporuje i řada studií a jejich výsledky naznačují, že přitažlivá vizuální prezentace produktu zvyšuje zákazníkův zájem v koupi daného produktu [27, 28, 29].

> *The results obtained show that there is a significant difference in the way customers view and rank products on the Internet depending on how it is presented. Respondents showed the greatest preference for a product presented using images, ... video, and music...*
> (Jovic, 2012) [28]

> *Získané výsledky naznačují, že je zde významná odchylka ve způsobu jak zákazníci nahlížejí a hodnotí produkt na internetu v závislosti na jeho prezentaci. Respondenti projevili největší preference v produktech prezentovaných obrázky, ... videi a hudbou...*
> (Přeloženo)



Obrázek 3.1: Produktové stránky e-shopů Alza.cz, Ebay.com, Ikea.cz a Zoot.cz

Nasnadě je pak otázka. Pokud lidé přikládají (audio)vizuální prezentaci produktů tak významnou roli, proč by tuto skutečnost nemohl využít i doporučovací systém a doporučovat produkty na základě jejich obrázků? Může takové řešení obstát alespoň v některých odvětvích? V této práci jsem se rozhodl možnosti takového doporučovacího modelu prozkoumat a zodpovědět tyto otázky.

V této práci navrhnu a implementuji několik vlastních *content-based* doporučovacích modelů, založených na modifikovaném algoritmu Item K-NN a podobnosti obrázků doporučovaných položek, abych zjistil, zda se může jednat o konkurenceschopnou alternativu k modelům kolaborativního filtrování. Samotné stanovení podobnosti dvojice obrázků pak provedu pomocí několika různých technik, určených ke zpracování obrazu.

3.1.1 Image K-NN

Navrhované doporučovací modely pro tuto práci postavím nad modifikovaným algoritmem Item K-NN. Originální funkci Sim, která určuje podobnost dvojice doporučovaných položek, zde navrhuji nahradit metodami pro zpracování obrazu a stanovení obsahové podobnosti obrázků dvojice doporučovaných položek. Tyto metody podrobně popíši v následujících kapitolách.

Nahrazením funkce Sim vlastním algoritmem, zpracovávajícím obrázky doporučovaných položek, se pokusím eliminovat dříve zmiňovaný *cold-start item* problém, kterým originální algoritmus Item K-NN trpí.

Výsledná utilitní funkce mého doporučovacího modelu F : Uživatelé $\times$ Položky $\rightarrow \mathbb{R}$ pro uživatele U a položku P pak tedy vypadá takto:

$$F(U,P) = \sum_{(U,P')\in I_U} Rating(U,P') \cdot \begin{cases} M(P,P') & P' \in Q_{P(K)} \\ 0 & \text{Jinak} \end{cases}$$

kde I_U opět označuje množinu známých interakcí uživatele U a funkce $Rating$: Uživatelé $\times$ Položky $\rightarrow \mathbb{R}$ známé hodnocení položky P' uživatelem U'. M : Položky $\times$ Položky $\rightarrow \mathbb{R}$ je v tomto případě proměnlivá funkce, zastupující jeden z algoritmů pro stanovení podobnosti dvojice položek P a P', určenou na základě obsahové podobnosti obrázku těchto položek.

Pro jasné budoucí odlišení originálního algoritmu Item K-NN od mnou navrhované mutace, budu nový algoritmus nazývat Image K-NN.

3.2 Podobnost obrázků

V předchozí kapitole jsem navrhl základní myšlenku CB doporučovacího modelu s využitím podobnosti obrázků. Pro stanovení této podobnosti použiji několik technik, které jsem teoreticky představil v předešlé kapitole. V této kapitole navrhnu nutné úpravy originálních metod a algoritmů tak, aby byly použitelné pro doporučovací modely.

3.2.1 Barevný histogram

První metodou ke zpracování obrázků, použitou k účelům této práce, je barevný histogram. Barevný histogram bude v této práci reprezentovat naivní techniku pro zpracování obrazu. Domnívám se ale, že i takto naivní přístup může mít zajímavé výsledky, je-li použit na obrázky ve správném odvětví. Vezměme si kupříkladu doporučovací systém určený pro e-shop s oblečením. Naivním předpokladem je, že různé typy zákazníků mají určité preference ohledně barvy svého oblečení a doporučení založená na barvě tedy mohou být relevantní. Tuto domněnku později ověřím na testovacích datech.

Základní barevný histogram sestavím ekvidistantní diskretizací RGB (Červená, Zelená, Modrá) prostoru do N distinktních segmentů pro každou složku. Každá ze složek R, G a B tak odpovídá jedné ose diskretizovaného třídimenzionálního prostoru a vznikne tak N^3 segmentů S_1 až S_N. Každý barevný histogram je pak možné efektivně kódovat vektorem (X_1, X_2, ..., X_{N^3}), kde hodnota každé složky tohoto vektoru odpovídá počtu pixelů náležících do příslušného segmentu diskrétního barevného histogramu.

Pro vektor histogramu obrázku I o rozměrech H a W potom platí:

$$X_i = \sum_{x=0}^{W} \sum_{y=0}^{H} \begin{cases} 1 & I_{xy} \in S_i \\ 0 & \text{Jinak} \end{cases}$$

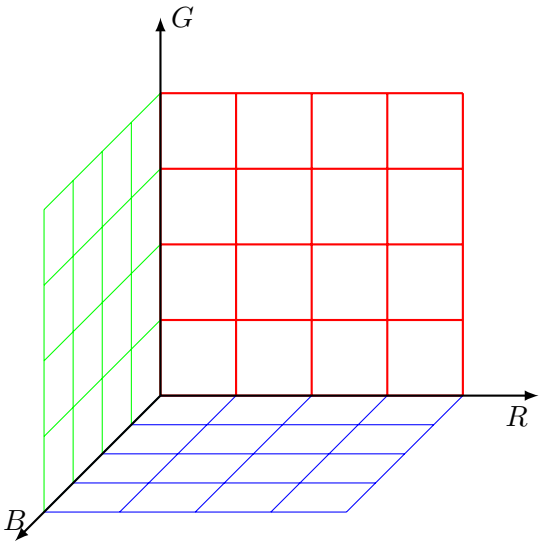
Pro zvýšení přesnosti extrahované informace může být vhodné některé složky výsledného vektoru zanedbat. Víme-li například, že všechny obrázky v datasetu obsahují bílé pozadí, je vhodné z výsledného vektoru odpovídající složku vynechat. Zvýší se tak poměr počtu pixelů, zaznamenaných ve vektoru histogramu, ve prospěch důležitých pixelů reprezentující objekty na obrázku, na úkor nepodstatných pixelů šumu, které mohou reprezentovat objekty v pozadí nebo pozadí samotné.

V druhém kroku je potřeba stanovit podobnost objektů na různých obrázcích pomocí sestavených vektorů histogramů. Pro tento úkon navrhuji kosinovou vektorovou podobnost. Vzhledem k tomu, že je citlivá pouze na vzájemnou orientaci porovnávaných vektorů a plně abstrahuje jejich velikosti, dobře tak abstrahuje celkové množství pixelů porovnávaných obrázků a porovnává tak pouze skutečnou barevnou distribuci pixelů na obrázku.

