Dobré odpoledne, jmenuji se Ladislav Martínek. Rád bych vám představil svoji Bakalářskou práci. Bakalářská práce je zaměřena na téma Evaluace algoritmů Lokálně senzitivního hashování v doporučovacích systémech. Zpracovával jsem jí pod vedením Ing. Tomáše Řehořka na Fakultě informačních technologií ČVUT v Praze.

Cílem bakalářské práce je nastudovat různé metody lokálně senzitivního hashování. Pro tyto metody navrhnout způsob, jak je začlenit do algoritmu nejbližších sousedů. Na tomto základu navrhnout knihovnu pro testování různých parametrizací LSH v doporučovacích systémech a tu následně implementovat. Otestovat různé parametrizace metod LSH na dvou datových sadách. Na závěr diskutovat výsledky testování.

Motivací pro výběr tohoto tématu pro mě byl především můj zájem o oblast doporučovacích systémů, která mě velice zajímá. Téma na aproximaci výpočtu jsem si zvolil na základě množství dat, které je každou vteřinou zaznamenáváno na internetu. Takové množství dat je pak velice složité zpracovávat v doporučovacích systémech. V neposlední řadě byla motivace vytvořit a publikovat knihovnu, pomocí které bude možné testovat různé parametrizace metod LSH na různých datových sadách.

Rád bych specifikoval základní pojmy o dané problematice. Doporučovací systém je systém, který uživateli může nabízet položky, které by pro něj mohli být zajímavé. Jedním z přístupů k tomuto doporučování je kolaborativní filtrování. Potencionální položky jsou vybírány na základě předchozí podobnosti položek nebo uživatelů. V mé práci se zabývám konkrétně podobností jednotlivých uživatelů a algoritmem pro hledání k nejbližších v metrickém prostoru kosinové podobnosti. Kosinová podobnosti je závislá na úhlu, který spolu svírají vektory uživatelů. Lokálně senzitivní hashování se na rozdíl od funkcí SHA a MD snaží, aby podobné vektory měli podobný nebo stejný hash. Uživatelské vektory jsou tedy převáděny na sekvenci bitů, pomocí kterých je prostor dělen na části, které se nazývají buckety. V bucketech jsou poté potencionální uživatelé, pokud je dotazující vektor uživatele zahashován do konkrétního bucketu.

Knihovna vytvořená v rámci práci, se skládá ze několika modulů. První modul předzpracování umožnuje získávat data. Data lze implementací abstraktní třídy získávat z různých datových struktur. Já jsem v práci načítal data z csv souborů jako n-tice uživatel, položka, hodnocení. Stažená data jsou předzpracována do slovníků pro rychlý přístup a vyhodnocení v algoritmech nejbližších sousedů. Dalším modulem je modul k-NN, který slouží pro hledání nejbližších sousedů v dané množině uživatelů. Pomocí modulu je spočítáno referenční řešení k porovnání s metodami LSH. Modul LSH slouží k indexaci prostoru a rozdělení uživatelů do bucketů, pomocí kterého je z rozdělení uživatelů hashovacími funkcemi vybrána podmnožina uživatelů. Model je nastaven podle argumentů z příkazové řádky. Při vyhodnocování, jsou vráceni potenciální uživatelé z bucketu, kam byl zahashován dotazovaný uživatel, nebo z blízkých bucketů. Hlavní součástí jsou dva moduly s testovacími skripty. Testování jsem rozdělil z důvodu odlišnosti testování přesnosti na k nejbližších sousedů, které je nutné vždy porovnávat vůči referenčnímu řešení, a testování úspěšnosti doporučování, které může probíhat nezávisle na referenčním modelu, ale je výhodné z důvodu porovnání úspěšnosti. Testování doporučování je prováděno leave-one-out křížovou validací.

Hashovací funkce jsem implementoval jako projekční matice, které jsou generováni z uniformního rozdělení od -1 do 1. Prvním přístupem, který lze využít ke zvýšení efektivity je přidávání hashovací funkcí, při kterém je nutné efektivně provádět sjednocení potencionálních uživatelů. Z důvodu sjednocení má ovšem takové přidávání hranici, kdy je ještě efektivní. Další metodou je metoda multi-probe. Metoda je založena na pravděpodobnosti, že by podobný uživatel zahashován do blízkého bucketu. V práci jsem tuto metodu implementovat pomocí hammingovy vzdálenosti binární reprezentace bucketů. Tedy v kolika bitech se jednotlivé buckety liší. Poslední metodou je úprava hashovací funkce podle popularity položek, protože v každé databázi jsou položky, které jsou u uživatelů oblíbenější než jiné. Tyto položky mají na výsledný hash větší vliv než jiné položky.

Knihovna je implementována v jazyce Python. Ke knihovně by zároveň implementován i modu pro tvorbu grafů, pomocí kterého lze jednoduše vytvářet přehledné grafy nad výsledku testování. Nejdůležitější argumenty, které slouží k nastavení metod LSH jsou. Dimenze hashovací d funkce určuje délku binární reprezentace bucketu, tedy počet bucketů a míru aproximace alogoritmu k-NN. Nastavení metody multi-probe c určuje maximální množství změn bitů v reprezentaci bucketu. Pomocí malého m lze určovat parametry každé hashovací funkci zvlášť, tedy rozsah hodnot, míra vlivu popularity položek. Velké M je počet standartních hashovacích funkcí.

Modely jsem nejprve testoval na přesnost hledání 10 nejbližších sousedů, protože toto testování bylo výrazně efektivnější. Úspěšnost doporučování jsem otestoval na vybraných modelech, kde počet hashovacích funkcí se pohyboval mezi 128–256 hashovacími funkcemi, velikost prohledávání okolí byla 1 a nebyla využita popularita položek. Většinu modelů jsem otestoval na databázi A, kde můžete vidět, že je výrazně menší než druhý databáze, uživatel v této databázi interagoval průměrně s 4 položkami. V databázi B bylo těchto interakcí 50, a databáze byla mnohem rozsáhlejší.

Na grafu jsou vidět optimální modely testované na databázi B na přesnost 10 nejbližších sousedů. Je vidět že při 10 % čase se přesnost pohybuje okolo 90 %. Tedy 9/10 sousedů je určeno správně. Při testování doporučování se při čase okolo 2–5 % pohyboval recall na hodnotě 96–98 %. Tedy byla výrazně snížena výpočetní náročnost při nepatrné změně úspěšnosti. Při aproximaci dosahovali modely LSH vyšší catalog coverage oproti referenčnímu řešení, tedy doporučovali více položek z databáze. Při testování byla také zjištěna lineární závislost mezi přesností 10-NN a úspěšností doporučování, díky které bylo možné otestovat velké množství různých modelů.

Na závěr bych řekl, že se pomocí kombinace implementovaných metod podařilo dosáhnout velmi dobrého poměru zrychlení na úkor zhoršení přesnosti, dále při testování byli zjištěny parametry optimálních modelů, které je možno lehce dopočítat pro libovolnou databázi. Při implementaci a testování, byli nalezeny cesty, kterými by bylo možné knihovnu rozšířit. V budoucí práci bych se určitě zaměřil na paralelizaci, omezení maximálního počtu uživatelů v jednom bucketu využitím metody multi-probe a také jak metody nasadit na online data, kde není předem známá velikost a modely by se museli upravovat při běhu.

Zde jsem uvedl důležité literární zdroje.

Děkuji za pozornost, nějaké dotazy?.