Pro dva obrázky I_1 a I_2 s vektory barevného histogramu X_1 a X_2 respektive pak platí, že jejich podobnost S_{12} je rovna:

$$S_{12} = \frac{1}{2} \cdot \left(1 - \frac{X_1 \cdot X_2}{|X_1| \cdot |X_2|}\right) = \frac{1}{2} \cdot \left(1 - \frac{\sum_{i=0}^{N^3} X_{1_i} \cdot X_{2_i}}{\sqrt{\sum_{i=0}^{N^3} (X_{1_i})^2} \cdot \sqrt{\sum_{i=0}^{N^3} (X_{2_i})^2}}\right)$$

U podobnosti založené na barevném histogramu se dá předpokládat, že nebude záležet na pozadí obrázku. Ze vzorce je skutečně vidět, že X_1 a X_2



Obrázek 3.2: Ilustrace diskretizovaného RGB prostoru pro N=4

jsou vůči sobě pozičně invariantní a je tedy možné I_1 a I_2 zaměnit. Z toho nutně platí:

$$S_{12} = S_{21}$$

Z toho vyplývá, že známe-li pro dvojici obrázků I_x a I_y jejich obsahovou podobnost S_{xy}, stanovenou podle algoritmu postaveném nad barevným histogramem výše, známe automaticky i S_{yx}. Vzhledem k tomu, že podobnost je nutné stanovit pro každou dvojici obrázků a předpočítání podobností má tím pádem netriviální složitost $O(n^2)$, kde n je počet obrázků v datasetu, je toto příjemná vlastnost, která redukuje výpočetní náročnost úkonu na polovinu.

3.2.2 Oriented FAST and Rotated BRIEF

Algoritmus ORB byl původně navržen k identifikaci dvojic nejpodobnějších bodů na různých obrázcích s různým obsahem. Vzhledem k tomu, že výstupem tohoto algoritmu je mimo vektoru párů nejpodobnějších bodů i jejich vzdálenost, je tento algoritmus velice jednoduše upravitelný na variantu určující podobnost dvou různých obrázků.

Algoritmus ORB formálně v prvním kroku vrátí pro každý obrázek vektor bodů $(P_1, P_2, ..., P_N)$. N je v tomto případě proměnlivé a záleží na počtu významných bodů, které algoritmus na obrázku najde. V druhém kroku následuje párování bodů obrázků, kde algoritmus vrátí vektor i trojic {bod prvního obrázku, bod druhého obrázku, obsahová vzdálenost bodů}. Proměnná i může nabývat hodnot z diskrétního intervalu [0, N].

$$I = (\{P1_a, P2_b, D_1\}, \{P1_c, P2_d, D_2\}, ..., \{P1_y, P2_z, D_i\})$$

Zde navrhuji rozšíření algoritmu o třetí krok, a to o stanovení podobnosti obrázků na základě nalezených dvojic bodů a jejich vzdáleností algoritmem ORB. Tu definuji jako součet čtverců vzdáleností prvních K detekovaných párů bodů.

Formálně tak pro pár obrázků s vektorem párovaných bodů a jejich vzdáleností X platí:

$$S_{12} = \sum_{i=0}^{K} \begin{cases} (X_i.D)^2 & |X| < i \\ 1 & \text{Jinak} \end{cases}$$

K je v tomto případě umělá konstanta stanovující, kolik nejbližších bodů je bráno v potaz při stanovování podobnosti obrázků. Hodnota tohoto parametru může značně ovlivnit výsledky algoritmu napříč různými doménami nebo datasety, zejména pak v závislosti na kvalitě obrázků v datasetu. Proto je nutné ji dobře zvolit a otestovat nebo zkusit několik variant na cílovém datasetu a vybrat nejlepší na základě výsledku testu ještě před nasazením algoritmu.

Za povšimnutí stojí penalizace podobnosti obrázků v případě, kdy algoritmus nedokáže detekovat dostatečné množství párů bodů z obou obrázků, tedy alespoň K. Vzhledem k tomu, že vzdálenosti bodů jsou vždy z intervalu $\langle 0, 1 \rangle$, nenalezení dostatečného množství párů bodů je hodnoceno stejně, jako nalezení naprosto nepodobných bodů. Tím se zachová konzistence podobností, vrácených tímto algoritmem.

Přihlédneme-li k faktu, že všechny kroky navrhovaného algoritmu jsou deterministické a vstupy všech kroků mají zaměnitelné pořadí bez změny výsledku, platí i pro celý model, že pro obrázky O_1 a O_2 vrátí vždy stejnou podobnost, nezáleže na vstupních pozicích obrázků. I pro tento model tedy platí:

$$S_{12} = S_{21}$$

3.2.3 Umělá neuronová síť

Poslední navrhovaný model, určený k predikci obsahové podobnosti doporučovaných položek, založím na umělé neuronové síti. Umělá neuronová síť je abstraktní učící algoritmus. K problému predikce obsahové podobnosti dvojice obrázků doporučovaných položek pomocí neuronové sítě tak lze přistupovat různě, od návrhu vlastní architektury umělé neuronové sítě na různých principech *self-supervising autoencoderů*, přes učenou klasifikaci založenou na velkém množství trénovacích dat, až po použití předtrénované neuronové sítě a případným upravením dle potřeby.

Návrh a trénování vlastní neuronové sítě není jednoduchý proces a z důvodu nároků na kvalitu trénovacího datasetu, nároků na výpočetní výkon a zejména z nutnosti vhodného návrhu architektury sítě, jsem se rozhodl ve své práci využít předtrénovanou síť. Oblast výzkumu rozpoznávání a zpra-

cování obrazu pomocí umělých neuronových sítí je momentálně významně zkoumaným odvětvím a je proto z čeho vybírat.

Nejvýraznějšími hráči v této oblasti jsou předtrénované sítě ResNet, InceptionNet a rodina sítí VGG Net [30]. Pro tuto práci jsem se rozhodl zvolit kompromis mezi velikostí sítě, tudíž výpočetní náročností, a kvalitou zpracování obrazu a zvolil jsem VGG Net D, jinak známou jako VGG16 [26]. Jedná se o neuronovou síť navrženou a trénovanou za účelem klasifikace objektů na obrázcích. Na vstupu očekává obrázek ve standardizovaném formátu a na výstupu klasifikuje objekty na předloženém obrázku do tisíce tříd typů objektů.

Architekturu sítě VGG16 můžeme rozdělit do třech hlavních částí. První částí je vstupní vrstva o $224 \times 224 \times 3$ neuronech. Počet neuronů jsem zde záměrně uvedl v tomto tvaru. Je z něho patrné, že je síť schopná zpracovat obrázek o rozměrech 224×224 pixelů a zároveň rozlišit každou ze složek RGB barevného prostoru a rozeznávat tak barvy.

Za vstupní vrstvou následuje kaskáda pěti konvolučních vrstev. Ty mají za úkol získat co nejvíce informací o obsahu zpracovávaného obrázku, nezávisle na umístění, a redukovat tak dimenzi dále zpracovávané informace.

Poslední částí sítě VGG16 je třívrstvá sekce hustě spojených neuronových vrstev. Ta je určena k vyhodnocení extrahovaných informací z konvolučních vrstev a klasifikaci do předem specifikovaných kategorií.

Vzhledem k tomu, že mým cílem není exaktní klasifikace objektů na obrázku, ale spíše relativní porovnání obsahu obrázků, navrhuji zde upravit předtrénovanou síť. Do svého modelu použiji pouze první dvě předtrénované části sítě, tedy vstupní vrstvu a kaskádu konvolučních vrstev. Výstupem této sítě bude vektor obsahující informaci o skutečném obsahu obrázku. Takto extrahované vektory pro obrázky doporučovaných položek porovnám vektorovou kosinovou podobností a prohlásím takto získanou obsahovou podobnost obrázků položek za podobnost doporučovaných položek.

Kapitola 4

Implementace

V předchozích kapitolách jsem teoreticky popsal doporučovací model Image K-NN, který navrhuji jako modifikaci originálního algoritmu Item K-NN, adresující *cold-start item* problém. Taktéž jsem představil algoritmy pro zpracování obrazu a navrhl jejich modifikace tak, abych je mohl použít ve svých doporučovacích modelech.

V této a následující kapitole se budu tématy práce zabývat především prakticky. Popsané metody pro zpracování obrazu implementuji a následně nasadím do doporučovacího modelu. V další části práce pak všechny implementované doporučovací modely podrobím testování z hlediska relevance doporučení a pokrytí datasetu pomocí metod *recall* a *catalog-coverage*.

4.1 Podobnost obrázků

V této kapitole popíši některé zajímavé detaily implementace algoritmů pro zpracování obrázků a stanovení obsahové podobnosti dvojice obrázků.

4.1.1 Barevný histogram

Barevný histogram, zástupce naivních algoritmů pro stanovení obsahové podobnosti dvojice obrázků, jsem implementoval přesně podle návrhu v předchozí kapitole. Při testech se ukázalo být dobrým počtem segmentů pro rozdělení barevného RGB prostoru 27. Tento počet reprezentoval dobrý kompromis, mezi přiměřenou generalizací a přílišným roztříštěním barevného prostoru.

Celá implementace modelu je k nahlédnutí v příloze na CD, v souborech *ColorHistogra.py* a *ColorHistogra.cs*.

Datasety, které mám k dispozici, a na kterých bude model testován, mají bílé pozadí. Této znalosti jsem při implementaci algoritmu využil a doplnil ho o dříve zmiňovanou funkcionalitu pro zanedbání bílých pixelů pozadí.

Obrázek 4.1: Výsledky barevného histogramu

Jak je vidět z obrázku, model založený na barevném histogramu je dle očekávání skutečně zaujatý především na barevnou dispozici cílových obrázků, méně pak na samotný tvar nebo jiný obsah objektů na obrázku.

4.1.2 ORB

ORB je poměrně komplexní a na implementaci ne právě nejjednodušší algoritmus. Vlastní implementací tohoto algoritmu bych strávil spoustu času činností, která není primární náplní této práce. Podle známého úsloví jsem zde nevymýšlel kolo a použil jsem již implementovanou verzi algoritmu, dostupnou v knihovně OpenCV.

OpenCV (https://OpenCV.org) je velice známá multiplatformní open-source knihovna, sdružující různé techniky a algoritmy pro zpracování obrazu. Disponuje připravenými rozhraními do nejrůznějších programovacích i skriptovacích jazyků. Její backend je napsaný v jazyce C s podporou hardwarové akcelerace na GPU, takže je i velice rychlá. Pro doporučovací model založený na algoritmu ORB jsem tedy použil připravenou implementaci algoritmu ORB z knihovny OpenCV a doprogramoval pouze samotné stanovení obsahové podobnosti dvojic obrázků podle návrhové části práce.

Celá implementace modelu je k nahlédnutí v příloze na CD, v souborech *ORB_Extract.py*, *ORB_ComputeSimilarity.py* a *ORB.cs*.

Obrázek 4.2: Výsledky algoritmu ORB

Algoritmus ORB je, oproti předcházejícímu algoritmu o poznání sofistikovanější. Zatímco k barevné distribuci pixelů na obrázku je téměř necitlivý, na rozdíl od barevného histogramu má poměrně dobrý cit k hranám a tvarům a tím i k typu objektů na obrázku. To je ostatně velice dobře vidět na obrázku 4.2. Je zde vidět, že algoritmus v datasetu velice dobře identifikoval podobné nádoby a židle, nehledě na jejich barvu nebo materiál.

4.1.3 Umělá neuronová síť

Jak už jsem uvedl v návrhové části práce, pro model založený na umělé neuronové síti použiji modifikovanou variantu předtrénované neuronové sítě VGG16. Pro urychlení implementace jsem použil hotovou knihovnu Keras (https://Keras.io), určenou k návrhu a trénování umělých neuronových sítí.

Samotnou podobnost vektorů, extrahovaných z obrázků pomocí neuronové sítě dle návrhu spočítám pomocí kosinové podobnosti. Při velikosti jednotlivých vektorů, přesahující 25 tisíc elementů, se ale jedná o poměrně velké množství výpočetních operací a jejich sekvenční výpočet na CPU by tak zabral nezanedbatelnou dobu. Vzhledem k tomu, že se ale jedná o přímočarou matematickou operaci, lze tento výpočet velice dobře akcelerovat na GPU, a dosáhnout tak mnohem kratšího výpočetního času.

Napsal jsem proto program, který je schopný provést akcelerovaný výpočet na GPU. Výsledkem bylo dvacetinásobné urychlení oproti sekvenčnímu průchodu na CPU a celková operace pro oba evaluační datasety, které představím v další části práce, tak trvala jen několik desítek minut.

Celá implementace modelu je k nahlédnutí v příloze na CD, v souborech *VGG16_VectorExtract.py* a *VGG16.cs*.

Obrázek 4.3: Výsledky umělé neuronové sítě

Podíváme-li se namátkou na výsledky, zatímco předešlé modely byly citlivé buď na barvu nebo na tvary na obrázku, neuronová síť kombinuje obě tyto dovednosti. Zatímco u příkladu červených šatů velice dobře detekovala důležitost právě červené barvy, u obrázků s nábytkem se pak správně zaměřovala především na tvar zobrazených objektů.

4.2 Doporučovací model

Jak už jsem uvedl v návrhové části práce, doporučovací model Image K-NN navrhuji se záměrem vyřešení *cold-start user* a *item* problémů. V souladu s návrhem, uvedeným v předchozí kapitole práce jsem implementoval modulární doporučovací model Image K-NN, který podporuje různé metody pro stanovení obsahové podobnosti doporučovaných položek.

Minimální implementace tohoto modelu je znázorněna na útržku kódu 4.4.

Celá implementace doporučovacího modelu je k nahlédnutí v příloze na CD v souboru *Recommender.cs*.

```
//Minimal recommender implementation
//Omitts lot of (sometimes important) implementation details!
class Recommender
{
  Recommender
  (
    ItemKnnModel itemKnns,
    InteractionData interactions
  )
  {
    //Accepts known user-item interactions...
    this.Interactions = interactions;

    //... and invariant model for image processing
    this.ItemKnns = itemKnns;
  }

  List<(string, float)>> RecommendForUser(string user, int K)
  {
    var knnAccumulator = ArrayOf<(string, float)>();
    var userInteractions =
       this.Interactions.GetInteractionsForUser(user);

    foreach(var (currentItem, itemRating) in userInteractions)
    {
      foreach(var (item, simlarity) in
         this.ItemKnns.ForItem(currentItem).Take(K)
      )
      {
        //Sum of item ratings, weighted by similarity score
        var value = itemRating * simlarity;
        knnAccumulator.Increment(item, value);
      }
    }

    return knnAccumulator.OrderBy(x => x.Value).ToList();
  }
}
```

Obrázek 4.4: Minimální implementace doporučovacího modelu Image K-NN

Kapitola 5

Experimenty

V předchozích kapitolách jsem navrhl a implementoval navržené *content-based* doporučovací modely, založené na zpracování obrázků doporučovaných položek. V této kapitole provedu evaluaci doporučení těchto modelů a na základě výsledků testů se pokusím zodpovědět několik dříve položených otázek ohledně kvality doporučení navrhovaných modelů.

Cílem této kapitoly je zjistit, zda může být CB doporučovací model založený na zpracování obrázků schopný cílovým uživatelům doporučovat relevantní položky. V případě, že výsledky testů podle dříve zmíněných metod offline evaluace ukáží, že doporučení těchto modelů jsou skutečně relevantní, druhotnou otázkou bude, jak moc se úspěšnost těchto systémů liší a nakolik záleží na kvalitě algoritmu, určeného ke zpracovaní obrázků doporučovaných položek a stanovení jejich podobnosti.

Model, který se podle offline testů ukáže být nejlepší, bude v druhé části testování nasazen do online A/B testu, a testován proti produkčnímu algoritmu založeném na CF. Online A/B test nám poskytne informace o tom, nakolik jsou doporučení modelu založeném na obrázcích skutečně relevantní pro cílové uživatele. Zároveň tím získám více podkladů k dalšímu vylepšení modelu.

5.1 Datasety

Pro offline evaluaci modelů použiji datasety s reálnými daty dvou online e-shopů.

Prvním z nich je dataset e-shopu se zaměřením na domácí nábytek a domácí dekorativní doplňky. Tento dataset budu dále nazývat jen jako *Nábytek*. Dataset obsahuje přes **10 tisíc produktů** s obrázky, téměř **500 tisíc uživatelů** a přes **2 miliony** zaznamenaných uživatelských **interakcí**.

Druhým datasetem, na kterém budu modely testovat, je z online obchodu se zaměřením na prodej nejrůznějšího oblečení, především dámských šatů.

Tento dataset obsahuje přes **110 tisíc položek**, více než **4,5 milionů uživatelů** a téměř **50 milionů** zaznamenaných uživatelských **interakcí**. Tento dataset budu dále nazývat jako dataset *Oblečení*.

5.2 Předzpracování dat

Před použitím dat z výše uvedených datasetů je nutné tyto data nejprve předzpracovat. Existuje k tomu několik důvodů.

Hlavní motivací k předzpracování datasetu je zvýšení kvality obsažených dat. V datasetu se surovými daty, tak jak byla nasbírána, se vyskytuje spousta anomálií a šumu, který by zkreslil výsledky prováděných testů. Jedním z příkladů jsou takzvaní „crawleři". Jedná se o virtuální robotické „uživatele", kteří mají za úkol projít co největší množství stránek a produktů na cílovém portálu a naindexovat jeho obsah. Neblahým vedlejším účinkem jejich práce je velké množství zaznamenaných uživatelských interakcí, které nejsou od skutečných uživatelů, a jsou tak pro testovací účely zavádějící. Tyto interakce a jiný šum je nutné před testem z datasetu odfiltrovat a testovat jen na opravdu validních datech.

Toto odfiltrování provedu sestavením blacklistu všech uživatelů, pro které dataset obsahuje více než 3 tisíce zaznamenaných interakcí. Dá se předpokládat, že v takto definované skupině bude většina odfiltrovaných uživatelů roboti a jen malé procento odfiltrovaných budou „lidští" uživatelé. Tyto uživatele a všechny jejich interakce následně z testovacích datasetů odstraním.

Sekundárním důvodem pro předzpracování dat je výkonnostní náročnost testů. Řadu výpočtů je možné provést předem a vyhnout se tak opakovanému počítání stejných dat v průběhu testů. Namísto toho se testovací program může odkazovat na hotové výsledky a testování se tak celkově urychlí. Tím pádem si budu moci dovolit provést větší množství testů a mé závěry tak budou podloženy větším množstvím dat.

Prvním takovým je před-výpočtem jak takzvaná interakční matice uživatelů a doporučovaných položek. Každý záznam v této matici označuje hodnotu zaujetí uživatele v jedné položce. Matice má rozměry $|U| \times |P|$, kde U označuje množinu uživatelů v systému a P množinu doporučovaných položek. Pro každý záznam v interakční matici, v případě mých datasetů, platí:

$$IM_{up} = \begin{cases} 0,75 & (u,p) \in P \\ 0 & \text{Jinak} \end{cases} + \begin{cases} 0,25 & (u,p) \in V \\ 0 & \text{Jinak} \end{cases} + \begin{cases} 0,75 & (u,p) \in C \\ 0 & \text{Jinak} \end{cases} + \begin{cases} Rat(u,p) & (u,p) \in H \\ 0 & \text{Jinak} \end{cases}$$

P, V, C a H v tomto případě značí podmnožiny zaznamenaných interakcí, nákup, zobrazení detailu, přidání do košíku, a explicitní ohodnocení položky

$$\mathbf{IM} = \underbrace{\left.\begin{pmatrix} 1 & 0 & 0,75 & \cdots & 0 \\ 0 & 0,25 & 0 & \cdots & 1 \\ 0 & 0 & 1 & \cdots & 0,25 \\ \vdots & & & \ddots & \\ 0 & 0,5 & 0 & \cdots & 0,75 \end{pmatrix}\right\} \text{Uživatelé}}_{\text{Položky}}$$

Obrázek 5.1: Interakční matice uživatelů a položek

respektive. Funkce $Rat : \text{Uživatelé} \times \text{Položky} \rightarrow \mathbb{R}$ označuje funkci, která vrací uživatelovo normalizované explicitní hodnocení, jaké dané položce zanechal.

Pro lepší představu je naivní vizualizace interakční matice zachycena na obrázku 5.1.

Druhým důležitým předzpracováním je stanovení podobností doporučovaných položek pro algoritmus Image K-NN. V mém případě to znamená zpracování obrázků doporučovaných položek všemi navrhovanými metodami pro stanovení obsahové podobnosti dvojic obrázků. Tento výpočet se rozhodně nedá považovat za výpočetně zanedbatelnou operaci a vyplatí se jej provést napřed, ještě před samotným testováním modelů.

5.3 Offline testy

V této kapitole se budu věnovat offline testování navrhovaných modelů na výše představených datasetech. Modely budu testovat pomocí dvojice metod. V první řadě modely otestuji na doporučovací *recall*. Ten napoví skutečnou relevanci doporučení jednotlivých modelů. Sekundárně pak doporučení jednotlivých modelů ověřím z hlediska *catalog coverage*. Je nejen důležité, aby doporučovací model doporučoval relevantní položky, ale je také cennou vlastností, pokud model doporučuje položky rozmanité, tak aby uživatelé objevovali nové položky.

Podle metodiky pro měření hodnoty *recall*, popsané v úvodu práce, jsem pro každého uživatele v datasetu nejprve stanovil jeho partikulární *recall*. Metodou *leave-one-out* jsem postupně otestoval každou položku, kterou uživatel hodnotil kladně, a testoval její přítomnost na prvním doporučeném místě, mezi prvními pěti, deseti, patnácti a dvaceti doporučeními. Výsledný *recall* pro celý testovací dataset jsem pak spočítal jako průměr partikulárních hodnot *recall* všech uživatelů v datasetu.

Tento proces jsem opakoval pro každý navrhovaný model a sadu hodnot parametru K algortimu Image K-NN. V tabulkách 5.1, 5.2, 5.3, 5.4 a 5.5 jsou podrobně zaznamenány všechny naměřené hodnoty *recallu* pro jednotlivé modely, napříč testovacími datasety a parametry jednotlivých modelů. Zde

bych zdůraznil, že směrodatné jsou hlavně řádky s daty pro prvních 5 a prvních 10 doporučení, potažmo pouze první doporučení. Toto jsou skutečně položky, které by uživatel ve většině případů viděl a měl by šanci vyhodnotit jejich relevanci a případně toto doporučení i využít. Řádky s 15 a 20 doporučeními jsou pak spíše orientační, pro dokreslení celkového obrazu o chování modelů.

Při testovaní modelů na míru pokrytí datasetu pomocí metody *catalog coverage* jsem postupoval obdobně. Uživatele v testovacím datasetu jsem nejprve náhodně rozdělil na 10 skupin a pro každou z nich spočítal její partikulární *catalog coverage*. Výsledný *catalog coverage* pro celý dataset jsem pak stanovil jako průměr naměřených výsledků jednotlivých skupin. Měření jsem opět provedl na obou datasetech, všech navrhovaných modelech a sadě hodnot parametru K. Všechny naměřené hodnoty jsou zaznamenány v tabulce 5.6.

Pro větší přehlednost jsou důležitá data z tabulek 5.1, 5.2, 5.3, 5.4, 5.5 a 5.6 vyobrazena v následujících grafech 5.2 a 5.3.

Musím zde uvést, že model založený na algoritmu ORB nebyl testován na datasetu *Oblečení* a vyhodnocení úspěšnosti tohoto modelu bude založeno pouze na datasetu *Nábytek*. Důvodem je jeho výpočetní náročnost. Výpočet podobností obrázků je nutné provést pro každou dvojici obrázků, což dává výpočetní složitost $O(n^2)$. Pro dataset *Nábytek*, který měl pouze 10 tisíc obrázků, trval výpočet 1×10^8 dvojic obrázků 30 hodin. Při optimistickém odhadu průměrného času jedné milisekundy na dvojici by tedy výpočet $1,44 \times 10^{10}$ dvojic obrázků pro dataset *Oblečení* zabral necelých 167 dní.

Tento fakt bohužel, v případě mnou dostupných výpočetních prostředků, diskvalifikuje model založený na algoritmu ORB pro velké datasety a mohu se tak pouze domnívat, že se tento model bude chovat na velkých datasetech stejně jako na těch malých, které mohu skutečně otestovat. Pro velké společnosti zabývající se doporučováním ale nemusí být výpočetní výkon problém, a tak může být model založený na algoritmu ORB stále validním modelem, použitelným i na velké datasety.

Za předpokladu, že by se býval model založený na algoritmu ORB choval ve srovnání s ostatními testovanými modely porovnatelně na datasetu *Oblečení* jako na datasetu *Nábytku*, můžeme z naměřených dat vyvodit několik závěrů. V první řadě je zřejmé, že model založený na neuronové síti dominuje ostatní modely napříč proměnlivým množstvím doporučených položek i parametry doporučovacího algoritmu Image K-NN. To je důležité pozorování, které ukazuje, že plasticita modelu pro stanovení obsahové podobnosti položek je skutečně významným faktorem, hrajícím roli ve výsledné kvalitě a relevanci doporučení modelu. Neuronová síť je učený model bez předkódovaných pravidel, a je tak mnohem plastičtější než ostatní testované modely. 2. místo v tomto srovnání obsadil model, založený na algoritmu ORB. Nejhůře z testovaných modelů dopadl model zastupující naivní algoritmy, tedy barevný histogram. Na rozdíl od umělé neuronové sítě a algoritmu ORB, *recall* modelu založeném na barevném histogramu neroste ani s vyšším počtem doporučení a jeho doporučení tedy nejsou příliš relevantní.

Zajímavým pozorováním je, že model založený na neuronové síti jako jediný benefituje z vyššího parametru K algoritmu Image K-NN. To lze vysvětlit fenoménem, o kterém jsem se již zmiňoval v první části práce, tedy že z vyšších hodnot parametru K benefitují především položky takzvaně „podobné všem“. Toto chování dobře demonstroval model založený na barevném histogramu, kde jsem při postupném zvyšování hodnoty parametru K pozoroval nárůst výskytu položek s obrázky, na kterých dominovaly různé odstíny šedé. Histogramy takových položek nejsou nikterak vyhraněné a mají blízko ke všem položkám, které už cílový uživatel viděl. Obrázkový prostor neuronové sítě se zdá být lépe rozčleněný a proto je při malé hodnotě parametru K imunní.

Dalším zajímavým poznatkem je, že všechny modely zaznamenaly horší výsledky nad datasetem *Oblečení*, oproti datasetu *Nábytku*. To může mít hned několik vysvětlení. Jako první a nejpravděpodobnější vysvětlení se nabízí možnost, že je dataset jednoduše příliš velký a obsahuje příliš mnoho podobných alternativ. Doporučení modelů tak sice mohla být správná a modely mohly doporučovat blízké alternativy, nejednalo se ale přesně o ty položky, které testovací algoritmus očekával, a hodnotil tak modely nižším *recall* skóre.

Alternativním vysvětlením může být změna domény. Jak už jsem uvedl v úvodu práce, CB doporučovací modely nejsou tak snadno přenositelné mezi doménami, jako modely založené na CF, a doména oblečení nemusí být tak vhodná pro obrázkové doporučování, jako doména s domácím nábytkem a bytovými doplňky.

Třetím možným vysvětlením tohoto chování je externí ovlivnění testovacích dat. V závislosti na možných promo akcích, které na portálech e-shopů pravděpodobně probíhaly v době sběru testovacích dat, a doporučovacím systému, který na e-shopu v danou dobu běžel, je možné, že chování a preference uživatelů byly do jisté míry ovlivněny právě těmito faktory a uživatelé se tak na webu nechovali naprosto přirozeně, čímž ovlivnili výsledky mých testů.

V každém případě je důležité, že se modely vůči sobě chovaly konzistentně napříč datasety a víme, který model nasadit do online testu.

Dop. pol.	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.014 359 304	0.007 519 365	0.002 860 736	0.000 295 083	0.006 671 352
5	0.053 227 566	0.035 812 836	0.009 166 274	0.000 969 475	0.023 221 353
10	0.088 490 108	0.062 201 138	0.015 281 097	0.001 634 25	0.037 323 987
15	0.109 792 656	0.077 506 074	0.019 166 93	0.002 092 685	0.045 145 215
20	0.123 063 292	0.087 055 766	0.021 676 034	0.002 364 015	0.049 659 474

Tabulka 5.1: Recall pro K=1. Datasety Nábytek (N) a Oblečení (O)

Dop. pol.	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.019 302 02	0.005 902 254	0.003 903 712	0.000 436 286	0.006 903 775
5	0.055 883 679	0.029 195 398	0.009 104 689	0.000 874 028	0.020 253 62
10	0.090 784 656	0.053 487 813	0.014 047 405	0.001 418 584	0.032 623 497
15	0.118 958 931	0.073 157 358	0.018 247 124	0.001 909 806	0.042 697 848
20	0.142 240 155	0.089 805 251	0.021 932 308	0.002 361 247	0.051 446 73

Tabulka 5.2: Recall pro K=5. Datasety Nábytek (N) a Oblečení (O)

Dop. pol.	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.019 649 679	0.005 099 659	0.003 287 86	0.000 474 464	0.006 129 857
5	0.053 273 258	0.024 336 121	0.008 904 04	0.000 979 675	0.018 916 639
10	0.081 723 673	0.044 349 351	0.012 950 789	0.001 331 735	0.029 153 03
15	0.108 739 747	0.061 625 017	0.017 186 268	0.001 909 806	0.038 721 296
20	0.129 720 462	0.075 861 151	0.020 192 027	0.002 226 164	0.046 270 319

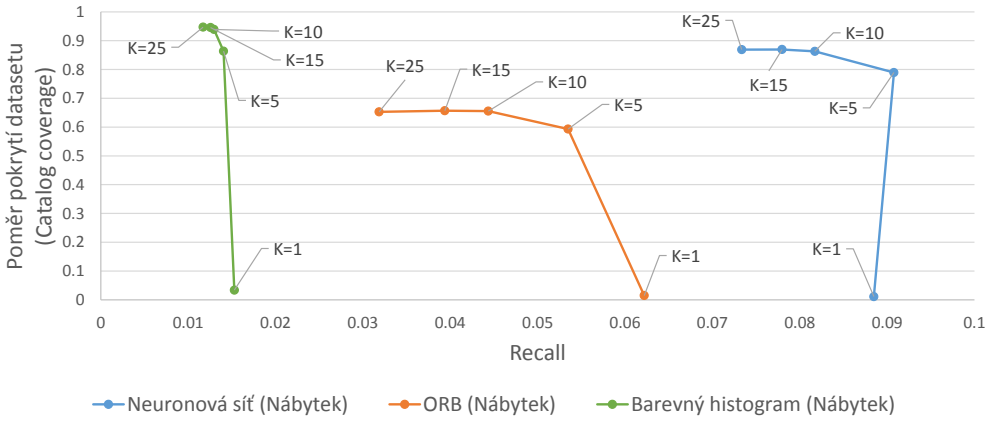
Tabulka 5.3: Recall pro K=10. Datasety Nábytek (N) a Oblečení (O)

Dop. pol.	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.018 853 044	0.004 366 595	0.003 095 157	0.000 506 377	0.005 628 726
5	0.052 965 331	0.021 652 194	0.008 421 291	0.001 009 402	0.017 679 914
10	0.077 976 904	0.039 376 836	0.012 557 438	0.001 439 276	0.027 764 173
15	0.098 999 338	0.053 603 037	0.015 821 458	0.001 750 68	0.035 713 198
20	0.120 407 178	0.066 323 378	0.019 707 291	0.002 314 762	0.043 717 305

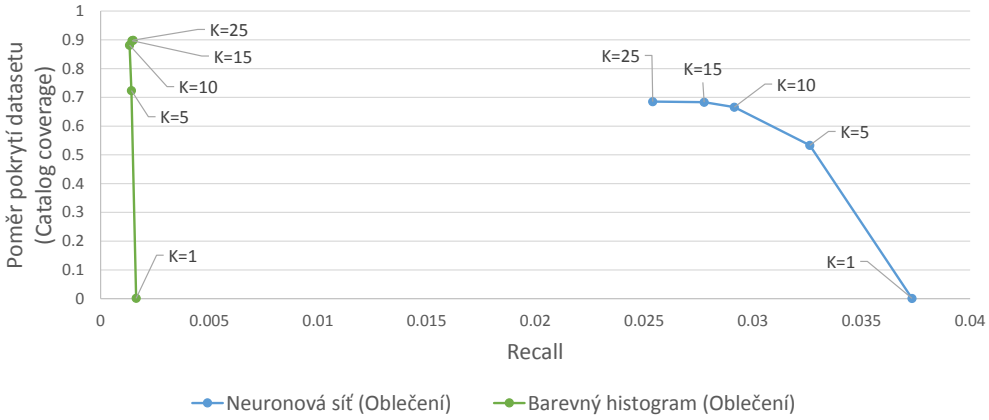
Tabulka 5.4: Recall pro K=15. Datasety Nábytek (N) a Oblečení (O)

Dop. pol.	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.018 133 886	0.003 351 431	0.002 719 686	0.000 501 568	0.004 964 097
5	0.049 828 455	0.017 023 365	0.007 745 84	0.001 042 772	0.015 993 788
10	0.073 356 021	0.031 853 499	0.011 701 204	0.001 492 027	0.025 398 118
15	0.090 438 984	0.043 594 435	0.014 814 241	0.001 871 336	0.033 121 86
20	0.105 205 546	0.053 738 127	0.017 601 472	0.002 200 226	0.039 741 773

Tabulka 5.5: Recall pro K=25. Datasety Nábytek (N) a Oblečení (O)



Obrázek 5.2: Recall a Catalog coverage pro různá K při top 10 doporučení



Obrázek 5.3: Recall a Catalog coverage pro různá K při top 10 doporučení

5.4 Online testy

Doporučovací model Image K-NN, založený na umělé neuronové síti, byl na základě výsledků offline testů zvolen jako nejlepší z navrhovaných modelů k online A/B testu. V něm se utkal s produkčním algoritmem User K-NN, založeným na kolaborativním filtrování.

Test probíhal ve dnech od 20. 3. 2018 do 26. 3. 2018 a zúčastnilo se ho celkem **7435 unikátních uživatelů**. Ti byli rozděleni do testovacích skupin v poměru 2:1 ve prospěch modelu User K-NN.

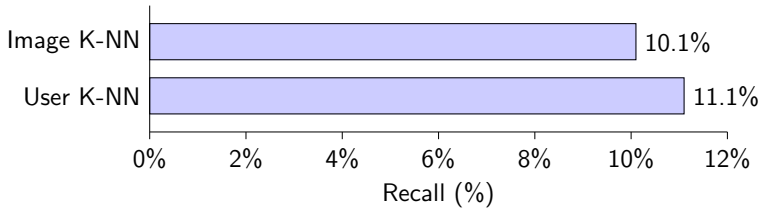
Dle metodiky, uvedené v úvodu práce, byly oba testované doporučovací modely měřeny na proklikovost jednotlivých doporučení. Výsledná proklikovost každého z modelů byla v tomto testu stanovena, jako průměr proklikovostí

K	NN (N)	ORB (N)	Histogram (N)	Histogram (O)	NN (O)
1	0.011 500 432	0.014 917 923	0.034 011 712	0.001 452 714	0.001 222 111
2	0.527 320 726	0.392 387 444	0.537 957 185	0.332 407 592	0.283 325 008
3	0.685 475 665	0.504 694 25	0.730 469 425	0.526 486 846	0.408 752 914
4	0.750 983 969	0.560 449 266	0.816 799 462	0.646 367 799	0.482 887 113
5	0.789 737 928	0.593 117 02	0.863 741 96	0.723 569 764	0.533 305 028
6	0.814 101 949	0.615 244 312	0.892 675 434	0.777 201 132	0.572 826 34
7	0.830 642 219	0.629 691 85	0.907 958 145	0.814 088 412	0.601 237 929
8	0.842 747 432	0.639 761 928	0.919 410 579	0.842 405 927	0.625 923 243
9	0.854 401 459	0.648 017 663	0.930 037 439	0.864 164 169	0.647 111 222
10	0.863 137 18	0.655 303 83	0.938 974 753	0.881 623 377	0.665 878 288
11	0.869 770 567	0.660 612 46	0.944 465 777	0.895 410 423	0.682 218 615
12	0.869 616 972	0.659 854 085	0.944 792 167	0.896 020 646	0.682 310 19
13	0.869 415 379	0.658 788 519	0.945 051 358	0.896 371 961	0.682 514 985
14	0.869 242 584	0.657 857 349	0.945 396 947	0.896 684 149	0.682 832 168
15	0.869 357 781	0.656 858 981	0.945 598 541	0.897 085 415	0.683 182 651
16	0.869 386 58	0.655 879 812	0.945 905 731	0.897 452 547	0.683 611 389
17	0.869 348 181	0.655 063 838	0.946 040 127	0.897 778 888	0.684 025 974
18	0.869 290 583	0.654 276 663	0.946 318 518	0.898 030 303	0.684 449 717
19	0.869 223 385	0.653 335 893	0.946 539 311	0.898 305 028	0.684 877 622
20	0.869 021 791	0.652 683 114	0.946 865 7	0.898 623 044	0.685 225 608

Tabulka 5.6: Pokrytí datasetu (Catalog coverage) pro různá K

všech uživatelů v testovací skupině. Tím se z výsledků testu eliminovaly rušivé faktory, jako jsou například dříve zmiňovaní crawleři.

Doporučovací model Image K-NN prokázal průměrnou proklikovost 10,1 % (10,11268642 %) zatím co produkční doporučovací model User-K-NN měl proklikovost 11,1 % (11,10827784 %). Z toho vyplývá, že model založený na kolaborativním filtrování měl lepší proklikovost o celých 8,96 % a obstál tedy proti mnou navrženému *content-based* doporučovacímu modelu. Přihlédneme-li ale k faktu, že proti sobě stály na jedné straně vyladěný model s dlouholetou praxí proti novému nevyladěnému modelu, jedná se o překvapivě dobrý výsledek, který zavdává podnět k pokračování v této práci a snahám o další vylepšení.



Obrázek 5.4: Porovnání modelů v online A/B testu

Závěr

Množství informací, které nás každý den obklopuje se stále zvětšuje. Není proto v lidských silách je všechny zpracovat manuálně a musíme naučit stroj, aby pro nás vybral přesně ty, které jsou relevantní právě pro nás.

Cílem této práce bylo navrhnout a porovnat několik *content-based* doporučovacích modelů jako alternativ k doporučovacím modelům založeným na kolaborativním filtrování, které by netrpěly problémy jako je *cold-start user* nebo *cold-start item problem*, ale zároveň by poskytovaly stejně dobrá doporučení.

V této práci se mi podařilo úspěšně navrhnout a implementovat několik takových modelů a na reálných datech online e-shopů jsem pak ukázal, že *content-based* doporučovací model, založený na zpracování obrázků doporučovaných položek, může být kolaborativnímu filtrování více než zajímavou alternativou. V testu se taktéž potvrdila prvotní domněnka, že pro doporučovací model založený na algoritmu Image K-NN je velmi důležitá kvalita prediktoru podobnosti doporučovaných položek.

Jako nejlepší se na testovacích datasetech ukázal být model založený na předtrénované umělé neuronové síti. Jeho doporučení, na rozdíl od druhého nejlepšího modelu založeného na algoritmu ORB, dosáhla lepších výsledků podle obou evaluačních metod. Nejhůře z testovaných dopadl doporučovací model založený na algoritmu barevného histogramu. Podle evaluační metody *catalog coverage* sice dosáhl ještě o něco lepšího pokrytí datasetu, než model založený na neuronové síti, z hlediska testu na *recall* ale naprosto propadl a můžeme říci, že jeho doporučení nejsou relevantní.

Díky své úspěšnosti v offline testech byl model založený na neuronové síti nasazen do online A/B testu proti produkčnímu algoritmu, založeném na kolaborativním filtrování. Výsledek testu ukázal, že doporučení poskytovaná navrhovaným modelem mají srovnatelnou proklikovost jako model založený na kolaborativním filtrování a má tedy smysl se tímto přístupem dále zabývat. Věřím, že v dalších iteracích budu schopný navrhovaný model ještě vylepšit a proklikovost produkčního doporučovacího modelu dorovnat, možná i předčit.

Pro svůj doporučovací model založený na neuronové síti jsem z důvodu trénovací náročnosti na data a výpočetní výkon použil předtrénovanou umělou neuronovou síť. Pro dosažení lepších výsledků by do budoucna mohlo být zajímavé prozkoumat možnosti návrhu vlastní architektury sítě a trénování na interakčních datech, se zaměřením přímo na doporučování. Síť by tak měla být schopná lépe detekovat důležité vlastnosti obrázků, které je činí skutečně interakčně blízkými, a které shledávají i uživatelé zajímavé.

Věřím, že doporučovací systémy mají smysl a investice do jejich vývoje se nám, jako lidstvu, dlouhodobě vyplatí.

Literatura

[1] Ricci, F.; Rokach, L.; Shapira, B.: Introduction to recommender systems handbook. In *Recommender systems handbook*, Springer, 2011, ISBN 978-0-387-85819-7, s. 1–35.

[2] Ha, S. H.: An intelligent system for personalized advertising on the Internet. In *International Conference on Electronic Commerce and Web Technologies*, Springer, 2004, ISBN 978-3-540-22917-9, s. 21–30.

[3] Marinho, L. B.; Nanopoulos, A.; Schmidt-Thieme, L.; aj.: Social tagging recommender systems. In *Recommender systems handbook*, Springer, 2011, ISBN 978-0-387-85819-7, s. 615–644.

[4] Das, A. S.; Datar, M.; Garg, A.; aj.: Google news personalization: scalable online collaborative filtering. In *Proceedings of the 16th international conference on World Wide Web*, ACM, 2007, ISBN 978-1-59593-654-7, s. 271–280.

[5] Linden, G.; Smith, B.; York, J.: Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, ročník 7, č. 1, 2003: s. 76–80, doi:10.1109/MIC.2003.1167344.

[6] Park, S.-T.; Pennock, D. M.: Applying collaborative filtering techniques to movie search for better ranking and browsing. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2007, ISBN 978-1-59593-609-7, s. 550–559.

[7] Melville, P.; Mooney, R. J.; Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In *In Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-2002*, 2002, ISBN 0-262-51129-0.

[8] Desrosiers, C.; Karypis, G.: A comprehensive survey of neighborhood-based recommendation methods. In *Recommender systems handbook*, Springer, 2011, ISBN 978-0-387-85819-7, s. 107–144.

[9] Lops, P.; De Gemmis, M.; Semeraro, G.: Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, Springer, 2011, ISBN 978-0-387-85819-7, s. 73–105.

[10] Pazzani, M. J.; Billsus, D.: Content-based recommendation systems. In *The adaptive web*, Springer, 2007, ISBN 978-3-540-72078-2, s. 325–341.

[11] Sarwar, B.; Karypis, G.; Konstan, J.; aj.: Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, ACM, 2001, s. 285–295, doi: 10.1145/371920.372071.

[12] Kohavi, R.; Longbotham, R.: Online controlled experiments and a/b testing. In *Encyclopedia of Machine Learning and Data Mining*, Springer, 2017, ISBN 978-1-4899-7685-7, s. 922–929.

[13] Beel, J.; Genzmehr, M.; Langer, S.; aj.: A comparative analysis of offline and online evaluations and discussion of research paper recommender system evaluation. In *Proceedings of the international workshop on reproducibility and replication in recommender systems evaluation*, ACM, 2013, ISBN 978-1-4503-2465-6, s. 7–14.

[14] Dembczynski, K.; Kotlowski, W.; Weiss, D.: Predicting ads clickthrough rate with decision rules. In *Workshop on targeting and ranking in online advertising*, ročník 2008, 2008.

[15] Ignatov, D. I.; Poelmans, J.; Dedene, G.; aj.: A new cross-validation technique to evaluate quality of recommender systems. In *Perception and Machine Intelligence*, Springer, 2012, ISBN 978-3-642-27386-5, s. 195–202.

[16] Massa, P.; Avesani, P.: Trust-aware bootstrapping of recommender systems. In *ECAI workshop on recommender systems*, 2006, s. 29–33.

[17] Cremonesi, P.; Koren, Y.; Turrin, R.: Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, ACM, 2010, ISBN 978-1-60558-906-0, s. 39–46.

[18] Campochiaro, E.; Casatta, R.; Cremonesi, P.; aj.: Do metrics make recommender algorithms? In *Advanced Information Networking and Applications Workshops, 2009. WAINA'09. International Conference on*, IEEE, 2009, s. 648–653.

[19] Ge, M.; Delgado-Battenfeld, C.; Jannach, D.: Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems*, ACM, 2010, ISBN 978-1-60558-906-0, s. 257–260.

[20] Ohta, Y.-I.; Kanade, T.; Sakai, T.: Color information for region segmentation. *Computer graphics and image processing*, ročník 13, č. 3, 1980: s. 222–241.

[21] Swain, M. J.; Ballard, D. H.: Color indexing. *International journal of computer vision*, ročník 7, č. 1, 1991: s. 11–32, doi:10.1007/BF00130487.

[22] Rublee, E.; Rabaud, V.; Konolige, K.; aj.: ORB: An efficient alternative to SIFT or SURF. In *Computer Vision (ICCV), 2011 IEEE international conference on*, IEEE, 2011, ISBN 978-1-4577-1101-5, s. 2564–2571.

[23] MaCulloch, W.: A logical calculus of the ideas immanent in nervous activity. *Bulletin of mathematical biophysics*, ročník 5, 1943: s. 115–133, doi: 10.1007/BF02478259.

[24] Russell, S. J.; Norvig, P.: *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited„ 2016.

[25] Jain, A. K.; Mao, J.; Mohiuddin, K. M.: Artificial neural networks: A tutorial. *Computer*, ročník 29, č. 3, 1996: s. 31–44.

[26] Simonyan, K.; Zisserman, A.: Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] Underwood, R. L.; Klein, N. M.; Burke, R. R.: Packaging communication: attentional effects of product imagery. *Journal of product & brand management*, ročník 10, č. 7, 2001: s. 403–422, doi:10.1108/10610420110410531.

[28] Jovic, M.; Milutinovic, D.; Kos, A.; aj.: Product Presentation Strategy for Online Customers. *J. UCS*, ročník 18, č. 10, 2012: s. 1323–1342, doi: 10.3217/jucs-018-10-1323.

[29] Swinyard, W. R.: The effects of mood, involvement, and quality of store experience on shopping intentions. *Journal of consumer research*, ročník 20, č. 2, 1993: s. 271–280, ISSN 00935301, 15375277.

[30] Szegedy, C.; Ioffe, S.; Vanhoucke, V.; aj.: Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, ročník 4, 2017, str. 12.

Dodatek A

Seznam použitých zkratek

CB Content-based doporučovací model

CF Collaborative filtering (Kolaborativní filtrování)

NN Artificial neural network (Umělá neuronová síť)

ORB Oriented FAST and Rotated BRIEF

K-NN K nearest neighbours (K nejbližších sousedů)

CPU Central processing unit

GPU Graphical processing unit

Dodatek B

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `src` ..... zdrojové kódy k práci
  - `impl` ..... zdrojové kódy implementace
    - `py` ..... pyhon skripty
    - `cs` ..... C-Sharp skripty
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF