Privacy-aware and scalable recommender systems uing sketching techniques

Raghavendran Balu

ANNÉE 2016

THÈSE / UNIVERSITÉ DE RENNES 1
sous le sceau de l'Université Bretagne Loire

pour le grade de

DOCTEUR DE L'UNIVERSITÉ DE RENNES 1

Mention : Informatique

École doctorale Matisse

présentée par

Raghavendran BALU

préparée à l'unité de recherche IRISA
Université de Rennes 1

Privacy-aware and Scalable Recommender Systems using Sketching Techniques

Thèse soutenue à Rennes
le 9/11/2016

devant le jury composé de :

Neil HURLEY
Senior Lecturer, University College Dublin / *Rapporteur*

Alexandros KARATZOGLOU
Senior Research Scientist, Telefonica Research / *Rapporteur*

Sébastien GAMBS
Professor, University of Québec, Montréal / *Examinateur*

Anne-Marie KERMARREC
Senior Researcher, INRIA-Rennes / *Examinatrice*

Laurent AMSALEG
Research scientist, CNRS/IRISA / *Directeur de thèse*

Teddy FURON
Researcher, INRIA-Rennes / *Co-directeur de thèse*

Acknowledgements

I owe my sincere gratitude to my advisors Laurent Amsaleg and Teddy Furon for their exceptional guidance and incessant support throughout the thesis work. Their diverse and in-depth technical expertise along with systematic approach towards problem solving carved a significant and indivisible impact in the thesis work. Their expertise in presenting and publishing not only helped in expressing the thesis contributions in an accessible and precise manner but also improved my technical communication skills for good. I also thank them for the innumerous discussions that I had with them and also for being available and approachable all throughout.

I also profoundly thank Sébastien Gambs and Hervé Jégou for their valuable contributions that shaped the thesis. Their unmatched technical knowledge and expertise enriched the contributions and made the thesis realizable.

I am also grateful to Alcatel-Lucent INRIA common labs for their generous funding and IRISA/INRIA-Rennes for providing all the necessary facilities for a conducive research environment. I also want to thank Armen Aghasaryan and his team for the hosting me during my regular visits to the Bell labs facility at Paris and their stimulating discussions during my stay at their facility.

I also would like to thank the other Jury members: Neil Hurley, Alexandros Karatzoglou and Anne-Marie Kermarrec for accepting to be part of the jury and expending their valuable time to evaluate the manuscript, also for their constructive comments and thoughtful questions.

I also want to thank my fellow team members and friends for making the entire stay lively and memorable. I finally want to thank my parents for all their sacrifices, motivation and support that made me who I am. Their unconditional love and belief in me is what keeps me going.

Contents

Résumé en français

Introduction

Internet modifie constamment nos façons de communiquer, d'apprendre et de découvrir de nouvelles choses. Les moteurs de recherche comme Google sont l'exemple le plus connu. Grâce à l'interconnexion incessante entre humains via Internet, notre pouvoir de découvrir n'est plus seulement lié à notre proche entourage, mais à l'ensemble de la planète. Archiver, synthétiser, et pouvoir questionner toute cette activité humaine souvent parcellaire est le défi du "crowd intelligence".

Un outil clé est le système de recommendation qui analyse notre activité passée pour nous proposer des suggestions dignes de notre intérêt. Cela nous pousse à donner une appréciation sur nos expériences ce qui améliorera en retour la qualité des futures suggestions qui nous seront destinées ainsi que pour tous ceux qui partagent les mêmes goûts.

Les systèmes de recommendation sont nés dans les années 90 pour la musique [SM95], les videos [HSRF95], les actualités [RIS+94] et même les blagues [GRGP01]. Le succès d'Amazon qui utilise cette technologie pour améliorer ses ventes [LSY03] a popularisé les systèmes de recommendation. Des compétitions internationales comme celle organisée par Netflix [BL07] ont intensifié la recherche dans cette discipline.

Les qualités d'un système de recommendation sont les suivantes :

- le passage à l'échelle : en commerce électronique, un système doit pouvoir traiter toujours plus d'utilisateurs et de biens. Dans la recherche d'information ou la publicité en ligne, la situation est encore plus critique car l'ordre de grandeur du nombre de pages web ou de publicités est beaucoup plus grand.

- la dynamicité : le système doit être dynamique, gérant de nouveaux utilisateurs ou des biens qui ne sont plus à vendre. Ceci doit se retrouver à la fois dans l'infrastructure qui analyse les données avec des contraintes d'espace mémoire et de temps de calcul, et dans la qualité des suggestions faites à des utilisateurs dont les goûts évoluent.

- la confidentialité : cet aspect, au sens du respect de la vie privée, a longtemps

été négligé. Les systèmes de recommendation analysent des données sensibles car personnelles. L'anonymisation des données n'est pas suffisante car elle peut être compromise en croisant différentes bases de données [NS08]. Il faut ainsi perturber aléatoirement les données selon le principe de la confidentialité différentielle ('differential privacy' en anglais).

Nos contributions sont les suivantes :

- Une étude défiant la confidentialité différentielle lorsqu'elle est appliquée au calcul de la similarité entre deux profils utilisateurs. Nous jouons ici le rôle de l'attaquant dont le but est de révéler quels biens ont été consommés ou appréciés par tel utilisateur. La confidentialité différentielle donne des garanties sur la difficulté d'une telle attaque lorsque l'attaquant devine le profil de l'utilisateur bien par bien. En revanche, nous montrons qu'en analysant conjointement des sous-ensembles de biens, l'attaque est moins difficile. Ceci est prouvé théoriquement et instancié sur deux systèmes basés sur le filtre de Bloom (système BLIP [AGK12]) et sur la transformée de Johnson-Lindenstrauss [KKMM12]. Nous montrons aussi comment réaliser une attaque conjointe en pratique au moyen d'une simulation MCMC (Markov Chain Monte Carlo).

- Un système complet de recommendation scalable, dynamique et conforme à la confidentialité différentielle. Ce système procède à une factorisation de la matrice des appréciations utilisateur / bien. Les vecteurs latents sont appris et stockés via la technique du 'count-sketch' pour passer à l'échelle. Pour assurer la confidentialité différentielle, nous modifions la descente du gradient stochastique en une dynamique de Langevin stochastique. De plus, nous avons mis en place un protocole expérimental dédié aux systèmes de recommendation pour évaluer le niveau de confidentialité différentielle.

- Nous proposons une amélioration d'une autre technique de 'sketching' très connue: Locality Sensitive Hashing, aussi connue sous le nom de 'cosine sketch'. Cette technique voit le 'sketching' comme une quantification vectorielle. Minimiser l'erreur de reconstruction conduit à de meilleurs résultats de recherche. De plus, cela permet de reconstruire approximativement les données intéressantes pour un calcul de similarité affiné.

Analyse de confidentialité différentielle

Soit P le profil d'un l'utilisateur vu comme une séquence binaire indiquant quels biens ont été consommés par celui-ci. Des approches [KKMM12, AGK12] proposent d'y associer une représentation compacte $\tilde{\mathbf{B}}_P$ difficile à inverser mais permettant un calcul approché de la similarité entre deux profils. Cette dernière opération est nécessaire pour

recommander à l'utilisateur les choix faits par d'autres utilisateurs proches, c'est-à-dire ayant un profil similaire. La représentation $\tilde{\mathbf{B}}_P$ est en fait aléatoire pour respecter le principe de la confidentialité différentielle.

Nous jouons maintenant le rôle de l'attaquant qui observe $\tilde{\mathbf{B}}_P$ et qui veut savoir si l'utilisateur a consommé le bien j. L'attaquant procède donc à un test d'hypothèse binaire : ou bien l'utilisateur a consommé ce bien ou bien il ne l'a pas fait. En découlent deux types d'erreur de probabilités α_1 (faux positifs) et α_2 (faux négatifs). Le lemme de Stein assure que

$$\alpha_1 \geq e^{-(I(\tilde{\mathbf{B}}_P;\mathbf{X})+1)/(1-\alpha_2)}, \tag{1}$$

où $I(\tilde{\mathbf{B}}_P;\mathbf{X})$ est l'information mutuelle entre la représentation $\tilde{\mathbf{B}}_P$ et la signature $\mathbf{X}$ du bien j utilisée dans le calcul de $\tilde{\mathbf{B}}_P$. En testant tous les N biens, l'attaquant reconstruit le profil de l'utilisateur avec une probabilité de faux positif globale η_1. Ceci n'est possible que si l'ensemble de biens n'est pas trop grand:

$$\log(N) \leq \frac{I(\tilde{\mathbf{B}}_P;\mathbf{X})}{1-\alpha_2} + \log \eta_1. \tag{2}$$

où on voit le rôle fondamental de $I(\tilde{\mathbf{B}}_P;\mathbf{X})$. En fait, la confidentialité différentielle oblige $I(\tilde{\mathbf{B}}_P;\mathbf{X})$ à prendre de petites valeurs. Par conséquent, en procédant bien par bien, l'attaquant ne pourra pas reconstruire tout le profil.

Supposons que l'utilisateur ait consommé c biens. L'attaquant considère maintenant conjointement un c-uplet de biens et teste si celui-ci serait égal au profil de l'utilisateur. Procéder par c-uplet amène plus d'infirmation, mais le nombre de c-uplets à tester est bien plus grand. Le même raisonnement conduit à une inégalité similaire :

$$\log(N) \leq \frac{I(\tilde{\mathbf{B}}_P;P)}{c(1-\alpha_2)} + \log \eta_1. \tag{3}$$

où $I(\tilde{\mathbf{B}}_P;P)$ est l'information mutuelle entre la représentation $\tilde{\mathbf{B}}_P$ et le profil P. Le théorème [Mou08, Eq. (3.4)] prouve que dans tous les cas $I(\tilde{\mathbf{B}}_P;P)/c \geq I(\tilde{\mathbf{B}}_P;\mathbf{X})$, ce qui montre qu'une analyse conjointe des biens produit toujours une meilleure attaque. Le reste de l'étude calcule ces informations mutuelles pour les deux schémas BLIP et JLT. Elle montre que si la confidentialité différentiellle assure bien que $I(\tilde{\mathbf{B}}_P;\mathbf{X}_j) \leq \epsilon$, ce n'est pas le cas pour $I(\tilde{\mathbf{B}}_P;P)/c$.

L'attaque jointe est théorique : en pratique, l'attaquant doit tester tous les sous-ensembles de taille c parmi N. Soit une complexité en $O(N^c)$. Nous proposons d'utiliser une chaîne de Markov et un échantillonneur de Gibbs multi-étapes pour explorer de manière efficace l'ensemble des c-uplets. Après une période transitoire, la chaîne échantillonne des profils suivant la distribution *a posteriori* des profils. Il ne reste plus qu'à retenir le profil ayant la plus grande vraisemblance ou la plus grande probabilité *a posteriori* (en incluant une information a priori sur les biens les plus consommés) ou encore à estimer les probabilités marginales *a posteriori* par bien afin de les classer.

Table 1: Caractéristiques des jeux de données

Dataset	Nb utilisateurs	Entraînement	Test	N	c_{avg}	Remplissage %
Digg	531	331	200	1237	317	25.63%
MovieLens	943	600	343	1682	106	6.30%

L'étude expérimentale se base sur deux jeux de données : Digg et MovieLens (cf. Tableau 1). Le lecteur trouvera les résultats expérimentaux dans le chapitre 3. Pour résumer, la confidentialité différentielle fonctionne bien : pour des ϵ petits, l'attaque bien par bien est contenue. Elle produit des résultats pire que l'attaque 'idiote' qui reconstruit le profil par les biens les plus populaires. En revanche, dès que l'attaquant sort du contexte de la confidentialité différentielle avec l'attaque jointe, plus aucune garantie ne tient. Cette étude révèle aussi que, pour la même utilité, le schéma JLT [KKMM12] produit plus de confidentialité que BLIP [AGK12], ce qui n'est pas étonnant puisque sa représentation est moins compacte.

Cette première contribution offre donc une analyse critique du concept de confidentialité différentielle.

Factorisation de matrice par technique de 'sketching'

Cette étude est une proposition d'un système de recommendation centralisé ayant les propriétés listées dans l'introduction. Soit $\mathbf{R}$ la matrice des appréciations utilisateur / bien : $r_{u,i}$ est la note laissée par l'utilisateur u concernant le bien i. Cette matrice contient des données manquantes, elle est très creuse. Notre système modélise les utilisateurs et les biens par des vecteurs latents à d composantes: $\mathbf{p}_u$ pour l'utilisateur u et $\mathbf{q}_i$ pour le bien i de telle manière que l'appréciation $r_{u,i}$ sera prédite par le produit scalaire $\mathbf{p}_u^\top \mathbf{q}_i$.

On définit la fonctionelle suivante :

$$L(\mathbf{P}, \mathbf{Q}) = \sum_{<u,i,r_{u,i}> \text{ observés}} L(r_{u,i}, \mathbf{p}_u^\top \mathbf{q}_i) \tag{4}$$

qui est la somme des coûts d'erreur de prédiction entre $r_{u,i}$ et $\mathbf{p}_u^\top \mathbf{q}_i$ (ici ce coût $L(\cdot, \cdot)$ est l'erreur quadratique). L'apprentissage consiste à trouver les facteurs $\mathbf{P}$ et $\mathbf{Q}$ (les matrices contenant les vecteurs latents des utilisateurs et des biens) minimisant cette fonctionelle avec un terme de régularisation introduit sous la forme d'une pénalité sur la norme au carré des vecteurs latents:

$$\mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) = \sum_{<u,i,r_{u,i}> \text{ observés}} L(r_{u,i}, \mathbf{p}_u^\top \mathbf{q}_i) + \frac{\lambda}{2}(\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2). \tag{5}$$

Ceci est un problème d'optimisation simple mais rendu difficile par la taille des facteurs. Une solution approchée avec une complexité raisonnable est habituellement donnée par l'algorithme de descente de gradient stochastique. Cependant, l'échelle de notre système est tellement grande que cette solution reste encore trop complexe. Stocker les facteurs est en fait problématique. C'est pour cette raison que nous proposons l'utilisation d'une technique de 'sketching' vue comme unité de stockage.

Nous avons choisi la technique du 'count sketch' pour stocker dans une matrice $\mathbf{C}$ unique de taille (w, k) toutes les composantes de tous les vecteurs latents, utilisateur et bien. Si l'étape de lecture du 'count sketch' utilise l'opérateur moyenne (et non médian comme souvent choisi), nous montrons que la descente de gradient sur les composantes des vecteurs latents stockés par count sketch revient à une descente de gradient sur les entrées de matrice $\mathbf{C}$ elles-mêmes où la pénalisation sur la norme au carré des vecteurs latents est en fait une pénalisation sur la variance de l'erreur de lecture du 'count-sketch'.

L'utilisation du 'count-sketch' réduit l'espace mémoire nécessaire mais introduit du bruit : une valeur lue est égale à la valeur écrite auparavant plus du bruit. Ainsi, on s'attend à une baisse de la qualité de la recommendation. L'étude expérimentale montre que cette baisse est relativement faible. Nous pensons que le bruit introduit a un effet 'self-régularisant' qui évite un sur-apprentissage. Une preuve expérimentale est que le paramètre λ de pénalisation a un impact moindre, et que le système converge plus vite. Ceci a le bon goût de rendre le système plus dynamique.

Ajouter du bruit dans les calculs internes d'un algorithme est un mécanisme récemment proposé pour se conformer au concept de confidentialité différentielle. Le bruit induit par le 'count-sketch' transforme la descente de gradient en une dynamique stochastique de Langevin (Stochastic Gradient Langevin Dynamics). Ainsi, l'apprentissage ne cherche plus les facteurs au sens du maximum *a posteriori* en minimisant (5), mais tire aléatoirement les facteurs suivant leur distribution *a posteriori* [WT11, VZT15]. Or des travaux récents [WFS15, LWS15] montre que, d'une manière très générale et sous certaines conditions, ceci est un mécanisme produisant une confidentialité différentielle de niveau ϵ.

Tout ceci est théorique, impliquant des hypothèses peu vérifiables en pratique. Nous avons créé un protocole expérimental pour mesurer en pratique le niveau de confidentialité différentielle, chose totalement absente dans la littérature. L'idée est toute simple : on tire au hasard une note $r_{u,i}$ dans le jeu de données d'apprentissage ou dans celui de test, on met au défi l'attaquant de savoir si cette appréciation a été utilisée ou non lors de l'apprentissage. L'attaquant a le droit de questionner le système de recommendation pour obtenir l'estimation $\mathbf{p}_u^\top \mathbf{q}_i$. En répétant ce défi un grand nombre de fois, nous obtenons la distribution empirique de l'erreur $r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i$ pour les notes de chaque jeu de données. Puis, nous vérifions que leur distance de Kullback-Leibler décroit avec la valeur théorique ϵ. Autrement dit, l'attaquant ne peux décider avec

précision si l'utilisateur u a soumis ou non une appréciation concernant le bien i et ce même s'il connaît la vraie valeur de cette appréciation.

Amélioration de recherche d'information par technique de 'sketching' LSH

La recherche des profils utilisateur les plus similaires ou des vecteurs latents les plus pertinents pour un utilisateur donné suppose le calcul d'un très grand nombre de similarités. Nous considérons ici les techniques d'encodage binaire qui transforment un vecteur $\mathbf{x}$ de $\mathbb{R}^d$ en un mot binaire $\mathbf{b}(\mathbf{x})$ dans $\mathbf{B}^L$ de telle manière que la similarité 'cosinus' $\mathbf{x}^\top\mathbf{y}/\|\mathbf{x}\|\|\mathbf{y}\|$ est estimée au moyen de la distance de Hamming $d_h(\mathbf{b}(\mathbf{x}), \mathbf{b}(\mathbf{y}))$ bien plus rapide à calculer sur les ordinateurs actuels. En particulier, nous analysons la technique dite LSH proposée par Charikar [Cha02]. Ayant généré aléatoirement L vecteurs $\{\mathbf{w}_j\}_{j=1}^L$ uniformément sur la sphère unité, le signe de la projection $\mathbf{w}_j^\top\mathbf{x}$ produit un bit de signature :

$$b_j(\mathbf{x}) = \mathsf{sign}(\mathbf{w}_j^\top\mathbf{x}). \tag{6}$$

La signature de $\mathbf{x}$ est la concaténation de ces bits :

$$\mathbf{b}(\mathbf{x}) = [b_j(x)]_{j=1}^L . \tag{7}$$

Si $L > d$, il est impossible de générer L directions orthogonales. Il est d'usage de prendre $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_L]$ [GVT98] telle que $\mathbf{W}.\mathbf{W}^\top \propto \mathbf{I}_d$ (autrement dit une 'tight frame') [JFF12, SVZ13].

Nous soulignons la sous optimalité de cette technique 'project and sign' d'un point de vue de la quantification. Nous considérons le vecteur reconstruit $\hat{\mathbf{x}}$ suivant :

$$\hat{\mathbf{x}} \propto \sum_{j=1}^{L} b_j(\mathbf{x})\mathbf{w}_j = \mathbf{W}\mathbf{b}(\mathbf{x}), \tag{8}$$

où la constante de proportionalité telle que $\|\hat{\mathbf{x}}\| = 1$. La technique LSH associée à cette reconstruction ne donne pas la meilleure quantification (des exemples simples en $d = 2$ dimensions sont faciles à trouver). Elle n'est pas surjective car certaines signatures sont impossibles. Cependant, la quantification optimale est très complexe : il s'agit de parcourir tous les 2^L mots de $\mathbf{B}^L$ pour retenir celui qui offre la meilleure reconstruction. Une alternative est la méthode des représentations étalées [Fuc11, JFF12].

Nous proposons une méthode ayant un compromis complexité / qualité de reconstruction plus attractif. Partant du résultat de la technique 'project and sign', nous cherchons quel bit parmi les L modifier pour améliorer la reconstruction. Cette recherche est effectuée séquentiellement M fois. Notre méthode est sous optimale mais simple.

Le point de vue de la quantification amène aussi une autre idée en recherche de vecteurs similaires. Pour un vecteur requête donné, on calcule rapidement les similarités approchées grâce aux signatures des vecteurs de la base. Ceux ayant les plus grandes similarités sont stockés dans une liste. Il est maintenant possible de reconstruire ces vecteurs à partir de leurs signatures, ce qui donnera une meilleure estimation de leurs similarités avec le vecteur requête. Autrement dit, nous pouvons raffiner ce premier classement à peu de frais.

Nous montrons que notre technique produit une estimation de la similarité 'cosinus' avec moins de biais et moins de variance (cf. fig. 5.1). Ceci donne de meilleurs résultats de recherche dans une grande base (cf. fig. 5.2).

Conclusion

Le thème central de la thèse est l'étude et l'évaluation de la confidentialité et de la scalabilité des systèmes de recommendation. Concernant la confidentialité, nous avons intégré, évalué et proposé de nouveaux mécanismes. Nous observons aussi que le niveau ϵ de confidentialité différentielle n'est qu'un paramètre de contrôle dans la littérature et nous avons ressenti le besoin de mesurer réellement cette garantie. Concernant la scalabilité, nous avons montré que les techniques de 'sketching' apportent une amélioration tout en ne sacrifiant pas trop la dynamicité et la qualité des recommendations.

Le message le plus important est que confidentialité et scalabilité ne s'opposent pas. La scalabilité induit nécessairement un compromis temps-espace *versus* approximation. C'est cette approximation qui non seulement prévient le sur-apprentissage, mais aussi renforce la confidentialité différentielle.

Les travaux de cette thèse ont été publiés dans les articles suivants :

- *Challenging differential privacy: the case of non-interactive mechanisms*, Raghavendran Balu, Teddy Furon, Sébastien Gambs. ESORICS European Symposium on Research in Computer Security, sep. 2014, Wroclaw, Pologne. **Prix du meilleur papier étudiant**.
- *Sketching techniques for very large matrix factorization*, Raghavendran Balu, Teddy Furon, Laurent Amsaleg. ECIR 38th European Conference on Information Retrieval, mars 2016, Padoue, Italie.
- *Differentially Private Matrix Factorization using Sketching Techniques*, Raghavendran Balu, Teddy Furon. IHMMSec ACM workshop on Information Hiding and MultiMedia SECurity, juin 2016, Vigo, Espagne.
- *Beyond "project and sign" for cosine estimation with binary codes*, Raghavendran Balu, Teddy Furon, Hervé Jégou. IEEE ICASSP International Conference on Acoustics, Speech, and Signal Processing, mai 2014, Florence, Italie.

Chapter 1

Introduction

The advent of internet has revolutionized the society by totally reshaping the way we learn, communicate, express our opinion and purchase products. Search engines like Google completely altered the manner in which we search and acquire information. This did not stop with search alone, resonating in other activities. One such is our approach to discover and try new things. The de-facto approach before internet era was to seek our close social connections and networks for suggestions. Thanks to the incessant interconnectivity, now we no longer have to rely on our immediate network, but contribute and probe from a much larger pool. These pooling in and probing from diverse sources with incomplete information is also called as crowd intelligence. This is accelerated with access to intelligent systems, which, using our previous behavior and preference, fine tune suggestions to us. These are collectively called as recommender systems. Recommender systems fill a niche gap of exploring new and unknown things and also exploiting known possibilities. This also motivates people to give feedback which can be utilized by other like minded users, closing the loop. As a side effect, it also gives the much necessary feedback, thereby facilitating a collective voice for the consumers.

The need for intelligent recommender systems was recognized during the late nineties, owing to the growth of the internet and the ever increasing content available for the end-users. It started out with domains such as music [SM95], videos [HSRF95], news [RIS+94] and jokes [GRGP01]. The success of Amazon Inc. in utilizing recommender systems [LSY03] to improve the sales and product engagement demonstrated its application in a commercial setup and motivated other service providers to follow the trail. The Netflix recommender systems for movies and their open competition [BL07] intensified the attention and fueled further research on this evolving discipline. The commonality of the problem among diverse applications motivated to devise generic solutions that are independent of the application domain.

Systems providing recommendations for Amazon, Netflix, Google and other big

internet players are typically centralized. Such architectures are easy to control and tune, solid and well established large scale, efficient and fault tolerant data storage solutions can be used, computational demands can be well addressed. In addition, the very important security and privacy requirements are easy to enforce in such common platforms.

This thesis is done in the context of a collaboration between INRIA and Alcatel-Lucent Bell labs. It therefore has a very strong industrial component. Alcatel-Lucent Bell is very much interested in recommendation systems (see [NAB11, ABK+13]). Proposing contributions that are close to the real world is hence a key element in their research agenda. For that reason, the work in this thesis is also considering centralized recommendation systems. We are well aware, however, that some academic works have recently proposed fully distributed recommendation systems that for example use the more sophisticated peer-to-peer architectures. These systems have nice properties such as a better preserving of the privacy of the users receiving recommendations. The industrial context for this thesis, however, is pushing out of our scope such non-centralized recommendation contributions.

1.1 Challenges

With the rise of a new discipline, came new challenges. Some of these challenges are scalability, privacy and dynamism.

Scalability

Thanks to relentless connectivity and access to wide sources of information, more and more people are turning to internet based services. The number of internet users is estimated to be in billions. This is only increasing with new interconnectivity in developing countries. On the other hand, more and more items are made available to these users. The magnitude of these items also depend on the domain. In e-commerce, millions of products are made available for end users, with new products being added to the catalogue constantly. Similar conditions prevail in creative industries like music, movies and fiction works. In domains like search and advertising, the situation is even more pronounced as the number of web pages and ads are much higher. Furthermore, the same user can be served hundreds of a time in any given day, making latency additional constraint. All these requires a highly scalable recommender system both to increasing user base and items. This motivates us to address the scalability problems of such recommender systems.

Dynamism

The users and items are not static in many cases and constantly evolving over time. It is becoming increasingly uncertain if the same user or item will be served again. Also the addition of new information such as items and feedback, that is fed to the system invalidates the user/item models in place. Hence static modeling is becoming a thing of past, requiring dynamic systems that can not just endure this data deluge, but take advantage of such evolving data over time. Dynamic capabilities are required from both infrastructure and relevance point of view. The infrastructure requirements are to make sure the system is capable of handling the volume of data without failing and preferably efficient in terms of space of time demands. The infrastructure demands should not be met at the cost of relevance as user satisfaction is equally important. A highly scalable system that suggests random items is of no good. Hence dynamic capabilities must be in the requirements checklist of any realistic industry-scale recommender systems.

Retrieval efficiency

The scalability challenges of prediction of relevant items is not given its due importance compared to modeling. In case of large-scale systems, the cost of retrieval is prohibitive, that is worsening with many concurrent requests. Prediction usually involves retrieval and ranking of relevant items from a large collection of available items. Retrieving and finding such relevant items is often modeled as nearest neighbor search problem, which is well studied. On very large high dimensional datasets, finding exact nearest neighbors is time consuming and impractical, leading to approximation techniques. From a large collection of vectors in a high dimensional space, the approximate most similar search aims at extracting the most similar vectors to a query. Approximate neighbor search techniques compromise relevance to improve storage retrieval runtime efficiency. Hence there is a scope for improving the trade-off for such approximate neighbor search techniques. These scalability enhancements resonate into usability improvements and improve the quality of service.

Privacy

Among the various properties, privacy is one aspect which is often forgotten while designing recommender systems. Recommender systems handle user behavioral data which is personal to the user and sensitive data. The consumed items have to be collected at a centralized database, to perform both analysis and prediction, which compromises user privacy. If the data is accompanied by identifiable information it might lead in trouble for the users, because of the potential of abuse by malicious adversaries.

The usual technique to ensure privacy is anonymization of the data before release.

Anonymization includes modifying or removing identifiable information of individuals. The usual anonymization alone is not sufficient as demonstrated in [NS08]. In this work the authors proved the de-anonymization risks by linking users contributed to the celebrated Netflix prize dataset with other public profiles such as IMDB. But user modeling requires historical information of behavioral data and this leads to a stand-off: To release or not. At the end, the users are required to trust a centralized service as only such systems are known to be commercially viable and practical. All these motivate to look for better privacy preserving mechanisms that are both rigorously defined and at the same time practical to implement.

1.2 Solution

Collaborative filtering for centralized recommendations

Among various approaches to recommender systems, collaborative filtering is emerging as a prominent technique to provide recommendations. It uses the user-item rating relationship to provide recommendations. Collaborative filtering techniques can be broadly classified into two kinds: Neighborhood based and model based. Neighborhood methods utilize user-item similarity to find neighbors of a given profile and aggregate information from those neighbors to provide suggestions. On the other hand, model based systems utilize statistical modeling techniques to approximate user-item similarity and represent users and items that enables fast and accurate predictions. Both approaches are widely popular, scalable and promising in terms of prediction relevance. We consider both approaches in our recommender systems, with more emphasis and importance to model based systems. In model based systems, we consider the popular matrix factorization [KBV09] as our modeling technique in our recommender systems. Factorization techniques map both users and items to a low dimensional representation, retaining pairwise similarity. Compared to other methods, matrix factorization is simple, space efficient and can generalize well with missing information. It also has many desirable properties that suits statistical modeling, justifying to be our technique of choice.

Differential privacy for privacy preserving personalization

It is observed that anonymization alone is insufficient to enforce privacy. An alternative ingenious solution to anonymization comes to the rescue through the means of randomized perturbation of the data before release. Differential privacy is one such randomization mechanisms. Differential privacy argues that absolute privacy is impossible to achieve and instead settles to relative level. It was originally devised to answer queries on statistical databases without compromising on row level privacy. Since then it has garnered wide spread attention among the research community for its theoretical

rigorousness and robustness to auxiliary information. The privacy objective is achieved by means of a randomizing mechanism, which takes the database as input and perturbs the output such that it is not differentiable, to an extent, from its neighboring databases that differs by a row. Owing to the success of differential privacy in other data release problems we motivate utilizing differentially private mechanisms for user modeling in our privacy preserving system.

Sketching techniques for scalable privacy preserving personalization

The scalability and privacy challenges calls for designing scalable privacy preserving recommender systems. Conventional techniques add explicit noise to perturb the data to enable privacy, which serves no additional purpose. Hence it is desirable to look for efficient perturbation mechanisms are that are also inherently random. Recent advances in database community lead to a class of randomized techniques called data sketching. Sketching techniques were originally proposed to efficiently estimate and answer queries over data that come in a streaming fashion. Since then it has found applications in various other domains. Sketching techniques are space-efficient and scalable to data. They are simple in construction and facilitate efficient inserts, updates and estimation. They are also randomized by design. The inherent randomness of such sketch structure also preserve privacy as a side effect. Hence these benefits prompted us to incorporate sketching techniques into user modeling and prediction to satisfy these objectives. Sketching techniques are task specific and are defined to approximate a particular statistic of interest.

We use sketching technique to improve both user modeling and prediction. In the context of modeling, we first consider the task of compactly representing the item consumption binary information and similarity estimation using such compact representation. There are differentially private similarity estimation mechanisms using these compact representations. We studied two recent promising approaches: BLIP [AGK12] and Johnson-Lindenstrauss transform (JLT) [KKMM12]. These two approaches motivated us to evaluate the privacy preserving guarantees of such similarity estimation mechanisms.

We then examine the problem of user modeling with more informative data such as user ratings. We utilize a frequency approximation sketching technique called count sketch, to store user models and evaluate its properties. Such a representation promises both scalability and enhances privacy.

In the context of retrieval, we use similarity approximation techniques called Locality sensitive hashing (LSH). In recent times, LSH [IM98] is proven to be promising in retrieving near-similar items efficiently. LSH is space-efficient, improves runtime and provides approximate retrieval guarantees. This motivates us to improve the time complexity of item retrieval using LSH and make it efficient without compromising much

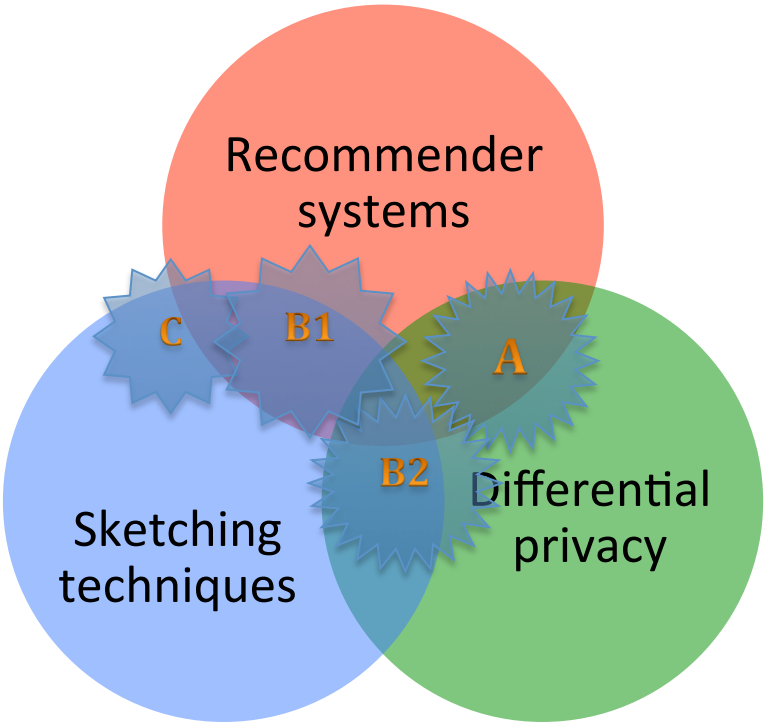

Figure 1.1: Overview of our contributions

on relevance.

1.3 Our Contributions

Our core contributions examines the scalability and privacy properties of sketching technique based recommender systems. The consequence is that the thesis is at an intersection of three different domains: recommender systems, sketching techniques, data privacy. We locate our contributions in the interaction of these three domains pictorially as in figure 1.1. We categorize the contribution into three groups: **contribution A** focuses on privacy aspects of compact-neighborhood based recommender systems, **contribution B** focuses on scalability and privacy aspects of model based recommender systems and **contribution C** focuses on the retrieval aspects. The relation of the contributions on the privacy and scalability aspects is tabulated in table 1.1, with italicized cells corresponding to existing techniques.

Technical background

As our contributions are at an intersection of three different domains: recommender systems, sketching techniques, data privacy; we provide the necessary background in

Contribution	Approach	Aspect	Evaluation	Mechanism
A	Neighborhood	Privacy	Joint decoder	*BLIP, JLT*
B1	Model	Recommendation	RMSE	Count sketch
B2	Model	Privacy	KLD	Count sketch
C	Model	Retrieval	Recall	LSH

Table 1.1: Our contributions on the scalability and privacy aspects

chapter 2 before describing our contributions. We first describe the technical concepts of recommender systems. It is followed by defining and describing the properties of sketching techniques. We then define differential privacy, its properties followed by a discussion on differentially private mechanisms. Our core contributions are in providing both scalable differential privacy mechanisms and estimating the privacy guarantees.

Contribution A

We first focus on the privacy aspects differentially private similarity approximation for recommender systems. We assume the recommender system approach to be neighborhood based and note that efficient differentially private mechanisms has been proposed in the past such as BLIP and JLT pertaining to this neighborhood setup. In chapter 3, we focus on the evaluation problem and asses the privacy guarantees of such differentially private systems. We quantify the amount of information that we can learn from sanitized database by playing the role of the attacker. We first describe a simple attack using single decoder and we improve it by using a more sophisticated joint decoder. We test our attack models on two sketch based differentially private recommender system techniques: BLIP and Johnson-Lindenstrauss transform. We begin the chapter 3 by motivating the problem theoretically and design practical decoders to work on the before mentioned schemes. We also experimentally validate our proposed attack models on public datasets.

Contribution B1 & B2

We then focus on the scalability and privacy aspects of model based recommender systems. We assume a centralized recommender system framework for our model based recommender system. The user representation is based on latent factor modeling and the parameters are learnt through matrix factorization technique. In chapter 4, we describe our matrix factorization approach using a sketching technique called count sketch. We also demonstrate the scalability and dynamicity properties of sketch based factorization through experimental results on benchmark public rating datasets. In particular, our contribution answers the storage scalability challenges faced by model based recommender system, when events arriving in a streaming fashion. We observe

through experimental results that our approach adapts to such new incoming data very well compared to regular factorization. We also observe other desirable properties related to modeling such as generalization capabilities.

We then describe a modification to our system to learn the models in a differentially private setup. We then describe a privacy evaluation metric using Kullback-Leibler divergence that is more relevant to our setup and evaluate it on standard datasets. We observe that our approach enables privacy capabilities implicitly, which is in contrast to conventional privacy preserving methods. Thus our contribution hints strong connection between privacy, ability to generalize and the randomized nature of sketching techniques.

Contribution C

We then address the storage and computational challenges in efficient retrieval of relevant items so that it can scale along well with growing item base. In chapter 5, we improve the retrieval efficiency of latent factor based recommender systems using locality sensitive hashing technique. We describe a family of LSH called cosine sketches which can approximate normalized inner product estimation. We also propose a new technique to improve the estimation efficiency of such cosine sketches. We finally validate our system on synthetic and benchmark real-vector datasets.

Chapter 2

Background

2.1 Introduction

In this chapter, we cover the background material necessary for understanding our contributions. Section 2.2 describes recommender systems in detail. It starts with characterizing recommender system based on factors like data source, approaches and techniques. We then discuss about prediction and evaluation strategies.

The section 2.3 describes sketching techniques in detail. It starts with problem statement and interpretation followed by estimation characteristics. We then discuss some of the common tasks like set membership, frequency approximation, norm approximation and similarity approximation. We also describe the sketching techniques used in our contributions: Bloom filter, count sketch, Johnson Lindenstrauss transform, Locality sensitive hashing and other related techniques. We also survey some of the sketching techniques that were used in recommender systems.

The section 2.4 describes differential privacy in detail starting with problem definition. It is followed by properties of differential privacy such as immunity to side information and composition. We also describe various differential privacy mechanisms such as Laplacian, Exponential and Bayesian. We then describe differentially private recommender systems, characterizing by various factors such as data source, computing model, mechanism and privacy definition.

Finally we conclude in section 2.5 with views on similarity among these three diverse topics, scope for work and position our contributions within the scope.

2.2 Recommender systems

Recommender systems can be characterized by the components that are part of the systems and how these components fit and work together for optimal user modeling and prediction. Some of the common parameters that define a recommender system

are:

- Data source: Nature of input data and its representation
- Approach: The basis on which recommendations are provided
- Modeling technique: Model representation and fitting
- Prediction: Systems output like individual suggestions, list or ranked list

2.2.1 Data Source

As modern recommender systems are intended to model user behavior, the primary data source is naturally the user interaction logs. The user interaction is often the feedback provided by the user in response to a suggested item. If the feedback is explicitly requested after the suggestion, it is known as explicit feedback. In some cases such explicit feedback is not possible, so the consumption of the item is considered as a positive feedback and absence of such information is considered as unknown. Implicit feedback is often domain dependent. Some of the examples for implicit feedback are user click or purchase in product suggestions, play duration in music/video suggestion, etc. Explicit feedback could be of different types ranging from a simple *like/favorite* in social media, to up(down) voting or 5-scale rating and even elaborate review/comment.

Data Representation: The usual representation is a sparse matrix $\mathbf{R}$ with non zero entries $r_{u,i}$ representing user feedback provided by the user $u \in \mathbb{U}$ for a given item $i \in \mathbb{I}$, where $\mathbb{U}$ is set of users and $\mathbb{I}$ is set of items. Each row vector in $\mathbf{R}$ corresponds to an user u and column vector to an item i. In case of implicit feedback, $r_{u,i} \in \{0, 1\}$ and for explicit feedback: $r_{u,i} \in \mathbb{R}$

2.2.2 Approaches

There are various approaches proposed in the past to provide recommendations. They can be broadly classified into content based, collaborative filtering and hybrid approaches.

2.2.2.1 Content based System

In this approach, the characteristics of a given item along with user consumption history is used as a basis to provide recommendations. These characteristics are typically called features and are combined with the given users's preference for these features. The user preference can be asked explicitly or learnt implicitly from his past consumption behavior. If in case the user preference is unavailable, his recent consumed items are used as a query to the system to fetch similar items. A typical use-case is when a user makes a purchase of an item in an e-commerce site in a guest account.

The factors used to represent the given item are often the characteristics of the item. Hence it is primarily a domain dependent approach. As it does not rely on user-generated transaction information, it does not suffer from *cold starting* problem like other approaches. But, characterizing items are usually non-trivial (e.g video recommendations) and not always possible for some domains. Also *cross-domain* recommendation (such as books to video) is a non trivial prediction problem.

2.2.2.2 Collaborative filtering

In this approach, the system utilizes similarity relationship between users and items to provide recommendations. Unlike content based filtering, collaborative filtering utilizes crowd intelligence to provide recommendations. It is achieved by taking into account of the consumption behavior of users similar to the given user and deciding based on collective preference. The relationship can be visualized as a bi-partite graph and is usually represented as a sparse matrix $\mathbf{R}$ with row representing users and column representing items. As it is primarily driven by the user interaction data it requires minimum domain information and can be bootstrapped easily for a new domain.

Collaborative systems is broadly classified based on the way in which the collective preferences are aggregated, as neighborhood based approach and model based approaches.

Neighborhood based: In neighborhood based approaches, the rating matrix is used as it is with minimal preprocessing. The rating matrix is stored as it is in memory and utilized during prediction. It is also known as memory based approach. The neighborhood includes all other elements (user/item) or just the top-n elements as in K-Nearest Neighbor search or a thresholded set of elements as in ϵ-Nearest Neighbor search. These techniques require a similarity measure defined on user-user or item-item. Some of the common similarity measures are Jaccard coefficient, Pearson coefficient and cosine similarity. The similarity measure is often used to weight the contribution of the other elements during aggregation.

Based on the axis of aggregation they can be classified as user-to-user, item-to-item and hybrid approaches. In user-to-user approach, the following sequence of operations is performed.

1. The given user is compared against all other users in the database based on a similarity measure to find neighbors.

2. The neighbors are filtered based on rank or threshold.

3. The items consumed by the given user's neighbors are aggregated. The aggregation can be simple or weighted.

4. The final aggregated set of items are post-processed to get the final prediction set. The post processing could be a ranking based as in top-n prediction or a threshold based filtering.

The item-to-item recommendation is very similar to user-to-user recommendation, with the weighting by item instead of by user and aggregation at user level compared to item level.

Neighborhood based collaborative filtering systems were one of the first successful techniques used to provide recommendations. GroupLens [RIS+94] was one such pioneering system that used user-to-user similarity to predict relevant articles in usenet groups. Similar systems were proposed for other domains such as music [SM95] and video recommendation [HSRF95]. Amazon.com is widely credited for employing neighborhood based recommender system in a large-scale commercial setup [LSY03]. Their system suggested products based on their similarity with other products. An alternative view of item-based collaborative filtering from user based is described in [SKKR01]. Such systems are evaluated in a top-n setup in [DK04] along with effects of normalization. The Netflix recommender systems competition [BL07] served as a catalyst and accelerated many new recommendation techniques.

Model based: In this approach, the rating matrix is used to build statistical models and these models are used during prediction. It also un-necessitates storing the rating matrix after model building. Much of the statistical models comes from machine learning domain. These models are usually compact and hence memory efficient. However model building often requires extensive training data, making it unsuitable for bootstrapping new systems unlike neighborhood based approaches. Some of the common statistical approaches used are data clustering, classification and dimensionality reduction techniques like factor models, Singular Value Decomposition (SVD), topic models and matrix factorization. Section 2.2.5 describes matrix factorization used in our approach in detail.

The success of dimensionality reduction techniques in information retrieval prompted similar endeavors in collaborative filtering. Eigentaste [GRGP01] was one of the first model recommender systems. It used principal component analysis to complete rating matrix. The rating matrix is dense and generated by rating jokes in a web interface. Billsus and Pazzani proposed using SVD in collaborative filtering in [BP98], followed by its success in Latent Semantic Analysis (LSA) of text documents. Applying dimensionality reduction techniques is examined in detail in [SKKR00]. A probabilistic variation of LSA called Probabilistic LSA is proposed in [Hof04] to tackle collaborative filtering. The formulation is not fully Bayesian, leading to difficulties in generalizing to new entrants. It is addressed in Latent Dirichlet Allocation (LDA) [BNJ03] by introducing Dirichlet priors. Model based collaborative filtering includes other diverse approaches such as factor analysis [HH05], Bayesian methods [BHK98], data clustering [UF98] and

restricted Boltzmann machines [SMH07].

2.2.2.3 Hybrid systems

Given that these two methods have their pros and cons as discussed before, it is natural to treat them as subsystems and combine them to get the best of both. These are called hybrid approaches. There are many ways to combine them:

1. Simple aggregation of results from the subsystems
2. Weighted aggregation
3. Using one approach as a preprocessing step to the other
4. Using one approach as a post processing step to the other
5. Integrating one system into the other in a tightly coupled way

2.2.3 Prediction

Given a recommender systems, there are many ways in which one can query it. The most common scenarios are testing whether the given item might interest the user. A lesser common scenario is coming with a ranked subset of items for the given user. It can be also be used to group disparate kinds of items as a single package, like a tourist itinerary.

2.2.4 Evaluation techniques

There are standard measures that can evaluate the prediction performance of recommender systems. The prediction measure depends on the type of the prediction.

For a simple testing of whether the given item will interest the user, the measure starts at atomic rating level and is aggregated for all the ratings. Some of the common measures are root mean square error and mean absolute error.

Root mean square error: The root mean square error materializes the deviation of the predicted rating $\tilde{r}_{u,i}$ from the actual rating $r_{u,i}$. This error is squared and averaged across all non-zero elements of the rating matrix to get mean squared error:

$$RMSE(\mathbf{R}') = \sqrt{\frac{1}{\|\mathbf{R}'\|_0} \sum_{r_{u,i} \in \mathbf{R}'} (\tilde{r}_{u,i} - r_{u,i})^2} \tag{2.1}$$

where $\mathbf{R}'$ is the restriction of $\mathbf{R}$ to the testing set.

Mean absolute error: The mean absolute error materializes the absolute deviation of the predicted rating $\tilde{r}_{u,i}$ from the actual rating $r_{u,i}$, averaged across all non-zero elements of the rating matrix:

$$MAE(\mathbf{R}') = \frac{1}{\|\mathbf{R}'\|_0} \sum_{r_{u,i} \in \mathbf{R}'} |\tilde{r}_{u,i} - r_{u,i}| \tag{2.2}$$

Precision and Recall: For an unordered set of predictions, precision and recall are commonly used as an evaluation measure. Given a set of items as ground truth $\mathbb{I}_u$, precision measures the ratio of relevant items that are retrieved to the actual number of retrieved items $\mathbb{I}'_u$ for a given user. Recall measures the ratio of relevant items to the size of the ground truth set for the given user. Precision and recall at a system level are found by averaging the values across all the users in the system.

$$Precision = \frac{1}{|\mathbb{U}|} \sum_{u \in \mathbb{U}} \frac{|\mathbb{I}'_u \cap \mathbb{I}_u|}{\mathbb{I}'_u} \tag{2.3}$$

$$Recall = \frac{1}{|\mathbb{U}|} \sum_{u \in \mathbb{U}} \frac{|\mathbb{I}'_u \cap \mathbb{I}_u|}{\mathbb{I}_u} \tag{2.4}$$

For a ranked list of prediction, the popular measures are Mean Average Precison (mAP), Normalized Discounted Cumulative Gain (NDCG) and Mean reciprocal rank (MRR).

Mean Average Precision: Mean Average Precision at K (mAP@K) given in (2.5) is defined as mean over the Q profiles in the test dataset of the average of the precisions at rank $1 \leq k \leq K$. The precision(k) refers to the fraction of correct items out of the top k predicted items. The mAP is sensitive to the order of the correct results and is a better gauge of the quality of a ranking.

$$\text{mAP@}K = \frac{1}{Q} \sum_{q=1}^{Q} \left(\frac{1}{K} \sum_{k=1}^{K} \text{precision}_q(k) \right). \tag{2.5}$$

NDCG metric scales the relevance scores of individual items based on its position in the ranked list and normalizes it to the ground truth. MRR calculates the expected reciprocal of the rank of the first relevant item in the list.

2.2.5 Matrix Factorization

Among various model based collaborative filtering techniques, dimensionality reduction techniques is gaining popularity in the recent years. Broadly, the approach consists of factorizing the huge and sparse $\mathbf{R}$ matrix into smaller dense factor models. Some of the popular dimensionality reduction techniques are SVD and variants and matrix factorization.

In latent factor models, each user u is associated with a vector $\mathbf{p}_u \in \mathbb{R}^d$. Similarly, item i is associated to a vector $\mathbf{q}_i \in \mathbb{R}^d$. This allows to approximate the rating $r_{u,i}$.

The latent vectors $\mathbf{p}_u$ and $\mathbf{q}_i$ of all users and items are represented together as $d \times |\mathbb{U}|$ matrix $\mathbf{P}$ and $d \times |\mathbb{I}|$ matrix $\mathbf{Q}$. Matrix factorization assumes a linear factor model and approximates $\mathbf{R}$ by the low rank matrix $\hat{\mathbf{R}} = \mathbf{P}^\top \mathbf{Q}$: rating $r_{u,i}$ is estimated by means of inner product estimation $\hat{r}_{u,i} = \mathbf{p}_u^\top \mathbf{q}_i$. A loss function $L(r_{u,i}, \hat{r}_{u,i})$ quantifies the deviation of the approximation to the observed value. It is often the squared error $\frac{1}{2}(r_{u,i} - \hat{r}_{u,i})^2$ that is used as the loss function. The error is summed across all observed ratings, measuring the aggregate residual error, to be used as objective for optimal approximation [SJ+03]:

$$L(\mathbf{P}, \mathbf{Q}) = \sum_{\text{observed } <u,i,r_{u,i}>} L(r_{u,i}, \mathbf{p}_u^\top \mathbf{q}_i) \tag{2.6}$$

2.2.5.1 Regularization

The objective (2.6), if minimized only on observed data might lead to undesirable results. As the parameter space of the factors are unbounded, it usually leads to large values. This results in poor generalization to unseen data [SJ+03], also called overfitting. Regularization terms controlled by a parameter λ are typically used to tune the model capacity and ensure generalization. We consider the one proposed by Koren et al. [KBV09], which leads to minimization of the combined objective function:

$$\mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) = \sum_{\text{observed } <u,i,r_{u,i}>} L(r_{u,i}, \mathbf{p}_u^\top \mathbf{q}_i) + \frac{\lambda}{2}(\|\mathbf{p}_u\|^2 + \|\mathbf{q}_i\|^2) \tag{2.7}$$

2.2.5.2 Optimization

Techniques to efficiently find the latent factors $(\mathbf{P}, \mathbf{Q})$ minimizers of the objective function $\mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q})$ of (2.7) work either offline or online. Offline techniques such as alternate least squares, gradient descent or online approaches such as the stochastic gradient descent proved to work well. Alternate least squares work by considering the factorization as bi-directional least squares and optimizes by alternating least squares approximation along rows and columns. This is equivalent to do least square approximation on user factor followed by item factors and so on. The convergence is fast, but the least squares approximation is an expensive operation for large datasets.

Gradient descent, a first-order optimization algorithm, works by taking steps proportional to the negative gradient of the objective function (2.7). The proportionality constant, called learning rate, controls the speed of convergence. The usual approach is to sweep the entire training set, before updating the parameters. For large data sets, this is expensive leading to slow convergence. Hence stochastic variants are used which

use random subsets of the original data to take small steps. Though it leads to high variance, the estimation-updates are inexpensive, which leads to faster convergence.

At each step, the stochastic gradient descent [KBV09] randomly picks an observed rating $r_{u,i}$ and optimizes the parameters with respect to that rating. This update relies on the gradient of the loss function with respect to parameters, controlled by the learning rate η:

$$\mathbf{p}_u \leftarrow \mathbf{p}_u - \eta \nabla_u \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) \tag{2.8}$$

$$\mathbf{q}_i \leftarrow \mathbf{q}_i - \eta \nabla_i \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) \tag{2.9}$$

where

$$\nabla_u \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) = \frac{\partial L(r_{u,i}, \hat{r}_{u,i})}{\partial \hat{r}_{u,i}} \frac{\partial \hat{r}_{u,i}}{\partial \mathbf{p}_u} + \lambda \mathbf{p}_u,$$

$$\nabla_i \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) = \frac{\partial L(r_{u,i}, \hat{r}_{u,i})}{\partial \hat{r}_{u,i}} \frac{\partial \hat{r}_{u,i}}{\partial \mathbf{q}_i} + \lambda \mathbf{q}_i.$$

As the algorithm is sequential, only the latent factors have to be stored in main memory. The updates can be parallelized as they are local only to the parameters corresponding to a particular rating. Hence the algorithm can scale well to increasing data size.

2.2.5.3 Experimental protocol

The factorization has two levels of optimization: the low level fitting optimal parameters for the latent factor models and the high level finding optimal hyper parameters. Some of the common hyper parameters are learning rate, regularization constant and model size. Using same training data for optimizing the hyper parameters and the parameters might be optimistic and incorrect. Hence a subset of training data is segregated as validation to fit the hyper parameters. A separate set is used to test the overall performance of the system. To sum up, the given rating matrix is partitioned into three: training, validation and test sets. The usual ratio is $\{80\%, 10\%, 10\%\}$ for training, validation and test sets respectively. The procedure is to train on the training set for different hyper parameters and pick the ones that perform best on the validation set. The hyper parameter search is a linear search on different possible values. If there are more than one hyper parameter, then it is called grid search to search in combinations. Finally, the performance of the system on the test set is reported as the overall system performance.

2.2.5.4 Variants

Matrix factorization, being a popular technique with diverse applications such as information retrieval, signal processing, pattern recognition, visualization, bio-informatics,

etc is well studied with numerous contributions. We emphasis approaches that solve collaborative filtering among these. Early approaches were based in Singular Value Decomposition (SVD), factorizing a large matrix into two orthonormal matrices and a diagonal singular matrix. The input matrix is highly sparse with the rest are missing values and SVD is not very efficient in handling these missing elements. Srebro et al. proposed an efficient factorization technique by optimizing only on observed values [SJ$^+$03]. They also regularized the optimization by means of L_2 norm of the latent factors. [KBV09] further improved it by weighting the regularization of factors based on frequency, as we described before.

Another variant is formulating as a large-margin optimization, where objective is to target low-norm instead of low rank [SRJ04]. It was improved in [RS05, WKS08]. Weimer et al. provided a ranking based framework with maximum margin optimization in [WKLS07].

Matrix factorization is also studied as a parametric estimation problem in a Bayesian setup. Salakhutdinov et al. introduced probabilistic formulation for matrix factorization with advanced complexity control techniques [SM08b]. The Bayesian formulation is further improved in [SM08a, RFGST09]. We describe the Bayesian formulation [SM08b] in the following sub section as one of our contributions is related to it.

2.2.5.5 Bayesian matrix factorization

The bayesian approach assume a probabilistic linear model where in the sparse rating model is decomposed using a linear model with gaussian noise. It is also common to assume a simple Gaussian prior distribution of latent factors:

$$\mathbb{P}[\mathbf{P}, \mathbf{Q}] = \Pi_{u=1}^{\mathbb{U}} \mathbb{P}[\mathbf{p}_u] \cdot \Pi_{i=1}^{\mathbb{I}} \mathbb{P}[\mathbf{q}_i] \tag{2.10}$$

with $\mathbb{P}[\mathbf{p}_u]$ and $\mathbb{P}[\mathbf{q}_i]$ are Gaussian distribution $\mathcal{N}(\mathbf{0}_d, \lambda^{-1}\mathbf{I}_d)$, with $\mathbf{0}_d$ the all zero $d \times 1$ vector, $\mathbf{I}_d$ the identity matrix of size d and $\lambda > 0$. The conditional pdf of the rating knowing the latent factors (*i.e.* the likelihood) is:

$$\mathbb{P}[r_{u,i}|\mathbf{P}, \mathbf{Q}] \cong \mathcal{N}(\mathbf{p}_u^\top \mathbf{q}_i, \nu^2), \text{ with } \nu = 1 \tag{2.11}$$

This implies that, once some ratings are observed, the a posteriori distribution of the latent factors is

$$\mathbb{P}[\mathbf{P}, \mathbf{Q}|_{\text{observed}} < u, i, r_{u,i} >] \propto e^{-\mathcal{R}_\lambda(\mathbf{P},\mathbf{Q})} \tag{2.12}$$

with $\mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q})$ defined in (2.7). Therefore, minimizing this functional as proposed in expression (2.7) amounts to chose $(\mathbf{P}, \mathbf{Q})$ as their MAP (Maximum A Posteriori) estimates [MS07].

An alternative parameter estimation approach is through posterior estimation. The issue then is how to draw according to such a complex a posteriori distribution. Recent

papers show that perturbating the stochastic gradient descent by some Gaussian noise is an efficient way to simulate such a sampling [WT11, VZT15]. This technique is called Stochastic Gradient Langevin Dynamics (SGLD). In our context, this would mean replacing (2.8) and (2.9) by:

$$\mathbf{p}_u \leftarrow \mathbf{p}_u - \eta \nabla_u \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) + \sqrt{\eta} B_P \tag{2.13}$$

$$\mathbf{q}_i \leftarrow \mathbf{q}_i - \eta \nabla_i \mathcal{R}_\lambda(\mathbf{P}, \mathbf{Q}) + \sqrt{\eta} B_Q \tag{2.14}$$

with B_P and $B_Q \sim \mathcal{N}(0, 1)$.

2.3 Sketching techniques

Sketching is an active area of development, particularly in a streaming setup. A data structure maintaining a particular synopsis of the data irrespective of the history of updates can be called a sketch [Cor11]. The space complexity of these techniques is usually sub-linear and the techniques also easy to update and query anytime. These techniques are highly interesting to scenarios where the linear complexity of naively storing is demanding, requiring a sub-linear storage complexity. Some of examples for such cases are user click logs and financial data.

2.3.1 General properties

Streaming setup

As sketching techniques are primarily designed to address statistical queries in streaming data, it is imperative to define the streaming setup.

Let us assume a set of items Ω and for a given item $e \in \Omega$, a quantity associated to the item arrive as a sequence $\{u_{e,t}\}_{t=1}^{T}$: at time t, we receive a quantity related to item e whose value is $u_{e,t}$. If the elements arrive in an order and only once, then it is time-series data and the $u_{e,t}$ is not considered as update. If there is no such order and the elements can repeat, then the stream type depends on the domain of the updates. If the domain of $u_{e,t}$ is non-negative then the stream is called *cash register model*, otherwise as *turn-stile model*.

With such data stream as input, the objective is to answer statistical query q_e related to any item $e \in \Omega$. It is desirable to have a sketch structure $S(k, w)$ that has a redundancy factor k and storage space w such that runtime complexity is $O(k)$ and storage complexity $O(w \times k)$ with $k \ll w \ll N$, where N is the number of observed elements.

Linear transformation model

The reduced storage space is usually achieved by performing a *linear transformation* of the input data before updating the structure. This can be summarized as follows: $S \leftarrow S \oplus P * D$, where S is the sketch structure, P is the linear projection and D is the input data. This property also allows it to compose well, as the sketch of union of dataset subsets can be computed by operating on the sketch of these subsets, without datasets themselves. The projection matrix P can be ranging from a dense matrix to sparse matrix and even sparser matrix defined by a set of hash functions defined on the input.

Estimation Characteristics

The quality of estimation is usually quantified by means of (ϵ, δ) parameters. These allow to understand how close are our approximation to the actual quantity. The most common approximation guarantees are additive, as defined in definition 2.3.1 that measures absolute error and the multiplicative approximation (definition 2.3.2), which measures the relative error.

Definition 2.3.1. *An estimation $\tilde{q}_e$ by Sketching technique T_S is said to be (ϵ, δ) additive approximate to the true quantity q_e, if $\mathbb{P}[|\tilde{q}_e - q_e| > \epsilon] \leq \delta$*

Definition 2.3.2. *An estimation $\tilde{q}_e$ by Sketching technique T_S is said to be (ϵ, δ) multiplicative approximate to the true quantity q_e, if $\mathbb{P}[|\tilde{q}_e - q_e| > \epsilon q_e] \leq \delta$*

Tasks and Techniques

Sketching techniques are in general query focused, designed and built to answer specific query patterns. This factor distinguishes it from other conventional techniques like sampling, that generalizes well. This property allows it to take advantage of the nature of the query to conserve space. Some of the common query patterns and the corresponding techniques are:

- **Set membership**: Bloom filter
- **Frequency approximation**: count min sketch, count sketch
- **Frequency moment approximation**: AMS sketch
- **Similarity approximation**: Johnson Lindenstrauss transformation, Locality sensitive hashing

2.3.2 Sketching for streaming

2.3.2.1 Set membership

The objective is to answer approximate membership queries on a set, with the elements arriving in a streaming fashion. In such setup two types of errors are possible: misreporting presence (false-positive) and missing an item (false-negative). An effective data structure should try to minimize both and improve space efficiency also. The most popular technique, called Bloom filter is described in detail.

2.3.2.2 Set membership with Bloom filter

The Bloom filter [Blo70] is a data structure proposed to check for the presence of elements in a finite set probabilistically. In terms of streaming setup, the update $u_{e,t}$ is the element e itself arriving sequentially over time and the query is to check if $e \in E = \{u_{e,t} | 1 \leq t \leq T\} \subset \Omega$ or not at time T. It consists of a uni-dimensional array B of w bits and a set of k hash functions $\{h_j(\cdot)\}_{j=1}^{k}$. Each hash function $h_j(\cdot)$ yields an index $h_j(e)$ for any item $e \in E$ in the w bit array.

Update: The given element $u_{e,t} = e$ is given as input to the k hash functions $h_j(\cdot)$, to get k indices in the w bit array B. These k bits are then set to 1 irrespective of its previous state: $\forall j, 1 \leq j \leq k, b_{h_j(e)} \leftarrow 1$.

Query: The query q_e is an indicator for the presence of item e in E: $q_e = I_E(e)$. The query element e is given as input to the k hash functions $h_j(\cdot)$ to be used as index in the w bit array B. If all the k bits are *on*, then item is reported to be present in the set, and not otherwise: $\tilde{q}(e) = \wedge(b_{h_j(e)})_{j=1}^{k}$.

Analysis: The k hash functions share the storage of the w bits and this induces collision. The collision leads to false positives, as the k bits for an item might have been set by some of the other N items, even if the given item was not updated. The false positive probability is approximately $\mathbb{P}[\tilde{q}_e = 1 | q_e = 0] \approx (1 - \exp^{\frac{-kN}{w}})^k$, whereas false negativity is $\mathbb{P}[\tilde{q}_e = 0 | q_e = 1] = 0$. It is evident that increasing the storage space or reducing the N decreases the probability of false positivity. Unlike other data structures like linked list or arrays, Bloom filters provide *constant-time* update and query guarantee as it is only a function of k and $k \ll N$.

Variants: Bloom filter was devised by B.H Bloom [Blo70] in 1970 and is used in many domains like networking, caching, search engines, etc. Bloom filter being one of the first-of-a-kind probabilistic data structures, invited many improvements and extensions. One such extension is the ability to maintain count with ($\approx$ 4-bit) counter cells in place of bits. This allows deletions to some extent. It was first introduced in [FCAB00] and improved in [BMP$^+$06], [PSS07] and [RKK14]. Bloomier filter [CKRT04] extends Bloom filter's membership query to an approximate associative array. [DR06] adds dynamicity property to Bloom filter focussing on streaming data. [ABPH07] improves

the scalability of Bloom filter to meet ever-growing demands. Not limiting to these, Bloom filter has inspired many popular sketching techniques solving diverse problems like frequency approximation [Mut05, CCFC02] and norm approximation [DKS10].

2.3.2.3 Frequency approximation

Frequency approximation algorithms aim to estimate frequency of elements, for which the updates arrive in a streaming fashion. The updates could be negative as in turnstile model or strictly non-negative as in cash register model. We describe two such techniques: Count-min sketch and count sketch.

2.3.2.4 Frequency approximation with count-min sketch

Count-min sketch is proposed to approximate frequency information of set of elements arriving in a cash register model [Mut05]. It is represented by a $k \times w$ matrix $\mathbf{C}$ and a set of *pairwise independent* hash functions $\{h_j(\cdot)\}_{j=1}^{k}$.

For a set of elements $e \in \Omega$ and a set of associated quantities v_e to each element e, we observe updates to the element quantities over time; update to element e being $u_{e,t}$ and the update stream being $< u_{e,t} >_{t=1}^{T} \mid u_{e,t} \in \mathbb{N}$. Our objective is to monitor the quantities over the time: $v_e^{(t+1)} = v_e^{(t)} + u_{e,t}$.

Update: Upon the reception of the update $u_{e,t}$ of the e-th quantity, k entries of matrix $\mathbf{C}$ are updated: $\forall j, 1 \leq j \leq k \; c_{j,h_j(e)} \leftarrow c_{j,h_j(e)} + u_{e,t}$.

Query: The query $q_e^{(T)}$ is to return the aggregated updates of a given item e: $q_e^{(T)} = \sum_{t=1}^{T} u_{e,t} = v_e^{(T)}$. At time T, given a query index e, minimum of $\{c_{j,h_j(e)}\}_{j=1}^{k}$ is returned as an approximation of $v_e^{(T)}$: $\tilde{q}_e^{(T)} = \min\{c_{j,h_j(e)}\}_{j=1}^{k}$.

Analysis: The hash indices returned at each row C_j of $\mathbf{C}$ is $\{1, \ldots, w\}$ and $w \ll N$. This induces collision in each row and leads to *over estimation* of the count. Taking minimum of k such rows minimizes such over estimation error: $\tilde{q}_e^{(T)} \geq q_e^{(T)}$. In terms of (ϵ, δ), the space required by the $\mathbf{C}$ array is of $O(\frac{1}{\epsilon} \log \frac{1}{\delta})$, for the estimate to be within $\epsilon : \mathbb{P}[|\tilde{q}_e^{(T)} - q_e^{(T)}| < \epsilon \left\| \mathbf{q}_{-e}^{(T)} \right\|_1] > \delta$, where $\|\cdot\|_1$ is the L_1 norm and $\mathbf{q}_{-e}^{(T)}$ is the vector of all the query elements except e.

2.3.2.5 Frequency approximation with count sketch

Count sketch [CCFC02] is a probabilistic data structure designed to maintain approximations of quantities constantly updated in a turnstile model data stream, but with sub-linear space complexity. It was originally proposed to find *heavy hitters*(high frequency items), but can also be used to approximate the frequencies in turnstile model. Structurally it is very similar to count-min sketch.

For a set of elements $e \in \Omega$ and a set of associated quantities v_e to each element e, we observe updates to the element quantities over time; update to element e being $u_{e,t}$

and the update stream being $< u_{e,t} >_{t=1}^{T} \mid u_{e,t} \in \mathbb{R}$. Our objective is to monitor the quantities over the time: $v_e^{(t+1)} = v_e^{(t)} + u_{e,t}$.

A count sketch is represented by a $k \times w$ matrix $\mathbf{C}$ and two sets of *pairwise independent* hash functions $\{h_j(\cdot), s_j(\cdot)\}_{j=1}^{k}$. The *address* hash function $h_j(\cdot)$ maps an element of Ω to the set $\{1, ..., w\}$ and the *sign* hash function s_j maps an element of Ω to $\{\pm 1\}$.

Update: Upon the reception of the update $u_{e,t}$ of the e-th quantity, k entries of matrix $\mathbf{C}$ are updated: $\forall j, 1 \leq j \leq k$

$$c_{j,h_j(e)} \leftarrow c_{j,h_j(e)} + s_j(e) \cdot u_{e,t}. \tag{2.15}$$

Query: The query $q_e^{(T)}$ is to return the aggregated updates of a given item e: $q_e^{(T)} = \sum_{t=1}^{T} u_{e,t} = v_e^{(T)}$. At time T, given a query index e, mean or median of $\{s_j(e)c_{j,h_j(e)}\}_{j=1}^{k}$ is returned as an approximation of $q_e^{(T)}$. The median operator is more robust to noise [CCFC02] but the mean is easier to compute. In the sequel, we choose the mean operator: $\tilde{q}_e^{(T)} = k^{-1} \sum_{j=1}^{k} s_j(e)c_{j,h_j(e)}$.

Analysis: The update and query run-time complexity are $O(k)$ like count-min sketch, whereas the estimation accuracy is modified by the inclusion of sign-function. For a given (ϵ, δ) and sketch width and depth of $w = O(\epsilon^{-2})$, $k = O(\log \frac{1}{\delta})$ respectively, the estimation accuracy is bounded by $\mathbb{P}[|\tilde{q}_e^{(T)} - q_e^{(T)}| < \epsilon \left\| \mathbf{q}_{-e}^{(T)} \right\|_2] > \delta$, where $\|\cdot\|_2$ is the L_2 norm.

Variants: Count sketch is an influential technique, with extensions solving related problems like norm approximation, parameter representation in machine learning, etc. The hashing technique of count sketch influenced a faster version AMS sketch [AMS96], described in section 2.3.2.7. A special case of count sketch, with $k = 1$, called *Feature hashing* [WDL+09] is used in machine learning applications for efficient parameter representation. The hash based addressing with collisions ensures compact representation of parameters, thereby realizing large-scale learning applications. It also influenced a sparse dimensionality reduction described in [DKS10, KN10]. In [PP13], it is used to approximate polynomial kernel, going by the name *Tensor sketch*.

2.3.2.6 Frequency moment approximation

The objective of frequency moment approximation is to estimate the F_p moment of a set of elements $e \in \Omega$, with updates to the elements arriving in a streaming fashion. Here frequency refers to counting the number of time an element appears in the stream: v_e. In terms of streaming setup, the update $u_{e,t} = 1$ to the element e arrives sequentially over time and the query is to estimate $q_p = \sum_{i=1}^{N} |v_e|^p$, which is a global statistic about the observed elements.

F_p estimation is an interesting problem with diverse applications. An equivalent problem is the L_p norm estimation. We describe the popular AMS Sketch below.

2.3.2.7 Frequency moment approximation with AMS sketch

AMS sketch, one of the earliest known sketches, tries to approximate the F_2 moment by utilizing a set of four-wise independent sign functions and array of counters that are updated based on the sign of hashed element. The original version described in [AMS96] consists of an array $\mathbf{C}$ of size w and w *four-wise* independent sign functions $\{s_j(\cdot)\}_{j=1}^{w}$ mapping from Ω to $\{\pm 1\}$. The estimation is further improved by maintaining k such independent copies of arrays.

Update: For an update $u_{e,t} = 1$ to component e, each element in the array $\mathbf{C}$ is updated by the *sign-transformed* value of the update: $\forall j \in [w]$, $c_j^{(t)} \leftarrow c_j^{(t)} + s_j(e) \cdot u_{e,t}$. If k multiple arrays are maintained, the same process is repeated for each array.

Query: The F_p moment is reported by averaging the p raised entries of the $\mathbf{C}$ array: $\frac{1}{w}\sum_{j=1}^{w} |C_j^{(T)}|^p$. If k independent copies are maintained, then median of these k copies are used to report the final value.

Analysis: In terms of (ϵ, δ), the sketch required $w = \epsilon^{-2}$ sized array with $k = O(\log \frac{1}{\delta})$ repetitions to make sure that $|\tilde{q_2} - q_2| < \epsilon q_2$ with δ probability. Update and query cost are $O(wk)$.

To reduce the update and query complexity, a hash based indexing is introduced similar to count-min and count sketch. This new techniques is referred as fast AMS sketch. It uses an additional set of k *pair-wise* independent hash functions $\{h_j(\cdot)\}_{j=1}^{k}$ mapping from Ω to $[w]$ that finds the array cell to be updated (queried) rather than updating (querying) all w cells. Hence it reduces the update and query complexity by a factor of w without impacting the estimation accuracy.

2.3.3 Sketching for similarity approximation

Similarity computation is central to many domains such as nearest neighbor search, ϵ-neighbor search, manifold learning and visualization. Similarity computation in high dimension is non-trivial due to the curse of dimensionality. There are many indexing techniques that approximate similarity computation in higher dimensions.

2.3.3.1 Similarity approximation with Johnson Lindenstrauss transform

Similarity preserving transforms have a wide range of applications including dimensionality reduction, nearest neighbor search, compressed sensing, visualization, etc. The objective is to come up with a mapping which transforms a d dimensional vector to k dimensional vector such that $k < d$ and similarity among data points are preserved. The Johnson Lindenstrauss lemma forms the foundation for such transforms. It is stated in lemma 2.3.1

Lemma 2.3.1. *For a $\mathbb{R}^d$ space with a set $\mathcal{X}$ of N points, there exists a linear map $f(\cdot) : \mathbb{R}^d \rightarrow \mathbb{R}^k$ with $k = O(\epsilon^{-2}\log N)$, such that for any two vectors $\mathbf{u}, \mathbf{v} \in \mathcal{X}^2$:*

$(1-\epsilon)\,\|\mathbf{u}-\mathbf{v}\| \leq \|f(\mathbf{u})-f(\mathbf{v})\| \leq (1+\epsilon)\,\|\mathbf{u}-\mathbf{v}\|$.

Given the existence of such linear map, the challenge is to find such a linear-map. The linear-map is also called as projection matrix. There are several such known distributions used to prove the lemma 2.3.1 :

- In the source paper [JL84] they proved it using the notion of Lipschitz constant.
- In [FM87] they assumed a random subspace of dimension k and its projection matrix.
- In [IM98] the projection matrix is sampled from 2-stable distribution (Gaussian) for approximating L_2 norm.

Variants: The naive approach requires $k \times d$ operations, which is expensive if it has to be repeated many times. This becomes increasingly evident in streaming cases, where updates to the entries of the d-dimensional vectors arrive as a stream. Hence many contributions were proposed to improve the runtime of the projection. Achlioptas et al. [Ach01] simplified the projection operation by sampling the entries from $\{\pm 1\}$, making it database friendly. Ailon and Chazelle [AC06] improved it by using projection matrix with sparsity defined by a probabilistic distribution. They also pre-condition the vector with a diagonal matrix with $\{\pm 1\}$ entries followed by a Walsh-Hadmard matrix H_d to make it work on sparse inputs also. In [LAS08], Ailon et al. utilized Lean-Walsh matrices to decrease the projection runtime down to $O(d)$.

Another line of approach is to increase the sparsity of the projection matrix, so that update time is reduced. This is desirable in streaming setup, where the entries of input vector arrives as a stream. In [Mat08], they established the sparsity limit of projection matrix achievable by resetting individual entries to zero. Hence Dasgupta et al. [DKS10] proposed an alternative approach using k-wise independent hash functions that directly address dimensions in the low-dimensional vector space. Kane et al. derandomized the technique in [KN10] and improved the sparsity by reducing the collisions among the hash addressing in [KN14].

2.3.3.2 Similarity approximation with locality sensitive hashing

Locality sensitive hashing (LSH) is widely known and successful among similarity approximation techniques. LSH aims to approximate L_p norm by taking advantage of the content directed hash collisions. In LSH, similar items (by normalized correlation) are mapped to same hash code with high probability. In practice, many such hash functions are used to create a compact representation called signature. The similarity computed on these signatures is used as an approximation to the original similarity measure. It is also common to use the hash functions to map elements to *buckets*, thereby indexing elements based on similarity.

Definition 2.3.3. ***Locality sensitive hash functions****: A family of functions defined on a $\mathbb{R}^d$ metric space with distance function $\mathcal{D}$ is called Locality sensitive, if it satisfies the following properties:*
$\forall(\mathbf{u},\mathbf{v}) \in (\mathbb{R}^d)^2$, $R > 0$, $c > 1$

- *if* $\mathcal{D}(\mathbf{u},\mathbf{v}) \leq R$ *then* $\mathbb{P}[h(\mathbf{u}) = h(\mathbf{v})] \geq P_1$
- *if* $\mathcal{D}(\mathbf{u},\mathbf{v}) \geq cR$ *then* $\mathbb{P}[h(\mathbf{u}) = h(\mathbf{v})] \leq P_2$

and $P_1 > P_2$.

There are many LSH functions defined to approximate well-known similarity measures. Of them, we will describe the popular approximation techniques : Minhash that approximates Jaccard similarity, p-stable projection that approximates Euclidean distance based similarity and project-sign that approximates Cosine similarity.

2.3.3.3 Similarity approximation with Min hashing

Min hashing [BCFM98] aims to preserve Jaccard similarity between two sets $\mathcal{A} \subset [N]$ and $\mathcal{B} \subset [N]$: $\mathcal{J}(\mathcal{A},\mathcal{B}) = \frac{|\mathcal{A}\cap\mathcal{B}|}{|\mathcal{A}\cup\mathcal{B}|}$. The technique utilizes a set of k random permuting functions $\{h_j(\cdot) : [N] \to [N]\}_{j=1}^{k}$. The elements of a given set is randomly permuted using the permuting function $h_j(\cdot)$ and the minimum index of the permuted set is retained: $b_j(\mathcal{A}) = \min(\{h_j(e)\}_{e\in\mathcal{A}})$. The probability of two sets two have the same minimum is equivalent to the Jaccard similarity. This property is used to approximate the similarity $\tilde{\mathcal{J}}(\mathcal{A},\mathcal{B})$.

The variance of such technique is high and hence to reduce it, k such permutations are computed to get k minimum indices: $B(\cdot) = [b_j(\cdot)]_{j=1}^{k}$. This vector of k indices is the signature of the set with $k \log N$ size. The similarity approximation $\tilde{\mathcal{J}}(\mathcal{A},\mathcal{B})$ is computed by averaging element-wise *collisions*: $\frac{1}{k}\sum_{j=1}^{k}[b_j(\mathcal{A}) == b_j(\mathcal{B})]$.

Variants: Min hashing was originally proposed to find approximately duplicates in documents using keyword shingles [Bro97]. Since then it has been applied to many other domains with new improvements and extension. The k different hash permutations used in Min hashing is an expensive operation on large item data sets. One simplification is to permute once and use k-minimum indices, improving indexing speed at the cost of accuracy. In restricted min hashing [MS03], the permutation is replaced by a k-wise independent hash function to improve the indexing speed. A notable improvement is b-bit min hashing technique [LK11], in which only b-bits of the min-index is used to reduce the space utilization. Another similar technique called conditional random sampling [LCH06] uses combination of permutation and random sampling to approximate L_1 and L_2 distance on sparse datasets.

2.3.3.4 Similarity approximation with p-stable distributions

In section 2.3.3.1, it is shown that N points in $\mathbb{R}^d$ can be projected to $k(\ll d)$ dimensional vector space. This property is used to map $\mathbb{R}^d$ space points to an integer vector by using k hash functions. The hashing function is defined as $h_j(\cdot) : \mathbb{R}^d \to \mathbb{Z}$ with $h_j(\mathbf{x}) = \lfloor \frac{\mathbf{w}_j \cdot \mathbf{x} + a_j}{b} \rfloor$.

Here a_j is uniformly sampled from $[0, b]$ and $\mathbf{w}_j$ is a d-dimensional vector with entries independently sampled from stable distributions [DIIM04]. A distribution is said to be p-stable, if $\sum_{i=1}^{n} a_i X_i$ of i.i.d variables with $X_i \sim \mathcal{D}$ distribution has the same distribution as $(\sum_{i=1}^{n} |a_i|^p)^{1/p} X$, where $X \sim \mathcal{D}$. As we are interested in L_2 norm, we use gaussian distribution which is a 2-stable distribution. To compare two vectors, their hash code is compared for exactness. To improve the estimation, k such hash functions are used and the average of k such similarities is used as an approximation, similar to section 2.3.3.3.

LSH based on 2-stable distribution, also called as Euclidean LSH, has strong connections with lattice based methods for low-dimensional embedding. Some of the popular approaches are described in [PJA10]. The technique also depends on multiple parameters, making it important to understand their impact and to find the optimal parameter. Stanley et al. studied parameter estimation of p-stable based LSH family in [SLH12].

2.3.3.5 Similarity approximation with cosine sketches

Binary coding LSH aims to approximate cosine similarity of unit-norm vectors in $\mathbb{R}^d$ space. They are also called as cosine sketch [Cha02]. Cosine sketches are usually constructed with random projections, each being defined by a vector $\mathbf{w}_j$, $j = 1 \ldots k$. For any vector $\mathbf{x} \in \mathbf{R}^d$, each projection produces a bit:

$$b_j(\mathbf{x}) = \mathsf{sign}(\mathbf{w}_j^\top \mathbf{x}). \tag{2.16}$$

The sketch of $\mathbf{x}$ is just the concatenation of these bits:

$$\mathbf{b}(\mathbf{x}) = [b_i(\mathbf{x})]_{i=1}^{k} \tag{2.17}$$

Let assume that the projection direction is random and uniformly drawn on the unit sphere. The hyper-plane whose normal vector is $\mathbf{w}_j$ separates two vectors $\mathbf{x}$ and $\mathbf{y}$ with a probability related to the unoriented angle θ between $\mathbf{x}$ and $\mathbf{y}$. This gives the following key property:

$$\mathbb{P}[b_j(\mathbf{x}) \neq b_j(\mathbf{y})] = \frac{\theta}{\pi}. \tag{2.18}$$

The expectation of the Hamming distance between the sketches is also related to this probability if the $\mathbf{w}_j$ are independently drawn: $\mathbb{E}(d_{\mathsf{h}}(\mathbf{b}(\mathbf{x}), \mathbf{b}(\mathbf{y}))) = k\mathbb{P}[b_j(\mathbf{x}) \neq b_j(\mathbf{y})]$. Therefore, the Hamming distance gives an unbiased estimator of the angle as

$$\hat{\theta} = \frac{\pi}{k} d_{\mathsf{h}}(\mathbf{b}(\mathbf{x}), \mathbf{b}(\mathbf{y})). \tag{2.19}$$

The cosine function is decreasing over the range of the unoriented angle $[0, \pi]$. Therefore, ranking vectors by increasing order of the Hamming distance between their sketches and the sketch of a query approximates the ranking by increasing angle or decreasing cosine similarity.

Several researchers have proposed extensions to this initial framework, *e.g.*, by proposing other kernel estimations [RR07a, WTF09a, RL10, GL11]. Note that this approach has been introduced in different communities: For instance, spectral hashing [WTF09a], universal quantizer [Bou12], and ℓ_2 binary sketches [DCL08] are very similar. These constructions are designed for the ℓ_2 distance and share the idea of cyclic quantization. The same idea is considered and analyzed in a compressive sensing framework [Bou12].

2.3.4 Sketching techniques for machine learning

Application of sketching techniques for machine learning is not a new topic, with numerous previous works. We highlight some of the works, which align well with the context of the thesis. Random projection is well known as a preprocessing technique applied on the input data, that is usually followed by supervised learning. In [WDL$^+$09], the authors use hashing to map feature space into a fixed-width representation to reduce the sparsity of input.In [LSMK11], the authors use a variant of min hashing to reduce the feature space, and integrate it with popular linear supervised learning algorithms. Feature maps that approximate kernel matrices are another recent active research area and some of the popular approaches are [RR07b, PP13].

On the other hand machine learning has improved the optimality of sketches such as compact binary codes [SH09, NB11, WTF09b, KD09b]. A complete summary of such compact binary code techniques is presented in [TFW08b]. Also in [Lib13], SVD is used to efficiently find item frequencies using the notion of "Frequent directions".

2.3.5 Sketching techniques for recommender systems

When it comes to recommender systems, sketching techniques are often used to represent user profiles. In case of memory based systems, the user profile is nothing but a function of the set of items consumed. There are many ways to approximate such a set. In [AGK12] the authors used Bloom filter to approximate such a set and exchanged with other users to compute similarity. LSH is another suitable candidate to perform efficient similarity search. A variant of LSH is adapted to measure proportional intersection in collaborative systems in [BPR09]. MinHash, an instance of LSH is shown to improve the scalability of collaborative filtering in [DDGR07]. A variant of minwise hashing called b-bit minwise hashing is incorporated into linear learning algorithms in [LSMK11]. In [KKMM12] they used Johnson-Lindenstrauss transform to reduce the sparse item-set to compact dense low-dimensional real vector. In [MDDC15] they used

count sketch to succinctly represent co-occurrence counts that can be used to identify similar profiles.

When it comes to model based systems, the model is often latent factors, which are themselves space-efficient. Still there were attempts to take advantage of the compactness property of sketches to further-efficiently represent these factors. Feature hashing [WDL$^+$09] is applied to matrix factorization in [KWS10]. One of our contributions is well related to this work as count sketch is a generalization of their hash sketch. Although LSH has a compact representation, it is a discrete structure and not suitable for continuous optimization problems like matrix factorization. Still there are existing works such as [ZZ12, WWY15, LHDL14] that utilize compact binary codes for model representation.

2.4 Differential privacy

We choose differential privacy [Dwo06a] as the data sanitization criterion. Differential privacy argues that absolute privacy is impossible to achieve and instead settles to relative level. It was originally devised to answer queries on databases without compromising on row level privacy. Since then it has garnered wide spread attention among the research community for its theoretical rigorousness and robustness to auxiliary information. The privacy objective is achieved by means of a randomizing mechanism, which takes the database as input and perturbes the output such that it is not differentiable, to an extent, from its neighboring databases that differs by a row.

2.4.1 Problem definition

Definition 2.4.1. *A randomized algorithm $\mathcal{A}$ is ϵ-differentially private if for all pairs of neighboring databases $\mathcal{D}, \mathcal{D}'$ that differs by 1 row ($|\mathcal{D} - \mathcal{D}'|_1 = 1$) and all $s \subseteq Range(\mathcal{A})$*

$$\mathbb{P}[\mathcal{A}(\mathcal{D}) = s] \leq e^{\epsilon}\mathbb{P}[\mathcal{A}(\mathcal{D}') = s] \tag{2.20}$$

A weaker variant of ϵ-differentially private is the (ϵ, δ)-differentially private [DKM$^+$06], can be seen as a probabilistic variant in which the guarantees of differential privacy hold with probability of $1 - \delta$. It is stated as follows:

Definition 2.4.2. *A randomized algorithm $\mathcal{A}$ is (ϵ, δ)-differentially private if for all pairs of neighboring databases $\mathcal{D}, \mathcal{D}'$ that differs by 1 row ($|\mathcal{D} - \mathcal{D}'|_1 = 1$) and all $s \subseteq Range(A)$*

$$\mathbb{P}[\mathcal{A}(\mathcal{D}) = s] \leq e^{\epsilon}\mathbb{P}[\mathcal{A}(\mathcal{D}') = s] + \delta \tag{2.21}$$

The promise of differential privacy is that any individual will not have additional impact, whether he decides to contribute in the released data or not. This is different from conventional techniques, where the promise is that individuals will not be

affected at all. The argument is that any individual cannot control the outcome of prediction even if he refuses to contribute. Hence absolute privacy is impossible to achieve. Instead, preserving the uncertainty about the contribution of the individual in a given database is more realistic. It not only mitigates *re-identification* risks but also encourages more people to contribute to the data release.

2.4.2 Post processing

The second promise of differential privacy is its immunity to auxiliary information, thanks to the randomization factor. This can be formally defined as:

Property 2.4.1. *Given a (ϵ, δ)-differentially private algorithm $\mathcal{A}$: $\mathcal{D} \to \mathbb{R}$, let f: $\mathbb{R} \to \mathbb{R}$ be any arbitrary randomized mapping. Then the composition $f \circ \mathcal{A}$ is (ϵ, δ)-differentially private.*

It suggests that any post processing of the outcome of a (ϵ, δ)-differentially private algorithm by any other mechanism will not reduce the (ϵ, δ) bounds.

2.4.3 Sequential composition

A desirable property of differential privacy is composability. Often the data processing pipeline is composed of many subsystems and each such sub-systems might require different level of access to database. Therefore it is imperative to talk about differential privacy guarantees at a system level, rather than at the granular subsystem level. This necessitates understanding how these heterogenous differentially private algorithms compose together and what is the *information leakage* when considering the whole system. Thankfully the (ϵ, δ) of the composition of these differentially private algorithms are simple enough to be just sum of (ϵ, δ) of individual algorithms, when queried on the same database. It is defined as follows:

Property 2.4.2. *Given set of algorithms $\{\mathcal{A}_i : \mathcal{D} \to \mathbb{R}\}_{i=1}^n$ that provides (ϵ_i, δ_i)- differential privacy, then composing them such that $\mathcal{A}_\times(\cdot) = (\mathcal{A}_i(\cdot))_{i=1}^n$ provides $(\sum_{i=1}^n \epsilon_i, \sum_{i=1}^n \delta_i)$ differential privacy.*

Composition also helps in defining the *privacy budget* of the system and deciding how to allocate or limit privacy leaks at various stages.

2.4.4 Computation model

The computation model in general could be interactive or non-interactive. Interactive mechanisms perturbate the output for a given statistical query and are done independently for each such queries. Non-interactive mechanisms, on the other hand, randomize

the output once-for-all and release the data, which will be used for all future queries. Interactive mechanisms can tune the perturbation to a given query and hence can improve the utility of the system, without compromising privacy, compared to non-interactive mechanisms. But, answering successive queries from the same database means diluting the privacy guarantees.

2.4.5 Variants

The ϵ and (ϵ, δ) form of differential privacy, though answering majority of privacy concerns, still leaves some questions unanswered. One concern is the data security. What happens if there is a security breach and someone could access the *internal state* of the system. *Pan privacy* aims to tackle such a scenario. Its objective is to maintain a differentially private internal state such that is robust to both inference and security attacks.

Another strong requirement is maintaining differential privacy under continuous observation. A typical use case is a streaming data setup, wherein incremental data arrives anytime and the system can be queried anytime in between.

A third variant is local privacy, which guarantees privacy at user level. The conventional techniques make assumptions about the trusted curator, who manages the data release. *Local privacy* makes minimal assumptions about such trusted third-parties and randomizes the response at user level to achieve stronger privacy [KLN$^+$11].

The (ϵ, δ) version is weaker than ϵ but still becomes unrealistic for certain scenarios like machine learning systems. Even hard-to-infer statistics used by such learning systems have very high ϵ values, which gives an impression of privacy dilution. Concentrated differential privacy [DR16] aims to tackle such use-cases. An algorithm is said to be (μ, τ)-Concentrated differentially private if the privacy loss is centered around at μ and the probability of the loss exceeding μ is τ sub-gaussian distributed. It is primarily tailored to the cases where there are repeated computations of the same variable, a case where the usual (ϵ, δ) measure is unrealistic.

2.4.6 Mechanisms

As such, differential privacy is a measure and not a mechanism by itself. There are many mechanisms that are proposed in the past that can achieve differential privacy including randomized response, Laplacian mechanism [Dwo06b], exponential mechanism [MT07], Bayesian inference [WFS15], smooth sensitivity and sample-aggregate frameworks [NRS07].

2.4.6.1 Laplacian mechanism

Laplacian mechanism achieves ϵ-differential privacy by adding Laplacian noise to the output. The amount of noise added depends on the properties of data, in particular the sensitivity of the data. The L_1-sensitivity of a function f, δf_{L_1}, is defined as the maximum variation in the function output for all 1-row neighboring databases. It is shown in [Dwo06b] that adding noise sampled from $Lap(\epsilon^{-1}\delta f_{L_1})$ distribution to the function output is ϵ-differentially private. Laplacian mechanism is generic and most popular of all other mechanisms.

2.4.6.2 Exponential mechanism

The Laplacian mechanism only cares about the L_1-sensitivity and is independent of the actual output of the function. It is desirable in some cases to *selectively prefer* sub-ranges of the function outputs. Exponential mechanism [MT07] caters to these cases and is shown to be superior to Laplacian mechanism. It is primarily designed to privately conduct auctions with revenue maximizing objective. The preference is usually expressed by means of a utility function $u(\cdot,\cdot) : (\mathcal{D}, \mathbb{R}) \to \mathbb{R}$ which takes both the database $\mathcal{D}$ and an output r and returns a utility score $u(\mathcal{D}, r)$. Given such a utility function with L_1 sensitivity, δu, it is shown in [MT07] that an algorithm that outputs $\mathcal{A}(\mathcal{D}) = r \in \mathbb{R}$ where $\mathbb{P}[\mathcal{A}(\mathcal{D}) = r] \propto \exp^{\frac{\epsilon u(\mathcal{D},r)}{2\delta u}}$ is ϵ-differentially private.

2.4.6.3 Bayesian inference

A third class of mechanism is Bayesian sampling. It is primarily used in a machine learning setup, where the parameter estimation is performed in a Bayesian setup. It was observed in [WFS15] that performing sampling on posterior distribution is differentially private to an extent. It is notable that this privacy comes as *free*, without any special effort. It is shown in [WFS15] that if the log-likelihood sensitivity is bounded by B: $sup_{\mathcal{D};\theta\in\Theta}|\log \mathbb{P}[\mathcal{D} \mid \theta]| \leq B$, then one sampling according to the posterior $\mathbb{P}[\theta \mid \mathcal{D}]$ is $4B$-differentially private, irrespective of the prior. In practice, $4B$ could be very large making the privacy level unusable. In [WFS15], they also propose a mechanism to scale the likelihood by $\max(\frac{\epsilon}{4B}, 1)$ to make it ϵ-differentially private. When the epsilon is very low, the likelihood scaling makes posterior distribution smoother.

The challenge is then on finding an efficient approximate sampling algorithm as exact sampling on large sets is intractable. One such approximate sampling is Stochastic Gradient Langevin Dynamics as described in section 2.2.5.5

2.4.7 Attack models

Even though the characteristics of perturbation based methods are well defined, a study from adversarial stance is essential to assess the information leakage. The attack

models are often probabilistic with optional auxiliary information. The complexity of such models increase with the assumptions these models make and improves the information gain often limited by the theoretical guarantees of the techniques.

A detailed survey on inference attacks on sanitized data can be found in [LGK08] and [CL08]. Common attacks include eigen-analysis [GW06, HDC05], MAP estimation [HDC05], Independent Component Analysis (ICA) [GW07] and distribution analysis [AA01]. MAP estimation and ICA make direct assumptions on the distribution of the original data, whereas distribution analysis and our approach estimate it from publicly available information. In addition, eigen-analysis makes even stronger assumptions on the representation of data and thus is not generic enough to apply to all representations. Furthermore, the possibility of using probabilistic inference techniques to attack sanitized histogram data has been illustrated in [DS98] and [Dob00]. In these works, bounds of records count are estimated from histogram of attributes coming from a Markov Chain Monte Carlo (MCMC) simulation. Application of probabilistic inference techniques for parameter estimation on differentially private data is illustrated in [WM10]. In this work, the authors have also experimentally validated their approach using MCMC on parameter estimation of logistic regression and probabilistic inference of principal components. Although their objective was not directly the reconstruction of data, their approach demonstrates that probabilistic inference is possible on differentially private data.

2.4.8 Differential privacy and machine learning

Since its introduction, Differential privacy has found applications beyond answering statistical queries including machine learning [CMS11]. Applying Differential privacy to machine learning is an active topic of research with many notable work. The works differ on the stage in which the randomization mechanism is added to the system. Initial approaches were advocating at the input and output level. In [CMS11], they advocated incorporating at the optimization level and demonstrated its superiority over output level perturbation. Differential privacy is also applied to online learning at [JKT11].

A detailed survey of different techniques is available at [SC13, Dwo08, JLE14]. [FS10] applied differential privacy to data mining. [KLN$^+$11] studied the theoretical properties of learning under differential privacy setup. [WFS15] proved that Bayesian posterior sampling is differentially private to an extent.

As for matrix factorization, [BFK$^+$15] compares different ways to ensure differential privacy, among them, the Laplace mechanism on the inputs or on the updates of the stochastic gradient descent. In [KT13],they proposed a differentially private low rank approximation using exponential mechanism. In [LWS15], they use Bayesian posterior sampling to perform differentially private matrix factorization.

2.4.9 Differentially private recommender systems

Recommender systems are based on personal data which are sensitive information. Simple recommender systems depend on just basic information like demographics and personal preferences. Advanced systems depend additionally on dynamic and incremental data sources like item consumption and feedback, which are even critical. This requires efficient privacy preserving mechanisms that can hide both item consumption and feedback information from user. Such privacy preserving mechanism can be characterized by the privacy measure, the kind of mechanism, the computing model and the privacy level. We are primarily interested in recommender systems based on differential privacy measure.

2.4.9.1 Mechanism

The most common differentially private mechanism used in recommender systems is Laplacian. In case of neighborhood based systems, this involves adding Laplacian noise to co-occurrence counts [MM09]. It is also used in model based systems such as matrix factorization [BFK$^+$15]. A recent innovation is utilizing Bayesian mechanism for efficient differentially private learning. It involves Bayesian sampling on latent factor posterior distribution to make the learning differentially private [LWS15].

2.4.9.2 Computing model

The computing model in privacy-aware recommender systems can be classified by the nature of the server: trusted server or untrusted server. In the trusted server, a centralized server does most of the computations leaving only presentation and feedback to the clients. This means that the data is stored at this server and released in a responsible way, such that information leakage is minimized. This also means that perturbation/encryption happens at this centralized server. It is assumed that such centralized service provider is not malicious and won't collude with other malicious third-parties. In the untrusted server case, no such guarantees are assumed. Hence noise is added at the input source level by individual users leading to local privacy, a stronger form of differential privacy.

2.4.9.3 Privacy level

The privacy can be defined at different levels: event level and user level. The event level is simpler of these two. The mechanism aims to assure that releasing the database does not change the odds of associating the user with an event. These events could be implicit item consumption or explicit feedback. These events are sometimes accompanied by contextual information like geo-location, timestamp, referral, platform, which are sensitive too. An even harder privacy assurance is at the user level [LWS15]. Here

the objective is to obscure that the user contributed to the database at all. One way to approach user level privacy is the aggregation of all events associated with the user.

2.4.9.4 Data source

These mechanisms also differ by the type of data that it uses (described in 2.2.1). We describe two common data sources: binary data representing item consumption and real data representing item rating.

Binary data: Mechanisms involving binary data are simple as the objective is to hide just the item presence information. Hence the differential privacy definition is similar to the one defined in section 2.4. In this case, a user profile is just a set of all items that he consumed in the past.

Definition 2.4.3. ***Differential privacy for user profile****: A randomized function* $\mathcal{F} : \mathcal{D}^n \rightarrow \mathcal{D}^n$ *is* ϵ*-differentially private, if for all neighboring profiles* $\mathbf{x}, \mathbf{x}' \in \mathcal{D}^n$ *and for all* $\mathbf{t} \in \mathcal{D}^n$*:*

$$\mathbb{P}[\mathcal{F}(\mathbf{x}) = \mathbf{t}] \leqslant e^{\epsilon} \cdot \mathbb{P}[\mathcal{F}(\mathbf{x}') = \mathbf{t}] \ .$$

This probability is taken over all the coin tosses of $\mathcal{F}$ *and* e *is the base of the natural logarithm.*

Two profiles $\mathbf{x}$ and $\mathbf{x}'$ are said to *differ in at most one element* or said to be *neighbors* if they are equal except for possibly one entry.

Real data: Mechanism that involve explicit feedback are more complex as the data source (*i.e.* ratings) is usually in $\mathbb{R}$. Let us consider the case of a trusted recommendation system with rating(event) level privacy. In this case, differential privacy convinces users to submit their ratings by showing that an attacker querying the recommendation system has difficulty in deciding whether a particular rating has been used by the system to perform prediction. Even with the side-information that user u rates item i by the true value $r_{u,i}$, the attacker cannot say whether the user submitted or not this information.

Denote by $\mathcal{D}$ the dataset of observed ratings used training the system, and $\mathcal{D}' = \mathcal{D} \cup \{< u', i', r_{u',i'} >\}$ s.t. these two datasets differs by one rating. Denote by $\hat{r}_{u,i}(\mathcal{D})$ the output of a recommendation system when trained on dataset $\mathcal{D}$ and queried about user u and item i.

Definition 2.4.4. *A trusted recommender system with rating level privacy is* ϵ*-differentially private, if* $\forall (a, b) \in \mathbf{R}^2, a < b, \forall u \in \mathbb{U}, \forall i \in \mathbb{I}$

$$\mathbb{P}[a < \hat{r}_{u,i}(\mathcal{D}) < b] \leq e^{\epsilon} \mathbb{P}[a < \hat{r}_{u,i}(\mathcal{D}') < b]. \tag{2.22}$$

2.4.10 Differentially private mechanisms using sketching techniques

Differential privacy enabling mechanisms depend on external source of randomness to generate the required noise. The usual methodology is to separate the computation from noise generation and mixing them at a point. Introducing sketching techniques changes the procedure significantly. Sketching techniques are also randomized by design, making it a good source of random noise. The approximate answering of sketching techniques resemble differentially private techniques albeit non-compact representations. Also if the output is a sketch, the noise can be added to the sketch directly. Hence the synergy of sketching techniques and differentially private mechanisms is irrefutable.

2.4.11 Differentially private recommender systems using sketching techniques

When it comes to recommender systems, sketching techniques are often used to represent user profiles. The sketch structure helps in selectively masking identifying individual consumed items, but facilitating efficient aggregation at the same time.

A centralized recommender system through anonymized channel is proposed by the Alcatel-Lucent Bell labs team [NAB11, ABK$^+$13]. In this system, the user feedback and request are routed through anonymous channel called *Tor* network and recommendations are aggregated among *like-minded* groups. Identification of the group and addressing is made possible with the help of (LSH) [IM98]. Though the scheme preserves privacy by means of closed-group aggregation, the availability of uncorrupted information among theses closed groups makes it vulnerable to linkage attacks. Also the privacy guarantees are hard to control and measure. Nevertheless, the proposal is more of a framework, leaving room for differentially private data aggregation within the discovered groups.

We introduce two existing techniques that are evaluated by our joint-decoder attack model, described in Chapter 3. In [AGK12] they used Bloom filter to approximate the set of consumed items and exchanged with other users to compute similarity. They also perturbate the bits of the Bloom filter before publishing to enforce ϵ-differential privacy. The perturbation is achieved by flipping the bits of Bloom filter using noise bits sampled from Bernoulli distribution. The technique is called BLIP (*BLoom filter and FLIP*). Another recent approach [KKMM12] utilizes Johnson-Lindenstrauss transform to reduce the sparse item-set to compact dense low-dimensional real vector. Then they perturbate the profile vector using random noise sampled from Gaussian distribution. This technique guarantees (ϵ, δ)-differential privacy. They also compare it against the classical randomized response technique [War65].

In [MDDC15], they proposed a recommender system by aggregating co-occurrence count in a privacy preserving way. They utilize count sketch to aggregate the frequency information and perturb to enforce differential privacy. [MS06] used sketch to provide

privacy. Their notion of privacy is very similar to differential privacy except that the loglikelihood ratio is bounded by a linear parameter instead of exponential.

2.5 Conclusion

In this chapter, we described the background necessary to understand our contributions towards using sketching techniques for building scalable and privacy-aware recommender systems. As we see, differential privacy and sketching techniques have many things in common: inexactness, approximations, query-focussed. Yet the intersection of differential privacy and sketching techniques is not given its due importance. We emphasize the connections in the following chapters through our contributions, so that it will be evident that differentially private recommender systems with sketching techniques are realizable. Sketching techniques also come with general benefits like scalability and problem-specific benefits like regularization. We also study these benefits through experimental analysis.

We also observe that differential privacy and model based recommender systems using machine learning techniques has complementary objectives. While differential privacy strives to mask identification of individual data, machine learning strives to *generalize* from such a data source. The connection is even strongly evident from regularization point of view. We demonstrate it in the following chapters. We mainly touch upon the intersection of differential privacy and machine learning and sketching techniques.

Chapter 3

Challenging differentially private compact user modeling mechanisms

Outline

In this chapter, we focus on existing techniques that enable privacy-friendly compact representation of user profiles using sketching techniques. These compact representations are used to perform neighborhood based collaborative filtering systems in a scalable way. In these techniques the profile of a user is sanitized by a non-interactive differentially private mechanism before publication.

We consider two existing schemes offering a differentially private representation of profiles: BLIP (BLoom-and-flIP) and JLT (Johnson-Lindenstrauss Transform), described in sections 3.1 and 3.2. For assessing their security levels, we play the role of an adversary aiming at reconstructing a user profile. We compare two inference attacks, namely *single* and *joint* decoding. The first one decides of the presence of a single item in the profile, and sequentially explores all the item set. The latter strategy decides whether a subset of items is likely to be the user profile, and considers all the possible subsets.

Our theoretical analysis shows that joint decoding is more powerful than single decoding. We also propose a joint decoding method based on the Monte-Carlo Markov Chain algorithm (MCMC). The theoretical analysis is described in 3.3 and a practical implementation is proposed in section 3.4. We tested experimentally the validity of our approach on datasets composed of real user profiles. We describe our experimental setup and the results

in section 3.5. The results obtained demonstrates the superiority of joint decoding, while also giving useful insights on how to set the differential privacy parameter ϵ.

Neighborhood based collaborative filtering systems are one of the promising methods in recommender systems. These systems produce relevant suggestions based on user behavior by finding similar users and aggregating their suggestions. A technical description of neighborhood methods were available in section 2.2.2.2. Finding similar users require computing some kind of pairwise similarity between the profiles of different users. Some of the challenges that such systems face include privacy and scalability issues.

The standard neighborhood based methods naively represent the user as a sparse vector, with non-zero elements representing the items consumed by the given user. With ever-increasing number of users and items, this naive representation pose scalability challenges. The storage of user profile as sparse vectors turns to be expensive with growing item consumption. Also comparing such user profiles require sparse vector similarity estimations which proves to be costly with increasing user database size. Sketching techniques comes to rescue as an efficient solution to compactly represent the user profiles. Sketch structures are simple and compact by setup and facilitate efficient insert, update and comparison operations. A detailed technical discussion of sketching techniques was presented in section 2.3. The inherent randomness of sketch structure provide privacy to an extent but is not well quantified and might be inadequate.

The inadequacy is a serious threat for the sensitive information of the profile database and some users may even refuse to participate if they have no guarantees on the privacy of their profiles. The privacy concern can be mitigated by using a more stronger privacy preserving notion such as differential privacy. Differential privacy is originally proposed to ensure row level privacy in a database. The concept of differential privacy and related mechanisms were described in section 2.4. In our setting, the input of the computation is the profile of a user and the randomized output will be a perturbed version of a compact representation of this profile (*e.g.*, a Bloom filter or a random projection). The popular differentially private mechanisms enforce privacy by randomizing the output of this computation, and this independently of the auxiliary information that the adversary might have gathered. Differential privacy definition pertaining to user profile was presented in section 2.4.9.4.

One of the usual limits of differential privacy is that each time a differentially private computation takes place, the user loses a little bit of privacy (as measured by the value of the privacy parameter ϵ). Therefore, if this computation takes place too many times, the user may spend all his privacy budget and remains with no privacy left. The adversary is then able to reconstruct almost entirely the user's profile. These mechanisms are called *interactive*. Interactive mechanisms require a two-way communication protocol

between the curator (the entity in charge of the database) and the client performing the query. Therefore, the curator has to be online in order to receive the query and prepare the associate response to this query. However, if the system is dynamic there might be no upper bound on the maximum number of similarity computations that can occur and therefore an interactive mechanism would be of limited applicability.

On the other hand, a *non-interactive* mechanism computes some function from the original database and releases it once and for all, which corresponds to a one-way communication protocol. The output released by the non-interactive mechanism can later be used by anyone to compute the answer to a particular class of queries (usually not just a single specific query), without requiring any further interactions with the curator. It is important to understand that the answer is computed from the output released by the non-interactive mechanism, thus after publishing this output the curator can go offline. One particular type of non-interactive mechanism is the *generation of a synthetic dataset* that allows the answer to certain class of queries (but not necessarily all) to be approximated. Examples of non-interactive mechanisms for differential privacy include [BNO08, LZWY11].

For these reasons, a scalable and privacy friendly approach is to compactly represent the user profile using sketching technique and sanitize the profile of a user with a non-interactive mechanism compliant with the concept of differential privacy before his publication. Hence, we are interested in two non-interactive mechanisms offering a compact differentially private representation of profiles: BLIP (BLoom-and-flIP) [AGK12] and JLT (Johnson-Lindenstrauss Transform) [KKMM12]. Both schemes have been designed to protect the privacy of the profile while still enabling efficient computations of distances and similarities on the differentially private representation of the profiles.

The privacy level offered by differential privacy is quantified by the parameter ϵ, which shares an inverse relationship with the amount of privacy offered: A low ϵ is equivalent to high privacy level and vice versa. Though differential privacy seals an upper bound on the maximum amount of *information leakage*, it does not thoroughly describe the information gain from an adversarial stance. This raises a question on the amount of information gained by performing inference attacks on the publicly released user profiles. In specific, we are interested in describing the privacy level by means of *mutual information* between the sanitized user profile and the *original* user profile. The mutual information point of view provides the necessary theoretical motivation to devise efficient inference attacks on the observed profile to predict and reconstruct the original user profile. We start with a simple inference attack called *single decoder* and then propose an advanced inference attack called *joint decoder*. The single decoder reconstructs the profile by testing items individually where as the joint decoder tests group of items. We analytically find that the mutual information present in a group-tested item-set is higher than testing individual items.

Our approach is inspired by a problem appearing in several applications such as in

group testing [MF11, KST96], traitor tracing [FGC12, MF12], multiple access channels, or compressed sensing. The problem is described in general terms as follows. We observe a vector which is the aggregation of some reference patterns plus some noise. These patterns are called codewords in traitor tracing or atoms in compressed sensing. They can be binary (group testing, traitor tracing) or real vectors (multiple access channel, compressed sensing). The set of all reference patterns is publicly known, however we don't know which subset was used to produce the observation. The aggregation process is also known (except in traitor tracing and compressed sensing). It is either a component-wise process, which is deterministic (like a sum or a XOR) or probabilistic (components are randomly selected) depending on the application. The observation is a noisy version of the aggregation. Again, the distribution of the noise is assumed to be known. The goal is to infer which reference patterns were used to create the observation. In these fields of application, it has been proven that jointly measuring the likelihood of a subset of reference patterns being at the root of the observation is theoretically more reliable than measuring the likelihood of a single pattern at a time.

Though the theoretical justifications favor joint decoder in place of single decoder, the combinatorial cost of testing all possible groups out of a set items grows exponentially, making it impractical. This computational bottleneck is not unique to our decoding problem, but common among all Bayesian estimation problems. Efficient approximation techniques exists such as *Variational inference* and *Markov Chain Monte Carlo (MCMC)* techniques. We use *Markov Chain Monte Carlo* technique to approximate the estimation and use it to propose a practical decoder owing to its practicality and popularity.

Together with single decoder, these two practical decoders are used perform inference attack that help to assess the privacy guarantee provided by the BLIP and JLT mechanisms. We also provide an analysis of the utility and the protection offered by BLIP and JLT against these attacks, by deriving theoretical bounds on the resulting approximation error generated by a specific value of the privacy parameter. Furthermore, we evaluate experimentally the trade-off between privacy and utility achieved by these mechanisms. These attacks helps to better understand the privacy guarantees offered by a differentially-private mechanism, while also enabling the privacy practitioner to tune ϵ experimentally.

In the next sections, we describe two non-interactive mechanisms that have recently been proposed. The first mechanism is based on randomizing a Bloom filter representation of the profile [AGK12] while the second relies on the application of the Johnson-Lindenstrauss transform and the addition of noise [KKMM12]. Both mechanisms preserve some global properties such as the ability to compute a distance between two profiles while hiding the details of the profiles themselves.

3.1 BLIP

The main objective of BLIP [AGK12] is to prevent the adversary from learning the presence (or absence) of an item in the profile of a user by observing the Bloom filter representation of this profile. Our theoretical analysis provided in Section 3.3 is based on the model of profiles and the BLIP sanitization described thereafter.

3.1.1 Setup of BLIP

The setup that we consider for the theoretical analysis is the following. We assume that a profile P is a list of c items randomly picked from a set of $N \in \mathbb{N}^\star$ possible items: $P = \{j_1, \ldots, j_c\}$. We denote the set of items by $[N]$, with $[N] \triangleq \{1, \ldots, N\}$ and the set of all possible profiles by $\mathcal{P}$. This set is a subset of the power set of $[N]$ and we have $|\mathcal{P}| = \binom{N}{c}$. For the moment, we make the assumption that c is publicly known, but this hypothesis will be lifted later by inferring this value directly from the Bloom filter.

The profile is first encoded in the form of a Bloom filter, which is a binary string of L bits. Each item $j \in P$ is hashed through K different hash functions $(h_1, \ldots, h_K)$. Each hash function yields a position $h_k(j)$ in the Bloom filter, pseudo-randomly selected based on the identifier of the item j. One simple technique to implement this is to rely on K cryptographic hash functions modulo L. We call the codeword $\mathbf{X}_j$ associated to item j the following string of L bits:

$$X_j(\ell) = \begin{cases} 1 & \text{if } \exists k \in [K] \text{ such that } h_k(j) = \ell, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

The Bloom filter associated to the profile $P = \{j_1, \ldots, j_c\}$ is denoted by $\mathbf{B}_P$ and computed as the aggregation of the codewords:

$$\mathbf{B}_P = \mathbf{X}_{j_1} \vee \ldots \vee \mathbf{X}_{j_c}, \tag{3.2}$$

in which $\vee$ denotes the bit-wise (inclusive) OR operator. Our presentation of Bloom filters is different than usual to stress the link with our general problem.

The BLIP mechanism adds noise to the Bloom filter representation of a profile before publishing it. We denote the output of BLIP by $\tilde{\mathbf{B}}_P$:

$$\tilde{\mathbf{B}}_P = \mathbf{B}_P \oplus \mathbf{N}, \tag{3.3}$$

in which $\oplus$ corresponds to the bit-wise logical (exclusive) XOR operator and $\mathbf{N} \in \{0,1\}^L$ is a random binary string of size L, whose symbols are *i.i.d.* (independent and identically distributed) as a Bernoulli distribution $\mathcal{B}(p_\epsilon)$ (*i.e.* , $N(\ell) \in \{0,1\}$ and $\mathbb{P}[N(\ell) = 1] = p_\epsilon$, $\forall \ell \in [L]$). Alaggan, Gambs and Kermarrec [AGK12] proved that the BLIP mechanism ensures ϵ-differential privacy for the items of the profile if

$$p_\epsilon = 1/(1 + e^{\epsilon/K}). \tag{3.4}$$

3.1.2 The simple model

We assume that the hash functions produce independently random outputs, which means that the probability that $h_k(j)$ "points" to a given index is $1/L$. This assumption implies that the bits of the codewords can be modeled as independent Bernoulli random variables: $X_j(\ell) \sim \mathcal{B}(p)$, $\forall (j,\ell) \in [N] \times [L]$ with

$$p \triangleq \mathbb{P}[X_j(\ell) = 1] = 1 - \left(1 - \frac{1}{L}\right)^K. \tag{3.5}$$

For a random P composed of c items, we have $B_P(\ell) \sim \mathcal{B}(\pi_c)$, $\forall \ell \in [L]$, with

$$\pi_c \triangleq \mathbb{P}[B_P(\ell) = 1] = 1 - (1-p)^c = 1 - \left(1 - \frac{1}{L}\right)^{cK}. \tag{3.6}$$

As for the BLIP, $\tilde{\mathbf{B}}_P$ contains i.i.d. random symbols $\tilde{B}_P(\ell) \sim \mathcal{B}(\tilde{\pi}_c)$ with

$$\tilde{\pi}_c \triangleq \mathbb{P}[\tilde{B}_P(\ell) = 1] = (1 - p_\epsilon)\pi_c + p_\epsilon(1 - \pi_c). \tag{3.7}$$

3.1.3 More complex models

This subsection presents two possible extensions of the simple model, in which we no longer assume that c is fixed in advance and publicly known.

To account for this, we introduce the probability $\mathbb{P}[|P| = c]$, in which $|P|$ denotes the number of items in P. Then, we have to replace π_c by:

$$\pi_c \to \pi = \sum_{c>0} \pi_c \mathbb{P}[|P| = c]. \tag{3.8}$$

This new expression leads to $\tilde{\pi} = (1-p_\epsilon)\pi + p_\epsilon(1-\pi)$. Not knowing c may not be a big challenge for the adversary because he can easily infer the number of items in a profile. The quantity $\omega(\tilde{\mathbf{B}}_P)/L$, in which $\omega(.)$ is the Hamming weight of a binary string (the number of bits set to one), is an unbiased estimator of $\tilde{\pi}_c$. Inverting (3.7) is possible when $p_\epsilon \neq 1/2$ (*i.e.* , $\epsilon > 0$) since p_ϵ is public:

$$\hat{\pi}_c = \frac{\omega(\tilde{\mathbf{B}}_P)/L - p_\epsilon}{1 - 2p_\epsilon}, \tag{3.9}$$

which in turn gives an estimator $\hat{c}$ by inverting (3.6). In the same way, a confidence interval for $\tilde{\pi}_c$ based on $\omega(\tilde{\mathbf{B}}_P)/L$ yields a confidence interval $[c_{\min}, c_{\max}]$ on c.

An even more refined model consists in taking into account the popularity of the items. Indeed, popular items impact the Bloom filter by ensuring that some of its bits are more likely to be set to one. To tackle this issue, we still pretend that the bits are independent but distributed according their own Bernoulli law: $B_P(\ell) \sim \mathcal{B}(\pi(\ell))$, $\forall \ell \in [L]$. The same model holds for the BLIP symbols: $\tilde{B}_P(\ell) \sim \mathcal{B}(\tilde{\pi}(\ell))$, with $\tilde{\pi}(\ell) = (1 - p_\epsilon)\pi(\ell) + p_\epsilon(1 - \pi(\ell))$.

3.2 JLT

Kenthapadi and co-authors [KKMM12] proposed another mechanism to prevent the adversary from learning the presence (or absence) of an item in the profile, although their scheme tackles a different data type (*i.e.*, real vector). In the sequel, we denote this proposal by JLT because it is based on the Johnson-Lindenstrauss Transform. A technical description of JL Transform is presented in the section 2.3.3.1.

3.2.1 Description

The profile is encoded in the form of a real vector of length L as follows. A codeword $\mathbf{X}_j$ associated to item j is a real vector. Its L components have been independently and identically drawn such that $X_j(i) \overset{i.i.d.}{\sim} \mathcal{N}(0, 1/L)$, $\forall (i,j) \in [L] \times N$. The codebook $(\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N)$ is generated once for all and is public. Profile P is encoded into vector $\mathbf{Y}_P = \sum_{j \in P} \mathbf{X}_j$, then the user adds a noise $\mathbf{N}$ (private data) before publishing $\tilde{\mathbf{Y}}_P = \mathbf{Y}_P + \mathbf{N}$. The authors of [KKMM12] recommend a white Gaussian noise: $N(i) \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$. According to [KKMM12, Lemma 2], if

$$L \geq 2(\log(N) + \log(2/\delta)), \quad \sigma \geq \frac{4}{\epsilon}\sqrt{\log(1/\delta)} \text{ and } \epsilon < \log(1/\delta) \tag{3.10}$$

then this mechanism complies with (ϵ, δ)-differential privacy (for $0 < \delta < 1$).

3.2.2 A simple probabilistic model

The adversary does not know the profile P and therefore he models the observation $\tilde{\mathbf{Y}}_P$ as a white Gaussian noise since $\tilde{\mathbf{Y}}_P$ is the sum of $c+1$ white Gaussian noises. As these patterns are statistically independent, their powers sum up so that $\tilde{Y}_P(i) \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2 + c/L)$. We assume now that σ^2 is a recommended noise power, and thus that it is a public parameter. This allows the adversary to estimate the number of items in profile P in the following manner:

$$\hat{c} = \frac{L}{L-1} \sum_{i=1}^{L} \tilde{Y}_P(i)^2 - L\sigma^2. \tag{3.11}$$

Consider now the case in which the adversary knows that the item j is in the profile. This knowledge stems into a refined statistical model of the observation: $\tilde{Y}_P(i) \overset{i.i.d.}{\sim} \mathcal{N}(X_j(i), \sigma^2 + (c-1)/L)$. In the same way, knowing the profile P ends up with $\tilde{Y}_P(i) \overset{i.i.d.}{\sim} \mathcal{N}\left(\sum_{j \in P} X_j(i), \sigma^2\right)$.

3.3 Theoretical analysis

In this section, we propose two decoders that can be used by an adversary to reconstruct the profile of a given user out of his public representation. This analysis is detailed for

the BLIP mechanism, but similar concepts hold for the JLT scheme. The expressions of the information theoretical quantities are given in subsection 3.3.3.1 for BLIP and subsection 3.3.3.2 for JLT.

3.3.1 Single decoder

From the observation of one BLIPed representation $\tilde{\mathbf{b}}$, the adversary would like to infer which item belongs to the original profile. The adversary can conduct this inference by analyzing the L symbols of $\tilde{\mathbf{b}}$ and making an hypothesis test about the presence of item j in the underlying profile.

- $\mathcal{H}_0$: Item j is not in the profile, which means that the observed BLIP symbols are statistically independent from the symbols of codeword $\mathbf{X}_j$: $\mathbb{P}[\tilde{B}_P(\ell), X_j(\ell)] = \mathbb{P}[\tilde{B}_P(\ell)]\mathbb{P}[X_j(\ell)]$, $\forall \ell \in [L]$.
- $\mathcal{H}_1$: Item j belongs to P, and thus there is a slight dependency between the symbols of the observed BLIP and that of codeword $\mathbf{X}_j$: $\mathbb{P}[\tilde{B}_P(\ell), X_i(\ell)] = \mathbb{P}[\tilde{B}_P(\ell)|X_i(\ell)]\mathbb{P}[X_i(\ell)]$, $\forall \ell \in [L]$.

For a given item, this test may make two types of error: 1) False positive rate α_1: The probability of detecting the presence of an item that does not belong to the profile; 2) False negative rate α_2: The probability of missing the presence of an item that belongs to the profile. Information theory gives an upper bound on the performance of the test thanks to the Stein's lemma. More precisely, for a given α_2, the probability of false positive cannot be lower than

$$\alpha_1 \geq e^{-(I(\tilde{\mathbf{B}}_P;\mathbf{X})+1)/(1-\alpha_2)}, \tag{3.12}$$

in which $I(\tilde{\mathbf{B}}_P;\mathbf{X})$ is the mutual information between a BLIPed filter and the codeword of an item of the profile.

This test concerns a particular item, but an adversary that wants to reconstruct the whole profile needs to repeat it for the whole ensemble of size N. This repetition increases the global probability of false positive η_1:

$$\eta_1 = 1 - (1-\alpha_1)^{N-c} \lesssim N\alpha_1, \tag{3.13}$$

in which we assume that $N\alpha_1 \ll 1$ and $c \ll N$. η_1 is the probability that at least one item not in the profile is detected as belonging to the profile. At the end, for targeted error probabilities (α_2, η_1), inequality (3.12) constraints the size of the item ensemble the adversary can deal with:

$$\log(N) \leq \frac{I(\tilde{\mathbf{B}}_P;\mathbf{X})}{1-\alpha_2} + \log \eta_1. \tag{3.14}$$

The last inequality stresses the important role of $I(\tilde{\mathbf{B}}_P;\mathbf{X})$. Subsections 3.3.3.1 and 3.3.3.2 provide expressions of this quantity for the BLIP and JLT mechanisms.

3.3.2 Joint decoder

Let us consider another strategy. From the observation $\tilde{\mathbf{b}}$, the adversary would like to test whether P was the original profile that gave birth to this BLIPed representation. The difference with the previous approach is that the presence of items are not tested independently but jointly, hence the name "joint decoder".

Basically, the analysis is the same as previously except that the information theoretic quantity is now $I(\tilde{\mathbf{B}}_P; P) = I(\tilde{\mathbf{B}}_P; (\mathbf{X}_{j_1}, \ldots, \mathbf{X}_{j_c}))$ and that the ensemble of profiles is much bigger. Roughly, $\log(|\mathcal{P}|) \approx c \log N$, thus we have:

$$\log(N) \leq \frac{I(\tilde{\mathbf{B}}_P; P)}{c(1-\alpha_2)} + \log \eta_1. \tag{3.15}$$

Stated differently, the performance of this approach is driven by the quantity $I(\tilde{\mathbf{B}}_P; P)/c$. Theorem [Mou08, Eq. (3.4)] states that $I(\tilde{\mathbf{B}}_P; (\mathbf{X}_{j_1}, \ldots, \mathbf{X}_{j_c}))/c \geq I(\tilde{\mathbf{B}}_P; \mathbf{X}_j)$, which means that considering the items jointly yields better performances. Subsections 3.3.3.1 and 3.3.3.2 provide expressions of this quantity for respectively the BLIP and JLT mechanisms. For this first scheme, subsection 3.3.3.1 shows that the difference $I(\tilde{\mathbf{B}}_P; (\mathbf{X}_{j_1}, \ldots, \mathbf{X}_{j_c}))/c - I(\tilde{\mathbf{B}}_P; \mathbf{X}_j)$ can be be substantial for practical setups. We also provide upper bounds simply depending on ϵ.

3.3.3 Information theoretic bounds

3.3.3.1 BLIP mechanism

We have $I(\tilde{\mathbf{B}}_P; \mathbf{X}) = H(\tilde{\mathbf{B}}_P) - H(\tilde{\mathbf{B}}_P|\mathbf{X})$, in which H is the (Shannon) entropy of a random variable. With the simple model detailed in Section 3.1.2, we get that

$$I(\tilde{\mathbf{B}}_P; \mathbf{X}) = L(h_b(\tilde{\pi}_c) - (1-p)h_b(\tilde{\pi}_{c-1}) - p h_b(p_\epsilon)), \tag{3.16}$$

with $h_b(p)$ the entropy of a Bernoulli distribution $\mathcal{B}(p)$ (in nats):

$$h_b(p) \triangleq -p \log(p) - (1-p) \log(1-p) = h_b(1-p). \tag{3.17}$$

The probabilities $\tilde{\pi}_c$ and $\tilde{\pi}_{c-1}$ appear in (3.16) because we assume that the profiles are of identical size c. When considering more complex but also more practical models, this difference vanishes as $\tilde{\pi}_c$ and $\tilde{\pi}_{c-1}$ are replaced by $\tilde{\pi}$:

$$I(\tilde{\mathbf{B}}_P; \mathbf{X}) \approx Lp(h_b(\tilde{\pi}) - h_b(p_\epsilon)). \tag{3.18}$$

As for the joint decoding, Bloom filter being a deterministic process, we write:

$$\begin{aligned} I(\tilde{\mathbf{B}}_P; P) &= I(\tilde{\mathbf{B}}_P; \mathbf{B}_P) = H(\tilde{\mathbf{B}}_P) - H(\tilde{\mathbf{B}}_P|\mathbf{B}_P) \\ &= H(\tilde{\mathbf{B}}_P) - H(\mathbf{N}) = L(h_b(\tilde{\pi}_c) - h_b(p_\epsilon)). \end{aligned} \tag{3.19}$$

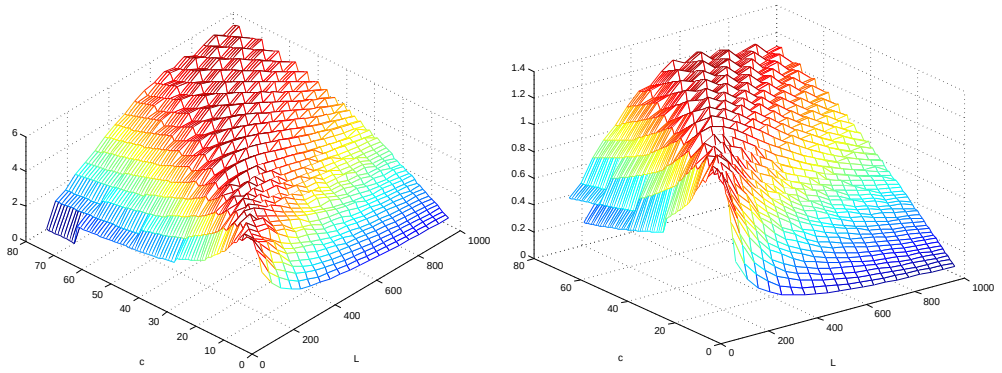

Figure 3.1: (Left): Mutual information of the joint decoder $I(\tilde{\mathbf{B}}_P; P)/c$ in nats as a function of (c, L). (Right): Difference $I(\tilde{\mathbf{B}}_P; P)/c - I(\tilde{\mathbf{B}}_P; X)$ in nats as a function of (c, L).

Example.

Figure 3.1 (left) shows $I(\tilde{\mathbf{B}}_P; P)/c$ as a function of c and L. From a particular (c, L), we set

$$K = \left\lfloor \log(2) \frac{L}{c} \right\rfloor, \tag{3.20}$$

which is the recommended number of hash functions in Bloom filter design, and we apply the model of Section 3.1.2 with $\epsilon = 20$. For a given c, too small L means too few observed symbols for reliably estimating the profile. Too large L implies a big K and therefore, p_ϵ tends to $1/2$ according to (3.4). Figure 3.1 (right) shows that $I(\tilde{\mathbf{B}}_P; P)/c - I(\tilde{\mathbf{B}}_P; \mathbf{X})$ can be substantial: a joint decoding allows the adversary to tackle up to 3.5 (*i.e.* $e^{1.25}$) times more items.

Upper bounds.

As $\epsilon \to 0$, $p_\epsilon \to 1/2$ as well as $\tilde{\pi}$, so that $I(\tilde{\mathbf{B}}_P; \mathbf{X}) \to 0$ and also $I(\tilde{\mathbf{B}}_P; P) \to 0$. When $\epsilon = 0$, observing the BLIP is useless since it brings no information. In this situation, neither the single nor the joint decoding can do anything. We can bound the quantity in common in both expressions as follows:

$$\begin{aligned} h_b(\tilde{\pi}_c) - h_b(p_\epsilon) &\leq \log(2) - h_b(p_\epsilon) \leq \log(2) - \log\left(1 + e^{\epsilon/K}\right) + \frac{\epsilon}{K} \frac{e^{\epsilon/K}}{1 + e^{\epsilon/K}} \\ &\leq \frac{\epsilon}{K} \frac{e^{\epsilon/K}}{1 + e^{\epsilon/K}} \leq \frac{\epsilon}{K}. \end{aligned} \tag{3.21}$$

Typical Bloom filter setup.

Figure 3.1 shows that estimating an important number of items is possible provided that L grows linearly with c. Indeed, it is also common practice in the design of Bloom

filter to set:

$$L = \left\lceil -c\frac{\log(P_{fp})}{(\log 2)^2} \right\rceil, \tag{3.22}$$

in which P_{fp} is the probability of false positive of the Bloom filter (*i.e.*, to detect the presence of an item not belonging to P). Inserting (3.20) and (3.22) in the expression of the mutual informations, we get quantities independent of c:

$$\frac{1}{c}I(\tilde{\mathbf{B}}_P; P) \sim -\frac{\log(P_{fp})}{\log(2)}\left(1 - \frac{1}{\log(2)}h_b\left((1+2^{\frac{\epsilon}{-\log(P_{fp})}})^{-1}\right)\right), \tag{3.23}$$

$$I(\tilde{\mathbf{B}}_P; \mathbf{X}) \sim \log(2).\frac{1}{c}I(\tilde{\mathbf{B}}_P; P). \tag{3.24}$$

This shows that if the Bloom filter is properly designed, the power of the attack does not depend on c but solely on the values of $-\log(P_{fp})$ and ϵ. Moreover, the joint decoder is $1/\log(2) \sim 1.44$ more "powerful" than the single decoder.

3.3.3.2 JLT mechanism

The same analysis holds for the JLT representation described in Section 3.2. The main difference lies in the fact that we manipulate differential entropies because the JLT representation is a real vector. The quantities at stake respectively for the single and joint decoders are upper bounded, thanks to conditions (3.10)

$$I(\tilde{\mathbf{Y}}_P; \mathbf{X}) = \frac{L}{2}\log\left(1 + \frac{1}{(c-1)+L\sigma^2}\right) \leq \frac{\epsilon}{32 + 2\epsilon(c-1)L}, \tag{3.25}$$

$$\frac{I(\tilde{\mathbf{Y}}_P; P)}{c} = \frac{L}{2c}\log\left(1 + \frac{c}{L\sigma^2}\right) \leq \frac{\epsilon}{32}, \tag{3.26}$$

3.4 Practical decoders

The previous section can be summarized as follows: joint decoding is theoretically more powerful than single decoding. However, no complexity argument has been so far taken into account. This section deals with this issue by proposing practical implementations of a single and a joint decoder. Again, we take the example of BLIP but our approach is more generic as it works also with JLT.

3.4.1 Single decoders

In practice, a single decoder computes from the observed BLIPed profile a score s_j for any item $j \in [N]$, which reflects the likelihood of belonging to the profile (*i.e.*, the most likely item has the highest score). The score is compared to a threshold to decide whether or not the item should be included in the reconstructed profile. The complexity of this single decoder is $O(N)$ since it is exhaustive and goes through all the possible items.

As a practical implementation, we propose the Maximum Likelihood decoder in which the score $s_j = \log \frac{\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}} | j \in P]}{\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}]}$ equals, by independence of the symbols:

$$s_j = n_{11} \log \frac{1 - p_\epsilon}{\tilde{\pi}} + n_{01} \log \frac{p_\epsilon}{1 - \tilde{\pi}}, \quad \text{with:} \tag{3.27}$$

$$n_{11} = |\{\ell \in [L] | \tilde{b}(\ell) = 1 \,\text{AND}\, X_j(\ell) = 1\}|, \tag{3.28}$$

$$n_{01} = |\{\ell \in [L] | \tilde{b}(\ell) = 0 \,\text{AND}\, X_j(\ell) = 1\}|. \tag{3.29}$$

This decoder is derived from models that are more realistic in which $\pi_c \approx \pi_{c-1} \approx \pi$, so that the score of item j only takes into account the $(n_{11} + n_{01})$ symbols in which $X_j(\ell) = 1$ (*i.e.*, at most K symbols over L).

3.4.2 Joint decoder

In practice, a joint decoder computes from the observed BLIPed filter a score for any profile $P' \in \mathcal{P}$, which reflects the likelihood that P' is the true profile. This score is computed by taking into account L symbols but the complexity of a joint decoder is proportional to $|\mathcal{P}|$ (*i.e.*, $O(N^c)$), which is computationally expensive. Yet, there exists at least three possible approaches that approximate joint decoding with a reasonable complexity: 1) Monte Carlo Markov Chain (MCMC) [KST96, FGC12], 2) Belief Propagation Decoder [SJ10] and 3) Joint Iterative Decoder [MF12].

We investigate the first approach. The MCMC decoder is based on two key ideas. First, it receives as input an observed BLIPed filter $\tilde{\mathbf{b}}$ and then creates a Markov Chain that will be used to sample profiles according to the posterior distribution $\mathbb{P}[P|\tilde{\mathbf{b}}]$. This sampling requires a burn-in period after which the Markov Chain has converged. Once this convergence has occurred, it samples profiles with the targeted posterior distribution. During a second phase, some profiles are sampled and statistics are computed such as the marginal a posteriori distribution $\hat{\mathbb{P}}[j \in P|\tilde{\mathbf{b}}]$ that item j belongs to the true profile.

Posterior distribution. The objective is to sample profiles according to the posterior distribution $\mathbb{P}[P|\tilde{\mathbf{b}}]$, which can be written as:

$$\mathbb{P}[P|\tilde{\mathbf{b}}] = \frac{\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}|P]\mathbb{P}[P]}{\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}]}. \tag{3.30}$$

In this equation, $\mathbb{P}[P]$ is the a priori probability of P. To simplify our presentation, we consider only the simple model exposed in Section 3.1.2. We denote by $|P|$ the size of profile P (*i.e.*, the number of his items), and we set by $\mathbb{P}[P] = 0$ if $|P| \neq c$, and $1/|\mathcal{P}|$ otherwise. Any profile is equally likely provided it has exactly c items. When we use more realistic models in our experimental work, the prior will be substantially different.

We denote by $\omega(\mathbf{B})$ the Hamming weight of a binary vector $\mathbf{B}$ (*i.e.*, the number of bits set to 1). The probability $\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}|P] = \mathbb{P}[\mathbf{N} = \mathbf{B}_P \oplus \tilde{\mathbf{b}}]$ has the following expression

$$\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}|P] = p_\epsilon^{\omega(\mathbf{B}_P \oplus \tilde{\mathbf{b}})}(1-p_\epsilon)^{L-\omega(\mathbf{B}_P \oplus \tilde{\mathbf{b}})}. \tag{3.31}$$

The evaluation of the last quantity $\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}]$ in (3.30) is more involved:

$$\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}] = \sum_{P \in \mathcal{P}} \mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}|P]\mathbb{P}[P]. \tag{3.32}$$

It requires a screening of $\mathcal{P}$, which is intractable for large c and N, which is why we will rely on the Markov chain.

Markov Chain. A Markov Chain is an iterative process with an internal state (*i.e.*, a profile in our case) taking value $P^{(t)}$ at iteration t. The next iteration draws a new state $P^{(t+1)}$ according to a transition probability distribution $\mathbb{P}[P^{(t+1)}|P^{(t)}]$. The Markov Chain is initialized randomly at state $P^{(0)}$. The probability distribution of transitions is crafted with care to enforce a convergence of the distribution of sampled profile $P^{(t)}$ to the posterior $\mathbb{P}[P|\tilde{\mathbf{b}}]$ of (3.30) as $t \to \infty$ (see Section 3.4.2). In practice, the convergence occurs after the first T iterations, the so-called burn-in period. Once this period has passed, it means that the Markov Chain has forgotten its starting point (*i.e.*, the samples are now independent of $P^{(0)}$) and that the distribution of the sample profiles has converged.

Monte Carlo method. After the burn-in period, the Markov Chain keeps on sampling for M more iterations. The marginal a posteriori probabilities are then estimated by a Monte Carlo method, which computes the empirical frequency that item j is present in sample $P^{(t)}$:

$$\hat{\mathbb{P}}[j \in P|\tilde{\mathbf{b}}] = |\{t \in [T+1, T+M]|j \in P^{(t)}\}|/M. \tag{3.33}$$

From these estimations, several post-processing are possible such as:

- inferring the most likely items of the true profile by ranking them in decreasing marginal probabilities,
- reconstructing the profile as the set of items whose marginal probability is above a given threshold,
- reconstructing the profile as the set of items with highest marginal.

Transition probabilities

Algorithmic coding of a profile.

Section 3.1.3 describes how to infer from the observed BLIP a maximum number $c_{\max}$ of items of the corresponding profile. In this algorithm, we code a profile as a vector of $c_{\max}$ components taking values in $[N] \cup \{0\}$. Some of these components may take the value "0" meaning an "empty item", while the others have different values (*i.e.*, there is no pair of non-zero components with the same value). For instance, for $c_{\max} = 5$, $P = (0, 3, 2, 0, 4)$ represents the profile of 3 items: #2, #3 and #4.

We define $\mathcal{V}(P_0, i)$ as the neighborhood of profile P_0 in the following manner:

$$\mathcal{V}(P_0, i) = \{P \in \mathcal{P} | P(k) = P_0(k) \quad \forall k \neq i\}. \tag{3.34}$$

This neighborhood profile is the set of all profiles whose coding differs at most from the i-th component. Note that $P_0 \in \mathcal{V}(P_0, i)$. If $P_0(i) = 0$, this neighborhood comprises profiles having at most one more item. Otherwise if $P_0(i) > 0$, this neighborhood contains profiles having at most one different item (*i.e.*, $P_0(i)$ is substituted by another item) and one profile having one less item (*i.e.*, item $P_0(i)$ is substituted by 0, the "empty item").

Multi-stage Gibbs sampling.

Instead of computing the transition probabilities for all the possible profiles, we restrict the transitions to the neighborhood of the actual state. At the iteration $t + 1$, an integer i is first uniformly drawn in $[c_{\max}]$ that indicates the subset $\mathcal{V}(P^{(t)}, i)$. Then, the following transition probability distribution is computed: $\forall P \in \mathcal{V}(P^{(t)}, i)$

$$\mathbb{P}[P^{(t+1)} = P | P^{(t)}] = \frac{\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}} | P] \mathbb{P}[P]}{\sum_{P' \in \mathcal{V}(P^{(t)}, i)} \mathbb{P}[\tilde{\mathbf{B}}_{P'} = \tilde{\mathbf{b}} | P'] \mathbb{P}[P']} \tag{3.35}$$

Iteration $t + 1$ ends by randomly drawing state $P^{(t+1)}$ from this distribution.

This choice of probabilistic transitions is called a multi-stage Gibbs sampler with random scan [RC04, Alg. A.42]. It guarantees that the law of sampled profiles converges to the stationary distribution $\mathbb{P}[P|\tilde{\mathbf{b}}]$, which legitimates our approach [RC04, Sect. 10.2.1]. The unknown multiplicative constant $\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}]$ in (3.30) has disappeared in the ratio. This transition probability distribution only depends on the priors $\mathbb{P}[P]$ (which depends on the mathematical model of a profile), and the conditional probabilities $\mathbb{P}[\tilde{\mathbf{B}}_P = \tilde{\mathbf{b}}|P]$ (which depends on the privacy-preserving mechanism). For instance, for the JLT mechanism, $\mathbb{P}[\tilde{\mathbf{Y}}_P = \tilde{\mathbf{y}}|P] \propto \exp(-\|\tilde{\mathbf{y}} - \sum_{j \in P} \mathbf{X}_j\|^2 / 2\sigma^2)$.

3.5 Experiments

3.5.1 Setup

In this section, we test the inference attacks designed on two real datasets: Digg and MovieLens. The Digg dataset has been collected on a social news aggregator and the

Table 3.1: Datasets characteristics

Dataset	Nb of users	Training set size	Testing set size	N	c_{avg}	Sparsity %
Digg	531	331	200	1237	317	25.63%
MovieLens	943	600	343	1682	106	6.30%

profile of a user is composed of the news he has read. The MovieLens dataset is a snapshot from a movie recommendation site and in this dataset the profile of a user is composed of the movies he likes. For the experiments, we split both datasets into two parts : the training set and the testing set. The characteristics of these datasets are summarized in Table 3.1, in which c_{avg} is the average number of items per profile and sparsity is the average occupancy of items among the user profiles.

During the experiments, we assume that the adversary has access to some raw profiles of users to estimate the item priors (*i.e.*, popularities of items). This is similar to assuming that the adversary has access to some global information about the general distribution of items in the population. We rely on the training dataset for computing the frequencies of items while the testing dataset is used solely for evaluating the performance of the attacks. In terms of parameters, for BLIP we set the number of hash functions $K = 20$ and the number of bits of the representation to $L = 5,000$. The values of ϵ are from the set $\{59, 28, 17, 8, 6, 5, 3, 2, 0\}$, which equivalently translate to the corresponding flipping p_ϵ from the range $\{0.05, 0.2, 0.3, 0.4, 0.42, 0.44, 0.46, 0.48, 0.5\}$. For the JLT scheme, we set the size of the representation L to $1,000$. L is set to a lower value as the representation, a dense real vector, is richer than the binary version of BLIP. The privacy parameter ϵ takes value in the set $\{600, 6, 3, 2, 1, 0.75, 0.5, 0.25, 0.1\}$, which translates into a noise level σ in $\{0, 1, 2, 3, 6, 8, 12, 24, 61\}$.

For MCMC, we used a burn-in period of $T = 1,000$ samples and estimation sample size of $M = 19,000$ for all the experiments. In practice, we observed that the performance is not very sensitive to the burn-in period length. As with other MCMC based approaches proper initialization for sampling is highly desirable for a faster convergence to the stationary distribution. We used the input public representation of the profile to estimate $\hat{c}$ and started with $\hat{c}$ random items. A poor estimation of $\hat{c}$ has to be traded-off with a longer burn-in period. We also prefilter items that are to be tested against the public profile for joint decoder, to reduce the search space. To realize this, we first predict the $f \times \hat{c}$ most probable items for a given profile ($f \in [2, 6]$) using single decoder and then run the joint decoder on the filtered items to return $\hat{c}$ items. This prefiltering decreases significantly the running time of the algorithm without impacting the prediction as only unlikely items will not be considered by the joint decoder.

3.5.2 Reconstruction attacks

We benchmark four attacks that produce a score per item:

- The single decoder.
- The popularity-based attack in which the score of an item is its prior estimated from the training data, independent of the given public representation.
- Our MCMC joint decoder with and without priors (*i.e.*, with flat priors) in which the scores are the estimated marginal a posteriori probabilities.

Reconstruction $\hat{P}$ is then the list of the top $\hat{c}$ items ranked based on their scores.

We measure the performance of a reconstruction attack by computing the cosine similarity between the reconstruction $\hat{P}$ and the true profile P as expressed in (3.36) for all the profiles of the testing set.

$$\cos(P, \hat{P}) = \frac{|P.\hat{P}|}{|P||\hat{P}|} \tag{3.36}$$

Afterwards, we compute the following statistics: average, the 10% and the 90% quantiles of the cosine similarities.

The plots in Figure 3.2 show that the performance of the reconstruction attack is better for high values of ϵ while it degrades as $\epsilon \to 0$. In this case, $p_\epsilon \to 0.5$ and every profile becomes equiprobable so that inferring the original profile becomes impossible. In addition, $\hat{c}$ depends on ϵ and low value results in a poor estimation of $\hat{c}$, which impacts the similarity measure as only top $\hat{c}$ items of the prediction is considered in the reconstructed profile. As the estimation of $\hat{c}$ is performed similarly for all the four attacks, the performance drop is common to all of them. Overall the performance of our MCMC attack is better than the single decoder of [AGK12] for almost all ϵ values over the two datasets. Another way to see this is to find the range of ϵ in which a given attack performs worse than the baseline (*i.e.*, the popularity-based attack). For instance, by setting $\epsilon = 8$, the designer is sure that the single attack is no longer a threat. However, a skilled adversary can reconstruct almost 50% of the profile thanks to our MCMC attack.

Taking into account the prior of items improves the efficiency in the reconstruction significantly, provided that the estimation is reliable. This improvement is clearly observed on the MovieLens dataset. As for the Digg setup, priors of the training set do not generalized to the test set, hence they do not help much. We conducted the same experiment with the JLT scheme. The figure is shown in 3.3, and the results that we obtained are very close from the one of BLIP and thus we can draw the same conclusions.

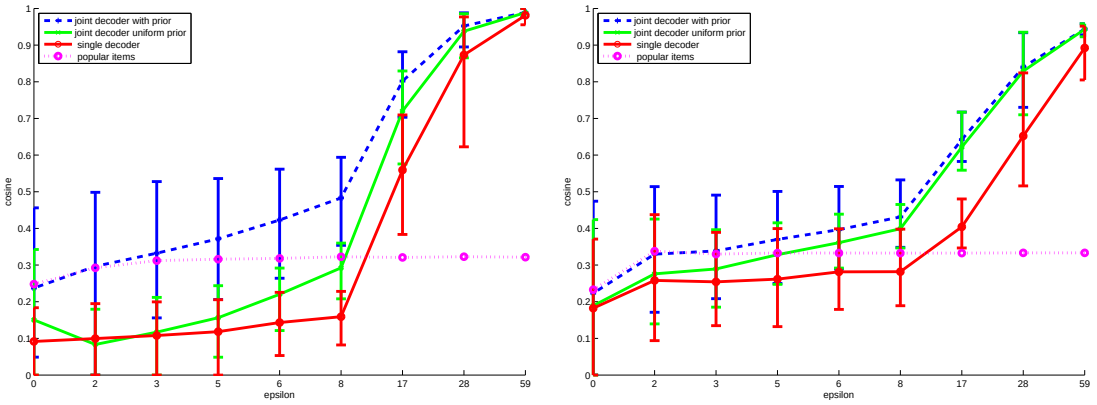

Figure 3.2: Values of the cosine similarity (average, 10% quantile and 90% quantile) of BLIP for MCMC with prior, with no prior and single decoding for various ϵ on Movielens (left) and Digg (right) dataset.

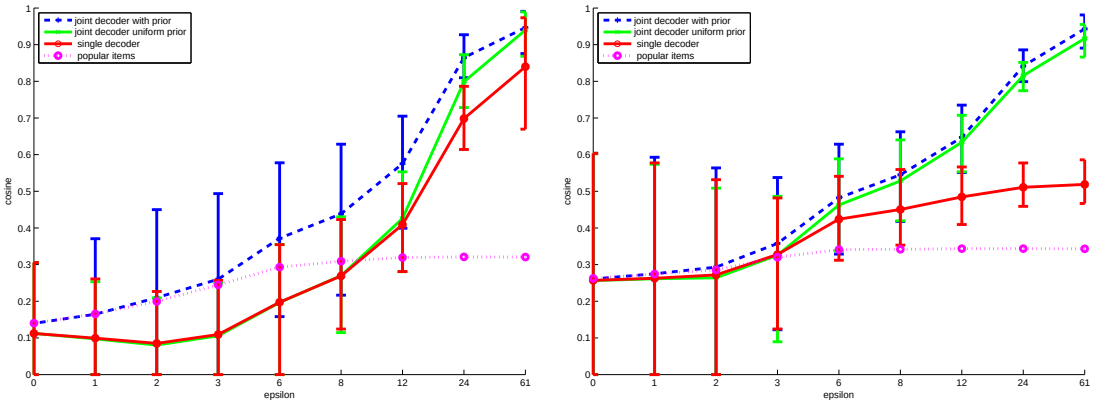

Figure 3.3: Values of the cosine similarity (average, 10% quantile and 90% quantile) of JLT for MCMC with prior, with no prior and single decoding for various ϵ on Movielens (left) and Digg (right) dataset.

3.5.3 Identifying the presence of an item

When ϵ is very small, Figure 3.2 clearly shows that the adversary cannot hope to reconstruct the full profile. In this situation, we evaluate the prediction of top R items, with $R \ll c$, as another assessment of the privacy guarantees. The success is measured in terms of the mean Average Precision at R (mAP@R) given in (3.37), which is the mean over the Q profiles in the test dataset of the average of the precisions at rank $1 \leq r \leq R$. The precision(r) refers to the fraction of correct items out of the top r predicted items. The mAP is sensitive to the order of the correct results and is a better

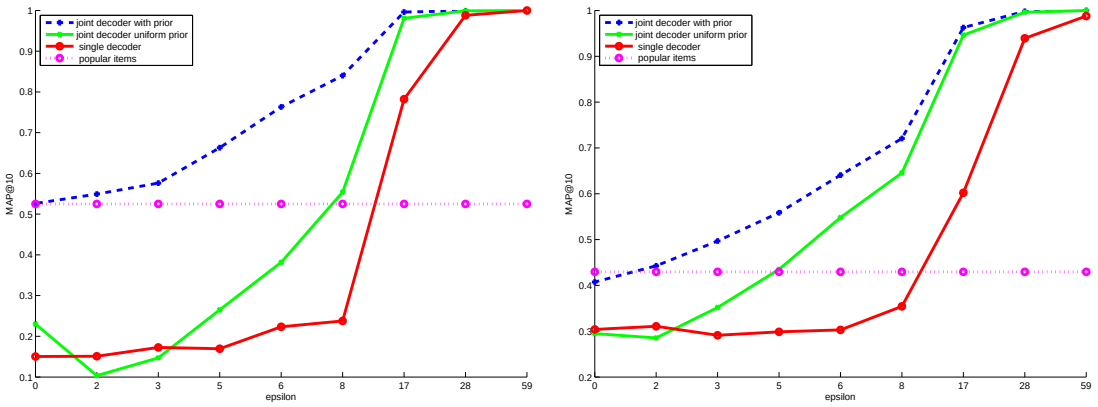

Figure 3.4: Mean Average Precision for $R = 10$ for BLIP for MCMC with prior, with no prior and single decoding for various ϵ on Movielens (left) and Digg (right) dataset.

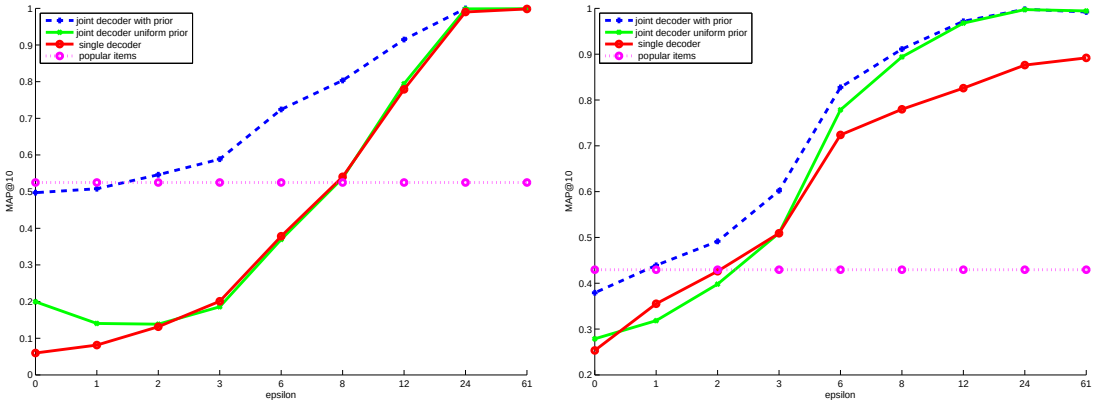

Figure 3.5: Mean Average Precision for $R = 10$ for JLT for MCMC with prior, with no prior and single decoding for various ϵ on Movielens (left) and Digg (right) dataset.

gauge of the quality of a ranking.

$$\text{mAP@}K = \frac{1}{Q}\sum_{q=1}^{Q}\left(\frac{1}{R}\sum_{r=1}^{R}\text{precision}_q(r)\right). \tag{3.37}$$

The characteristics of mAP@R depicted in Figures 3.4 and 3.5 are almost similar to the exact reconstruction measurement. Even if the exact reconstruction of profile is hardly possible for a given ϵ, predicting the top R items work. For instance, the maximum reconstruction for $\epsilon = 0$ for Movielens is 0.23 whereas the mean average precision is close to 0.5. The same conclusion holds for the Digg dataset.

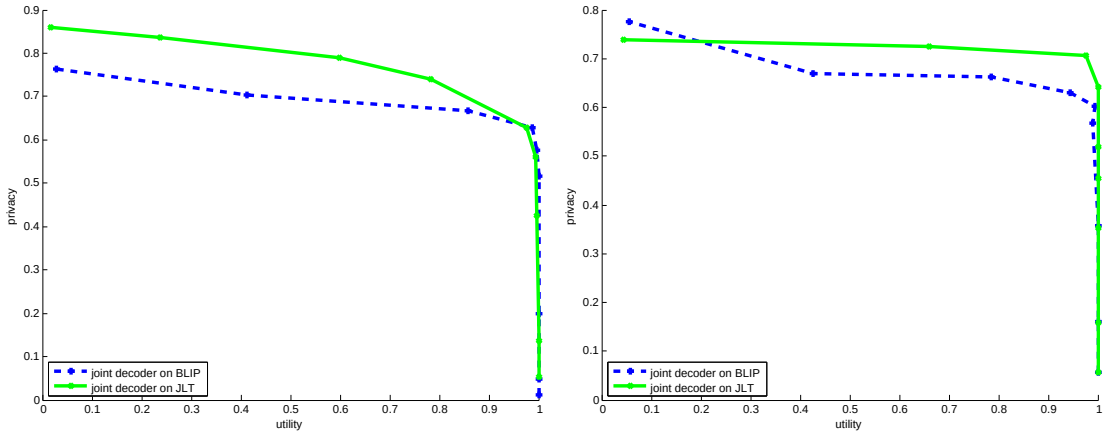

Figure 3.6: Utility against privacy for BLIP and JLT for various ϵ on Movielens (left) and Digg (right) datasets.

3.5.4 Utility-privacy trade-off

Finally, we also studied the achievable trade-off between privacy and utility. Since BLIP and JLT are used for similarity estimation, we quantify the utility in terms of the recall, which is defined as the probability of identifying the k-nearest neighbors (we set $k = 10$ in our experiments). In this experiment, we measure privacy as $1 - \cos(P, \hat{P})$ (see (3.36)) based on the joint decoder. Figure 3.6 illustrates the utility-privacy trade-off obtained for various ϵ. The trade-off is almost similar on the two datasets. The privacy preserving properties of JLT transform is slightly better than BLIP, at least for the parameters we used in our simulation. This difference in performance is due partially to the representation superiority of dense real vector over binary vector. However, BLIP offers a more compact representation of the profile ($5,000$ bits versus $1,000$ scalars). The plot is helpful in fixing ϵ giving good utility without compromising much on privacy.

3.6 Conclusion

In differential privacy, the trade-off between utility and privacy is set by the parameter ϵ. However, being able to choose an appropriate value for this parameter is still an open research question, which has not been deeply investigated, with a few exceptions [LC11, AACP11]. In this chapter, we have made a step forward to answer this question by proposing two generic inference attacks, namely single and joint decoding, whose objective is to reconstruct the profile of a user out of a differentially-private representation produced through a non-interactive mechanism. The first inference attack decides the presence of a single item and sequentially explores all the item set, while the latter strategy decides whether a subset of items is likely to be the user profile and considers all possible subsets.

We have evaluated the effectiveness of the attack on two schemes producing differentially private representations: BLIP (BLoom-and-flIP) [AGK12] and JLT (Johnson-Lindenstrauss Transform) [KKMM12]. Our theoretical analysis as well as the experimental results clearly shows that joint decoding is more powerful than single decoding. Overall, we believe that this attack helps better understanding the privacy guarantees offered by a wide class of differentially-private mechanisms (interactive or not) as well as for the privacy practitioner to tune experimentally ϵ to ensure the maximum utility without compromising much on privacy.

Chapter 4

Count sketching for matrix factorization

Outline

In this chapter, we describe our contributions towards improving the scalability and privacy aspects of model based collaborative filtering using sketching techniques. In particular, we use count sketch to store the latent factors in matrix factorization. We observe that sketch based factorization improves the scalability of factorization in a highly dynamic setup and is self regularized.

We describe the proposed method in section 4.1 and experimentally study the technique on benchmark real datasets with focus on the scalability and regularization properties of the proposed technique. We move on to the privacy properties of such sketch based factorization in section 4.2. Our approach utilizes the inherent randomness of the storage structure to preserve data privacy, contrasting to explicit noise added in conventional techniques. We compare our approach with a recent bayesian formulation of matrix factorization, which guarantees differential privacy guarantees. We then suggest a novel method of *clipping* the training data to vary the ϵ parameter in differential privacy. We also suggest a novel evaluation metric to measure the expected *information loss* of model based factorization. We experimentally validate on real datasets and study the privacy-utility tradeoff and the effectiveness of our proposed metric.

The rapid expansion of internet necessitated an efficient mechanism for discovering new and popular interesting content across diverse domains. The evolution of web 2.0 fueled the necessity further, with outpouring contributions from end users, whose previous role was just limited to be a consumer. Collaborative filtering systems filled

the niche gap of being such efficient mechanism by employing similarity among user and items to suggest new items to these users. A technical background of collaborative filtering systems was presented in section 2.2.2.2. Such collaborative filtering systems were revolutionized with the influx of new wave of users, inviting new challenges and making way for evolving and reprioritizing the requirements. The challenges faced by this transition is more pronounced in model based collaborative filtering systems. Two most prominent challenges among them are scalability and privacy.

Models built under a static data setup are becoming ineffective in handling the changes pertaining to user profiles and the volume of such changes calls for a scalable solution. The scalability challenges are not limited to a single function of the system but at various operational points such as user modeling, model updates and retrieval. In this chapter, we focus on the modeling and updates to the model. The scalability demands are both to the runtime and storage aspects of model based systems. We also observe that storage complexity dominates runtime complexity in large scale systems requiring an efficient means to utilize the allocated storage space for the user and item models.

Another challenge that is often forgotten is user data privacy, which is a cost the user pays for these personalization services. The consumed items have to be collected at a centralized database, to perform both analysis and prediction, which compromises user privacy. The usual anonymization alone is not sufficient as demonstrated in [CKN+11]. This calls for robust privacy preserving techniques. The usual perturbation method is to explicitly add noise to the sensitive statistic to prevent data leakage. The noise added is extraneous and serve no other purpose other than privacy. Hence there is a scope for efficient storage mechanisms that are inherently random, thereby privacy preserving and also provides additional benefits.

We provide sketching as a solution for both scalability and privacy problems in model based collaborative filtering systems. Sketch structures are simple in construction, efficient and inherently random, making it a good fit for all the above said problems. In particular, we demonstrate using count sketch to store the user and item models. A technical description of the count sketch technique was presented in section 2.3.2.5. To demonstrate the effectiveness of our technique, we chose matrix factorization, an acclaimed technique used primarily for approximate matrix reconstruction and noise reduction.

The section 4.1 describes addressing the scalability challenges and section 4.2 describes the privacy challenges solved by count sketch based matrix factorization technique.

4.1 Sketching techniques for very large matrix factorization

In this section, we demonstrate the scalability properties of a count sketch based matrix factorization, when employed to perform collaborative filtering in a streaming setup. The common scenario in collaborative filtering systems is that:

- maintaining the entire matrix $\mathbf{R}$ in memory is inefficient because it consumes way too much memory,
- the observed elements in $\mathbf{R}$ are, in many cases, not even available as a static data beforehand, but instead as a stream of tuples $< u, i, r_{u,i} >$.

Alternative representations for $\mathbf{R}$ have been invented, such as the popular *latent factor* model. The latent factor model maps both users and items to a low dimensional representation, retaining pairwise similarity. Matrix factorization learns these representation vectors from some observed ratings. They are then used to predict (by inner product) the missing entries and thereby to fill the incomplete user-item matrix [KBV09]. Compared to other methods, matrix factorization is simple, space efficient and can generalize well with missing information. It can be easily customized with different loss functions, regularization methods and optimization techniques. A technical discussion of matrix factorization is presented in section 2.2.5.

The storage requirements for the latent factors are significantly lower, however, the memory required for the factors grows linearly with the number of users and items. This becomes increasingly cumbersome when the numbers are in millions, a frequent real-world situation for domains like advertising and search personalization. The situation is complicated further when there are new incoming users and/or items. In this case, managing latent factors and running factorization techniques becomes inefficient.

Overall, supporting real-world large scale and dynamic recommendation applications asks for designing a much more compact representation for the latent factors, for techniques to more efficiently manipulate them (updates) while facilitating the insertion of new users and new items. Hence we propose using sketching techniques, in particular count sketch, to represent the latent factors in order to achieve the above goals. Count sketch enable to use extremely compact representations for the parameters, which help scaling. It also, by construction, facilitates updates and inserts.

We find through experimental results that sketch based factorization improves storage efficiency without compromising much on prediction quality. Furthermore, the randomized nature of these sketches provide inherent regularization abilities.

4.1.1 Sketching vectors

In regular matrix factorization, d-dimensional vectors $\{\mathbf{p}_u\}_{u\in\mathbb{U}}$ and $\{\mathbf{q}_i\}_{i\in\mathbb{I}}$ are stored as dense arrays $\mathbf{P}$ and $\mathbf{Q}$, contiguous in memory. This facilitates indexing on the two

dimensional array by increments of d. We propose replacing this matrix representation with a single count sketch. Surprisingly, although user and item vectors carry different semantic, their underlying representations are the same, therefore we store both of them in the same structure, which should provide estimates for $N = d(|\mathbb{U}| + |\mathbb{I}|)$ elements. For the sake of clarity, we introduce two families of address hash functions: $\{h_j^u(\cdot)\}_{j=1}^k$ for the users, $\{h_j^i(\cdot)\}_{j=1}^k$ for the items. Same for the sign hash functions.

By varying (w, k), we explore the trade-off between the storage efficiency and the quality of the estimation. The storage improvement comes at the cost of increasing the retrieval complexity from $O(d)$ to $O(kd)$ for a d-dimensional vector. The trade-off is acceptable, supported by the observation that memory bound computation are more common than CPU bound computation. Lowering memory requirements also makes it possible to process huge sparse matrices using main memory alone and avoiding out-of-the-core computations, thereby improving run time as well.

It should be noted that

- When $k = 1$, count sketch is equivalent to feature hashing and our approach is identical to the one described in [KWS10]. Hence we are generalizing the sketch based factorization to arbitrary k values.
- As the latent vectors are just estimations, dimension d can be modified any time (unlike regular factorization). This gives the flexibility to adapt at run time, based on the requirements.

4.1.2 Sketch based factorization

The sketch based online factorization differs from regular online factorization (Section 2.2.5) in the latent factor retrieval and gradient updates merging. When a new tuple $< u, i, r_{u,i} >$ arrives, the count sketch is queried to approximately reconstruct user and item latent vectors. Both user ID u and component index l, $1 \leq l \leq d$, are used as inputs to the k pairs of address and sign hash functions to get a mean estimate of the vector component as follows (the same holds for item):

$$\tilde{p}_{u,l} = \frac{1}{k} \sum_{j=1}^{k} s_j^u(u,l) \cdot c_{j,h_j^u(u,l)}, \quad \forall l \in \{1, \cdots, d\}, \tag{4.1}$$

$$\tilde{q}_{i,l} = \frac{1}{k} \sum_{j=1}^{k} s_j^i(i,l) \cdot c_{j,h_j^i(i,l)}, \quad \forall l \in \{1, \cdots, d\}. \tag{4.2}$$

Yet for the same user or item, the kd accessed cells in the count sketch structure are not adjacent but at random locations addressed by the hash functions.

The estimated rating $\tilde{r}_{u,i} = \tilde{\mathbf{p}}_u^\top \tilde{\mathbf{q}}_i$ is compared with the actual rating $r_{u,i}$ to get the loss $L(r_{u,i}, \tilde{r}_{u,i})$. The gradient updates for $\tilde{\mathbf{p}}_u$ and $\tilde{\mathbf{q}}_i$ are just computed as in (2.8)

and (2.9). Then for each component of $\tilde{\mathbf{p}}_u$ (as well as $\tilde{\mathbf{q}}_i$), the k respective cells $\mathbf{C}$ are updated with their sign corrected gradients:

$$c_{j,h_j^u(u,l)} \leftarrow c_{j,h_j^u(u,l)} - \eta s_j^u(u,l)\left(\frac{\partial L(r_{u,i},\tilde{r}_{u,i})}{\partial \tilde{r}_{u,i}}\tilde{q}_{i,l} + \lambda \tilde{p}_{u,l}\right) \quad \forall l \in \{1...d\} \quad (4.3)$$

$$c_{j,h_j^i(i,l)} \leftarrow c_{j,h_j^i(i,l)} - \eta s_j^i(i,l)\left(\frac{\partial L(r_{u,i},\tilde{r}_{u,i})}{\partial \tilde{r}_{u,i}}\tilde{p}_{u,l} + \lambda \tilde{q}_{i,l}\right) \quad \forall l \in \{1...d\} \quad (4.4)$$

4.1.3 Approximation and equivalence

Our approach leads to approximations compared to the original online algorithm. When we update a quantity, *i.e.* $p_{u,l}$ or $q_{i,l}$, the sketching technique inherently modifies this quantity. It means that a write directly followed by a read access of the count sketch sees a modification of the update. This hurts our algorithm twice: In (4.3) and (4.4), not only $\tilde{p}_{u,l}$ and $\tilde{q}_{i,l}$ are noisy versions of what was maintained along previous iterations, but also $\tilde{r}_{u,i}$ is different from $\hat{r}_{u,i}$. The true update depends on the derivative of the loss $L(r_{u,i},\hat{r}_{u,i})$ in (2.8) and (2.9), a quantity which is not computed in our scheme. Instead, the sketching technique yields a reconstructed loss $L(r_{u,i},\tilde{r}_{u,i})$. Our system is no longer linear and it is difficult to see how these double approximations cumulate along with the updates.

An easier way to prove the soundness of our approach is to show that it is indeed equivalent to directly optimizing the count-sketch structure:

$$arg - min_{\mathbf{C}}\ \mathcal{R}_\lambda(\tilde{\mathbf{P}}(\mathbf{C}),\tilde{\mathbf{Q}}(\mathbf{C})), \quad (4.5)$$

where $\tilde{\mathbf{P}}(\mathbf{C})$ and $\tilde{\mathbf{Q}}(\mathbf{C})$ are the reconstructed latent vectors (as given by (4.1) and (4.2)). Let us consider a particular cell $c_{j,m}$ of $\mathbf{C}$. Its update triggered by the observation $< u, i, r_{u,i} >$ is:

$$\delta c_{j,m} = -\eta\left(\frac{\partial R_\lambda(\tilde{\mathbf{P}},\tilde{\mathbf{Q}})}{\partial \tilde{\mathbf{p}}_u}^\top \cdot \frac{\partial \tilde{\mathbf{p}}_u}{\partial c_{j,m}} + \frac{\partial R_\lambda(\tilde{\mathbf{P}},\tilde{\mathbf{Q}})}{\partial \tilde{\mathbf{q}}_i}^\top \cdot \frac{\partial \tilde{\mathbf{q}}_i}{\partial c_{j,m}}\right). \quad (4.6)$$

The expression of vector $\partial\tilde{\mathbf{p}}_u/\partial c_{j,m}$ is derived from the read access to the count sketch (4.1): Its l-th component equals $s_j(u,l)k^{-1}\mathbb{1}_{[h_j^u(u,l)==m]}$ (same for $\partial\tilde{\mathbf{q}}_i/\partial c_{j,m}$). In the end, this stems into the following update rules: $\forall l \in \{1,\cdots,d\}$

$$\begin{aligned}\delta c_{j,m} = -\frac{\eta}{k} \quad & \left(\left(s_j^u(u,l)\frac{\partial L(r_{u,i},\tilde{r}_{u,i})}{\partial \tilde{r}_{u,i}}\tilde{q}_{i,l} + \lambda\tilde{p}_{u,l}\right)\mathbb{1}_{[h_j^u(u,l)==m]}\right. \\ + \quad & \left.\left(s_j^i(i,l)\frac{\partial L(r_{u,i},\tilde{r}_{u,i})}{\partial \tilde{r}_{u,i}}\tilde{p}_{u,l} + \lambda\tilde{q}_{i,l}\right)\mathbb{1}_{[h_j^i(i,l)==m]}\right). \end{aligned} \quad (4.7)$$

We find back the same update rules as in (4.3) and (4.4) up to a factor k^{-1} due to the read access to the count-sketch based on the mean operator. However, this is not an issue as the gradient is at the end multiplied by the learning rate η.

4.1.4 Regularization based on sketch representation

The L_p norm based regularization is generally associated with matrix factorization, as it can be controlled and customized well to the needs. There are other methods to regularize like corrupting the input data. It is shown by Bishop et al in [Bis95] that training with corrupted data is equivalent to Tikhonov regularization. In our scheme, we observe that the sketch structure itself regularizes the learnt latent vectors. The error $\tilde{p}_{u,l} - p_{u,l}$ is due to the address hash collision among different elements. Thanks to the pairwise independence assumption, the error is independent of the cell location. Also the sign hash function $s_j(\cdot)$ makes sure that the expected error is centered: $E(\tilde{p}_{u,l} - p_{u,l}) = 0$. We surmise that these errors indeed provide regularization capabilities like corrupting the input data. We experimentally prove the claim in section 4.1.5.5.

We implement the regularization with a Tikhonov penalization: $\lambda \sum_{(u,i)} \|\tilde{\mathbf{p}}_u\|^2 + \|\tilde{\mathbf{q}}_i\|^2$. This has an interpretation in the equivalent problem (4.5): the variance of the count sketch estimation error is proportional to σ^2 (Section 4.2), which, in our case, is $\sum_{(u,i)} \|\tilde{\mathbf{p}}_u\|^2 + \|\tilde{\mathbf{q}}_i\|^2$. Our method thus aims at minimizing a combination of the error prediction and the error estimation of the latent vectors.

4.1.5 Experiments

In this section, we benchmark our method against regular online matrix factorization and feature hashing based factorization [KWS10] as it is a special case of our approach ($k = 1$).

4.1.5.1 Setup

Dataset

We use three publicly available datasets: Movielens1M and 10M [mov], EachMovie and Netflix. Data characteristics are in Table 4.1. The data is preprocessed and randomly partitioned into the training, validation and test sets with proportion $[0.8, 0.1, 0.1]$. Preprocessing includes mean correction and frequency based thresholding. User, item and global means are substracted from rating to remove user and item bias. Ratings with user/item frequency < 10 are removed from the test and validation sets. The same procedure is repeated 10 times to obtain 10 fold dataset.

Evaluation

We use root mean square error to measure the quality of recommendations. The error materializes the deviation of the predicted rating from the actual rating. This error is squared and averaged across all non-zero elements of the rating matrix to get mean

squared error:

$$RMSE(\mathbf{R}') = \sqrt{\frac{1}{\|\mathbf{R}'\|_0} \sum_{r_{u,i} \in \mathbf{R}'} (\tilde{\mathbf{p}}_u^\top \tilde{\mathbf{q}}_i - r_{u,i})^2}, \tag{4.8}$$

where $\mathbf{R}'$ is the restriction of $\mathbf{R}$ to the testing set.

Parameters

We compare the performance for various configurations (w, k) of the sketch and different latent factor dimensions. The sketch depth k is picked from $\{1, 4\}$ and the latent factor dimension d is chosen from $\{1, 2, 3, 4, 6, 8, 11, 16{,}23, 32\}$. We measure the space gain γ by the ratio of space that the regular factorization would need for the same dimension d to the space actually utilized by sketch based factorization. We vary γ within $\{1, 2, 2.83, 4, 5.66, 8, 11.31, 16, 23, 32\}$. We determine the sketch width based on the space gain, dimension d and sketch depth k:

$$w = \left\lceil \frac{(|\mathbb{U}| + |\mathbb{I}|)d}{\gamma k} \right\rceil. \tag{4.9}$$

We choose optimal parameters for learning rate η and regularization constant λ by a two stage line search in log-scale, based on validation set prediction score. We iterate for $T = 20$ epoch over the training set, before predicting on the testing set. Learning rate is scaled down at every iteration using the formula $\eta_t = \frac{\eta}{1+t/T}$.

Dataset	$\|\mathbb{U}\|$	$\|\mathbb{I}\|$	$\|\mathbf{R}\|$	rating
MovieLens 1M	6,040	3,952	1,000,209	1:5 (5)
EachMovie	61,265	1,623	2,811,718	1:6 (6)
MovieLens 10M	69,878	10,677	10,000,054	0.5:5 (10)
Netflix	480,136	17,167	96,649,938	1:5 (5)

Table 4.1: Dataset characteristics

4.1.5.2 RMSE comparison on various standard datasets

Table 4.2 reports RMSE with $d = 32$ and $\gamma = 1$, *i.e.* the three techniques needs the same space. We also initialize the parameters to small random values, sampled from same distribution for all the three algorithms. Best values (η, λ) are found by two stage linear search on validation set. The results are averaged over 10 fold dataset. Except for MovieLens 1M, the table shows that performance of sketch based factorization is similar to regular factorization while feature hashing is slightly worse.

Dataset	regular factorization	count sketch ($k = 4$)*	feature hashing
MovieLens 1M	0.873	0.876	0.906
MovieLens 10M	0.811	0.809	0.818
EachMovie	1.145	1.146	1.159
Netflix	0.854	0.855	0.862

Table 4.2: RMSE on various real datasets for $d = 32, \gamma = 1$

4.1.5.3 Variation of RMSE with factors size and space gain

We now evaluate the effect of space gain and dimension d on RMSE measure. Results are displayed as heatmaps for different sketch depth values in Figure 4.2 for MovieLens 1M and EachMovie datasets. The axes are in $\log_2$ scale, with horizontal and vertical axes representing d and γ. The best λ and η depends on γ and d as observed through 10 fold cross validation on individual configurations. The map with $k = 1$ corresponds to feature hashing and $k = 4$ corresponds to count sketch.

As expected, the RMSE increases when d decreases, as it impacts the representation capacity of the model, and when space gain γ increases because it implies smaller sketch width w and hence higher variance of estimated for $\tilde{\mathbf{p}}_u$ and $\tilde{\mathbf{q}}_i$ (due to more collisions in the count sketch). We can also observe that there is an improvement in RMSE score with higher k. This effect is more amplified for low γ values. As the trade-off is both ways and the improvement is almost diagonal, we can fix a desirable error bound and determine an optimal configuration (d, w).

4.1.5.4 Variation of RMSE with model size on dynamic updates

We first evaluate the convergence of RMSE along the number of epochs on training data. Figure 4.3 shows the result for MovieLens 10M dataset for the same setting as in Table 4.2. The three algorithms take the same space. We observe that convergence of count sketch is faster than the other two algorithms. Our explanation is the following: Every new observation $< u, i, r_{u,i} >$ stems into $2kd$ cell updates for the count sketch compared to $2d$ only cell updates for the other two methods, and this for the same space. We surmise then that count sketch factorization can be more suitable to collaborative filtering systems with dynamic updates as it has better convergence properties.

To simulate a dynamic environment, we do one pass over the training data and report the results on test data for best λ and η values. The RMSE scores are averaged on 10 fold data as described in the setup. The space of regular factorization is varied by increasing the dimension d_r from 1 to 32. The space of the two sketch based factorizations is matched with the former by fixing $d_s = 32$ but varying w according to (4.9). In other words, the space gain ranges from 32 to 1 while d_r goes from 1 to 32 to maintain the same space between the three methods.

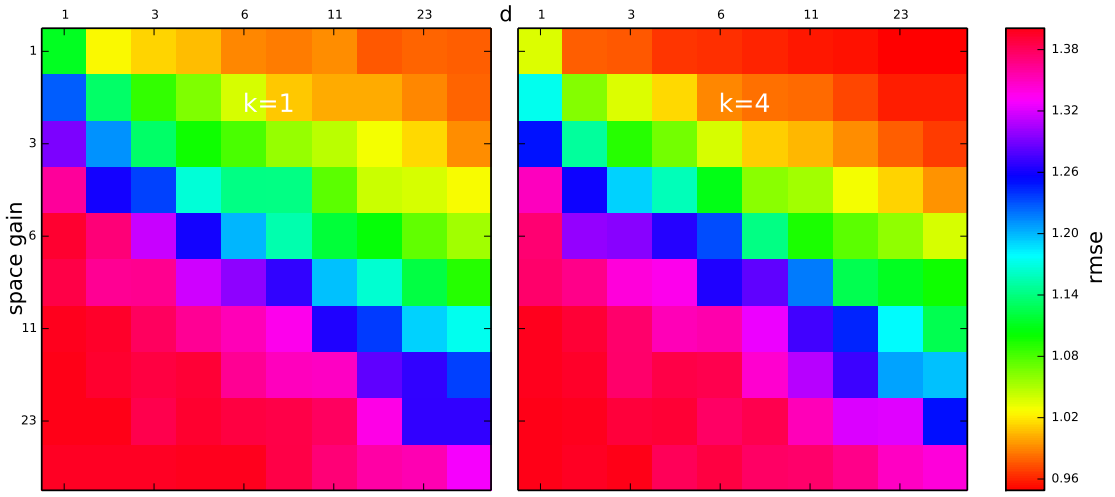

Figure 4.1: Heatmaps of RMSE for feature hashing (left) and count sketch (right) on MovieLens 1M

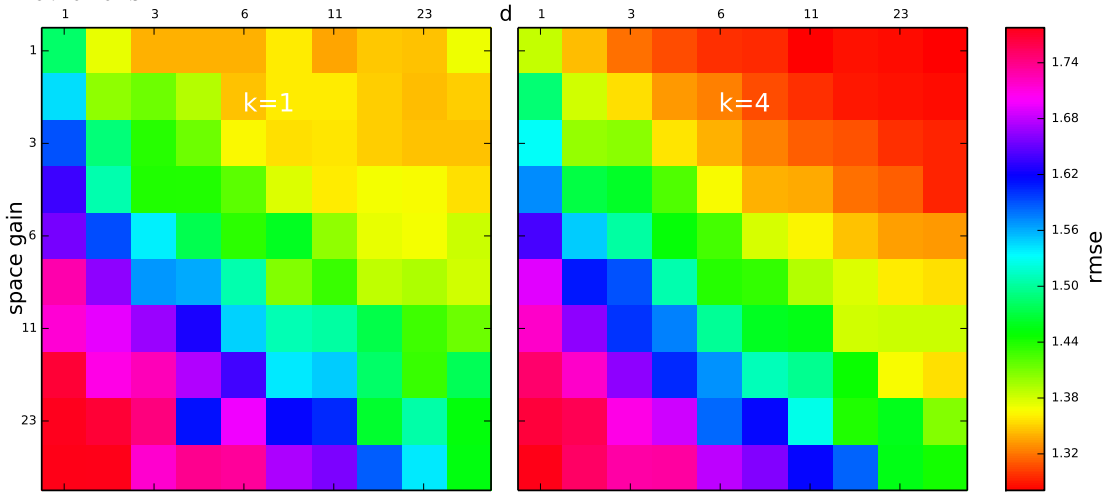

Figure 4.2: Heatmaps of RMSE for feature hashing (left) and count sketch (right) on EachMovie

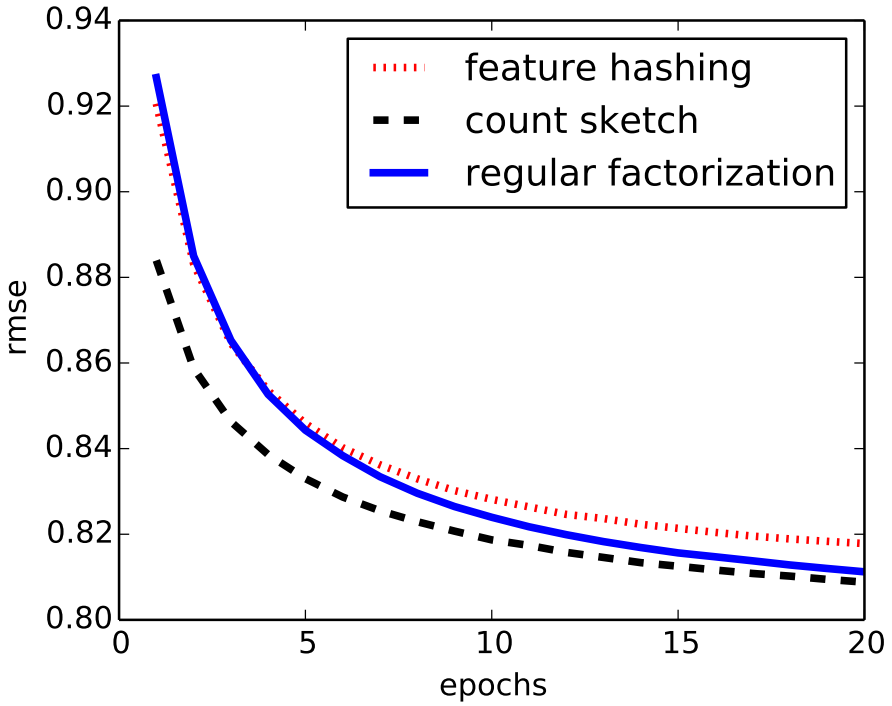

Figure 4.3: Convergence on MovieLens 10M.

Figure 4.4 shows that the performance of count sketch is better than other approaches for $d_r > 7$ (or $\gamma < 32/7$). This shows that count sketch factorization converges faster on dynamic data. We also observe that with increase in space, the performance of the other two techniques degrades, whereas count sketch saturates. This reveals that there are not enough data to train the other two techniques yielding some overfitting, whereas count sketch copes with k times more updates as above mentioned.

4.1.5.5 Regularizing effect of count sketch

We now study the effect of regularization parameter λ on RMSE. We use EachMovie dataset under the setup of Table 4.2 ($d = 32$, $\gamma = 1$). We vary λ from 10 to 0 in log scale. Figure 4.5 compares the performance. The three techniques share the following observations: The optimal λ is around 0.1 and the RMSE increases with λ beyond this value. When we decrease λ below the optimal value, the RMSE degrades and this is attributed to overfitting. The degradation of count sketch factorization is not as worse as the other two techniques. Even when the regularization is turned off ($\lambda = 0$), our scheme performs better than the other two, for the same model complexity. This shows that the 'noisy' sketch structure by itself provides some regularization capabilities.

The heatmap of Figure 4.6 represents the optimal lambda values for various (d, γ) pairs on MovieLens 1M dataset. The optimal value is often lower than 10^{-2}, except

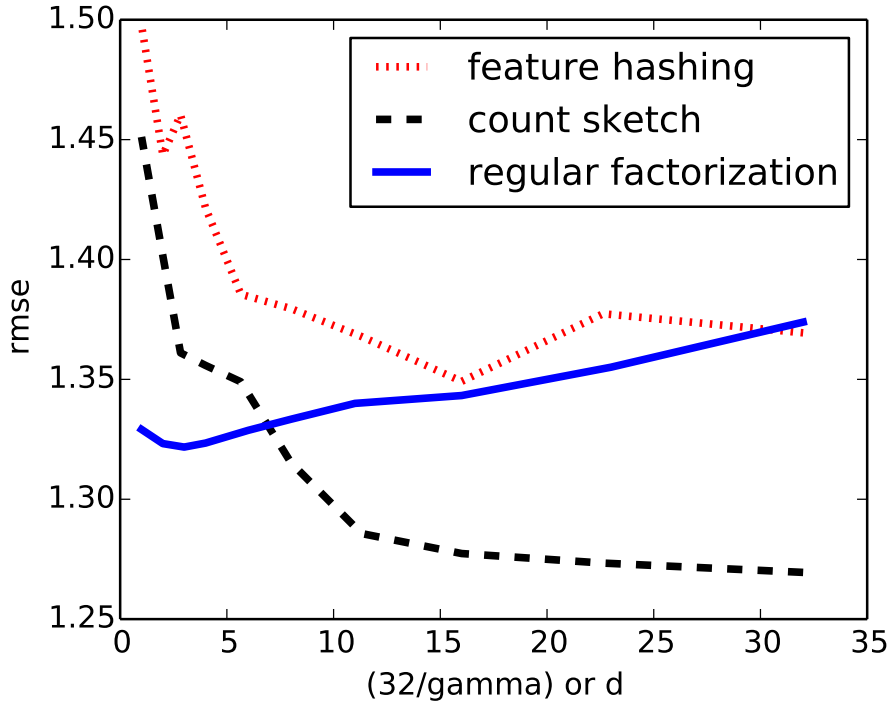

Figure 4.4: Dynamic setting on EachMovie.

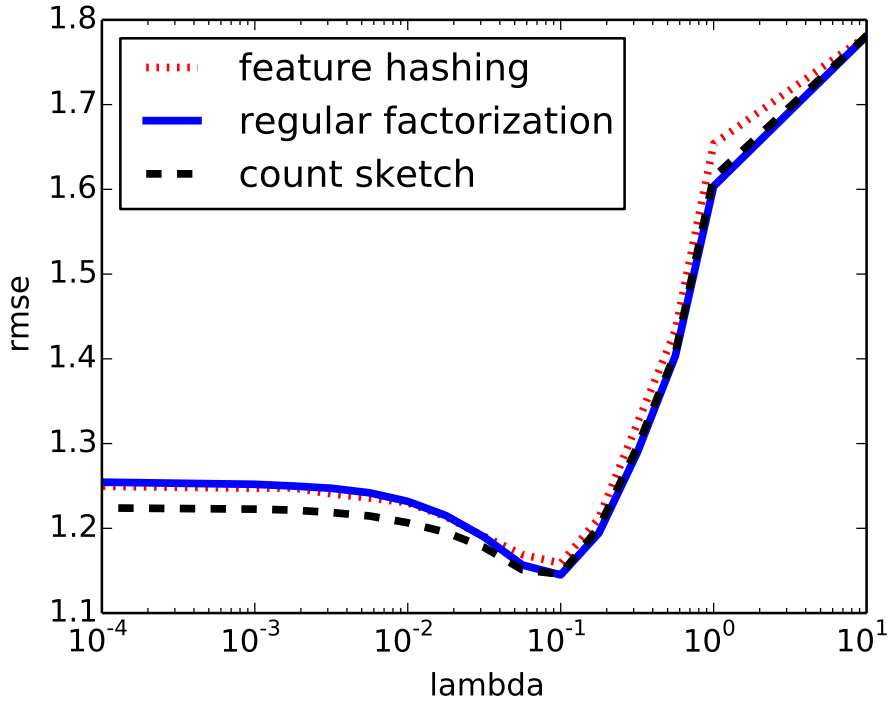

Figure 4.5: RMSE w.r.t. λ, EachMovie

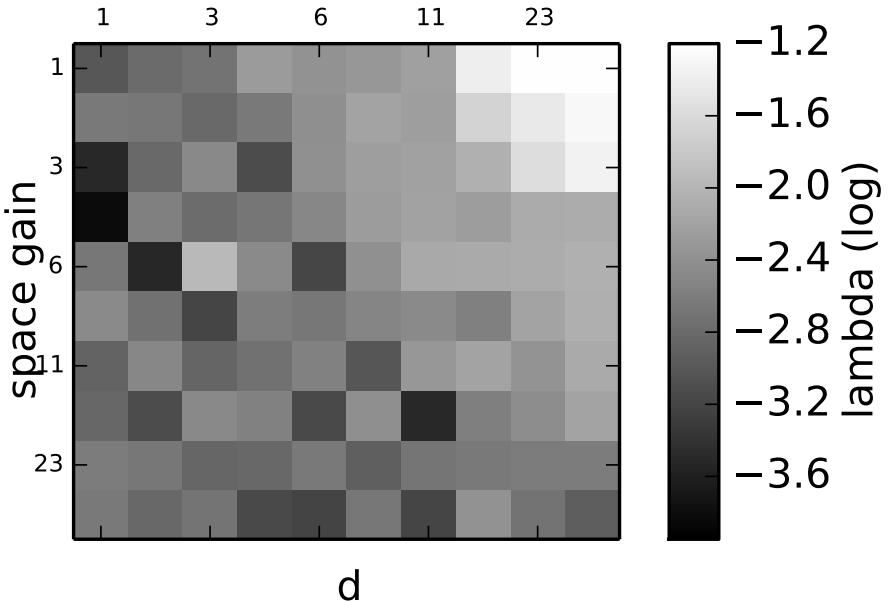

Figure 4.6: Best λ w.r.t. (d, γ), MovieLens 1M

in the top right corner, where γ is small while d is big. This setting indeed ensures superfluous parameter space, which does require stronger regularization to avoid overfitting. The λ value diminishes with d: a smaller model requires less regularization. An interesting observation is that λ lowers with increase in γ and it is true even for a fixed d. This increases address hash collisions and hence the variance of the count sketch estimation (Sect. 4.1.4) which helps model generalization like when learning on noisy data.

4.2 Privacy aware matrix factorization using sketching techniques

We demonstrated the scalability and regularization properties of our sketch based factorization and also observe that it is inherently random. It is also known that adding independent noise to a statistic of interest enforces privacy [AS00], hence supporting privacy preserving characteristic of our approach. It is then imperative to quantify that privacy level and also the relationship between privacy and the utility of the system. As user profile data is sensitive, assessing the privacy guarantees becomes crucial. We start with understanding the noise characteristics of count sketch in a streaming setup.

Count sketch error analysis: The accuracy of the estimation is related to the size of the count sketch [Cor11]. Note that if we query index e just before and after its update, the difference of the approximates is the true update:

$\tilde{v}_e^{(t+1)} - \tilde{v}_e^{(t)} = \delta v_e$. However, updating the e-th quantity might have modified the others due to collision. In the j-th row of the count sketch, one entry has been modified by $\pm\delta v_e$ (see (2.15)) whereas the $w-1$ others remained the same. In other words, the entries of $\mathbf{C}$ have been modified by random variables i.i.d. according to the p.m.f. $(1-w^{-1})\partial_0 + (2w)^{-1}(\partial_{\delta v_e} + \partial_{-\delta v_e})$, where ∂_a represent the Dirac distribution on $x = a$. The expectation is zero and the variance $(\delta v_e)^2/w$. This implies that the update of the e-th quantity adds on all the others $\tilde{v}_{e'}$, $e' \neq e$, a centered noise of variance $(\delta v_e)^2/wk$. Overall, the estimate based on the mean operator is unbiased with variance σ^2/wk, where $\sigma^2 = \sum_{e\in[N]}(\delta v_e)^2$. For a given (w,k), the accuracy decreases with N because the variance of the estimate increases with σ^2. In other words, the representational capacity N of count sketch can be controlled by varying (w,k).

We thus note that the noise introduced by our sketch structure our factor model parameters is equivalent to adding random noise centered with variance of σ^2/wk to the latent factors. If we assume that the noise to be gaussian, we observe that our approach is similar to a Bayesian learning enforcing differential privacy [LWS15].

The following section describes our novel method of altering the ϵ parameter of differential privacy by means of clipping instances of training data that are *too revealing*.

4.2.1 Data clipped factorization

The data clipped factorization is a modification to the sketch factorization described in section 4.1.2. The modification is primarily in the update phase, where in the latent factors are updated only if the estimated loss between observed and predicted rating is bounded by ϵ. We describe the approach in detail anyway for readability purpose.

When a new tuple $< u, i, r_{u,i} >$ arrives, the count sketch is first queried to approximately reconstruct user and item latent vectors. Both user ID u and component index l, $1 \leq l \leq d$, are used as inputs to the k pairs of address and sign hash functions to get a mean estimate of the vector component as follows (the same holds for item):

$$\tilde{p}_{u,l} = \frac{1}{k}\sum_{j=1}^{k} s_j^u(u,l)\cdot c_{j,h_j^u(u,l)}, \quad \forall l \in \{1,\cdots,d\}, \tag{4.10}$$

$$\tilde{q}_{i,l} = \frac{1}{k}\sum_{j=1}^{k} s_j^i(i,l)\cdot c_{j,h_j^i(i,l)}, \quad \forall l \in \{1,\cdots,d\}. \tag{4.11}$$

The estimated rating $\hat{r}_{u,i} = \tilde{\mathbf{p}}_u^\top \tilde{\mathbf{q}}_i$ is compared with the observed $r_{u,i}$ to get the loss $L(r_{u,i}, \hat{r}_{u,i})$.

If $L(r_{u,i}, \hat{r}_{u,i}) \leq \epsilon$, the gradient updates for $\tilde{\mathbf{p}}_u$ and $\tilde{\mathbf{q}}_i$ are just computed as in (2.8) and (2.9). Then for each component of $\tilde{\mathbf{p}}_u$ (as well as $\tilde{\mathbf{q}}_i$), the k respective cells $\mathbf{C}$ are updated with their sign corrected gradients:

$$c_{j,h_j^u(u,l)} \leftarrow c_{j,h_j^u(u,l)} - \eta s_j^u(u,l) \nabla_{u,l} \mathcal{R}_\lambda(\tilde{\mathbf{P}}, \tilde{\mathbf{Q}}), \tag{4.12}$$

$$c_{j,h_j^i(i,l)} \leftarrow c_{j,h_j^i(i,l)} - \eta s_j^i(i,l) \nabla_{i,l} \mathcal{R}_\lambda(\tilde{\mathbf{P}}, \tilde{\mathbf{Q}}). \tag{4.13}$$

$\nabla_{u,l} \mathcal{R}_\lambda(\tilde{\mathbf{P}}, \tilde{\mathbf{Q}})$ denotes the l-th component of the gradient: $(r_{u,i} - \mathbf{p}_u^\top \mathbf{q}_i)\mathbf{q}_i + \lambda \mathbf{p}_u$, and similarly for the item gradient.

Otherwise (*i.e.* $L(r_{u,i}, \hat{r}_{u,i}) > \epsilon$), the cells are not updated at all. This is equivalent to not taking into account this particular observation. Our rationale behind the clipping is that an update corresponding to large deviation carries more information and enforces radical changes to the model parameters. Hence if such an update is taken into account, it will lead to information leak for that event of the user. It should be noted that we reject the observation for that particular epoch and not forever. Hence if the same observation fits well into the model at a later stage (epoch), it will be taken into consideration for updating the corresponding model parameters.

4.2.2 Differential privacy

Our algorithm inherently adds noise on the updates thanks to the count sketch. The noise perturbation and estimation mechanism makes it similar to the bayesian matrix factorization, described in section 2.2.5.5. It is shown in [WFS15, LWS15] that Baysian inference is differentially private, as described in section 2.4.6.3. In particular, we follow the real data source definition described in section 2.4.9.4, as user rating is in Real scale. Though our setup looks exactly like the SGLD (2.13) (2.14), this is not the case. We can find the following differences.

1. When updated, latent factors related to observed $r_{u,i}$ are not corrupted by noise, whereas all the others are.
2. The noise induced by the count sketch has a variance equalling δ^2/wk where δ is the last update (see Sec. 2.3.2.5). Therefore, this variance is proportional to η^2/wk and not η as in the SGLD algorithm.
3. This noise results from $2d$ latent factors updates only therefore it is certainly not Gaussian distributed.

The differential privacy is enforced by clipping the log-likelihood s.t.

$$|\log p(r_{u,i}|\mathbf{P}, \mathbf{Q})| = \begin{cases} (r_{u,i} - \hat{r}_{u,i})^2/2 & \text{if } (r_{u,i} - \hat{r}_{u,i})^2 \leq \epsilon \\ \epsilon/2 & \text{otherwise} \end{cases}$$

This enables ϵ-DP as shown in [WFS15, LWS15], except that this no longer defines a valid conditional probability.

4.2.3 Kullback-Leibler divergence to measure privacy

Though differential privacy provides a concrete upper bound on the information leak, it does not describe any other privacy preserving property of the system. Also the perturbation requirements are, at times, too stringent and idealistic to be used in a large scale realistic setup. Metrics such as (ϵ, δ)-differential privacy [DKM+06], concentrated differential privacy [DR16], heterogenous differential privacy [AGK15] were proposed to relax the perturbation requirements owing to practical constraints. Still the ϵ parameter in all these measures is a control parameter and not a measured variable. Hence to provide an alternate view of the information leakage, we propose to estimate the expected information loss as a privacy measure.

In particular, we use Kullback-Leibler divergence to gauge privacy. The KL divergence gives the expected amount of information 'leaked' about the fact that user u submitted his rating about item i. This divergence quantifies the difference between the probability distributions of the prediction for observed ratings (the training set $\mathcal{D}_{tr}$) and non observed ratings (the testing set $\mathcal{D}_{te}$).

$$KLD = \mathbb{E}\left|\frac{\log \mathbb{P}[\hat{r}_{u,i} - r_{u,i} | r_{u,i} \in \mathcal{D}_{tr}]}{\log \mathbb{P}[\hat{r}_{u,i} - r_{u,i} | r_{u,i} \in \mathcal{D}_{te}]}\right| \tag{4.14}$$

A low divergence means that the predicted rating $\hat{r}_{u,i}$ is statistically similar whether $r_{u,i}$ was used in training or not. An attacker observing $\hat{r}_{u,i}$ and *even knowing* $r_{u,i}$ can not decide whether this rating was submitted to the recommendation system. To this aim, we experimentally observed that the prediction error $(\hat{r}_{u,i} - r_{u,i}) \sim \mathcal{N}(m, s^2)$. We measure (m_{tr}, s^2_{tr}) over the training set and (m_{te}, s^2_{te}) over the testing set, and compute

$$KLD = \frac{1}{2}\left(\frac{s^2_{tr}}{s^2_{te}} + \frac{(m_{tr} - m_{te})^2}{s^2_{te}} - 1 + \log\frac{s^2_{te}}{s^2_{tr}}\right). \tag{4.15}$$

4.2.4 Experiments

This section evaluates the claimed benefits of our approach. We first experimentally study its regularization capabilities. Then we analyze the privacy-utility trade-off and compare to regular factorization (without privacy).

4.2.4.1 Setup

Dataset

We use two publicly available datasets: Movielens1M [mov] and EachMovie. Data characteristics are in Table 4.1. The data preprocessing steps are similar to the description in section 4.1.5.1.

Evaluation

We use root mean square error to measure the quality of recommendations as described in equation 4.8 We use Kullback-Leibler divergence to gauge privacy as described in section 4.2.3 and estimate using the expression (4.15).

Parameters

We compare the performance for various configurations (w, k) of the sketch and different latent factor dimensions. The sketch depth k is picked from $\{1, 4\}$ and the latent factor dimension d is chosen from $\{8, 16, 32\}$. We measure the space gain γ by the ratio of space that the regular factorization would need for the same dimension d to the space actually utilized by sketch based factorization. We vary γ within $\{1, 2, 4\}$. We determine the sketch width based on the space gain, dimension d and sketch depth k as in equation 4.9 We choose optimal parameters for learning rate η and regularization constant λ by a two stage line search in log-scale, based on validation set prediction score. We iterate for $T = 100$ epochs over the training set, before predicting on the testing set, measuring $RMSE(\mathbf{R}')$ and KLD. Learning rate is scaled down using the formula $\eta_t = \frac{\eta}{1+b \cdot t/T}$.

4.2.4.2 Privacy utility tradeoff

In this section, we compare the variation of RMSE with respect to parameter ϵ as shown in figure 4.7. We vary ϵ in the range of $\{2, 4, 8, \infty\}$ and benchmark it against regular factorization. We have ϵ on x-axis and RMSE on y-axis, plotted for each latent factor dimension $d \in \{8, 16, 32\}$. As expected, lowering the privacy by increasing ϵ improves the utility, *i.e.* decreases RMSE. The improvement is significant in the lower range of ϵ and drops gradually as we increase ϵ further. When the epsilon is above 16 the RMSE is close to regular factorization.

4.2.4.3 KL divergence as a privacy measure

We take the role of an attacker willing to know whether a particular rating was used in the training. We use KL divergence to measure the amount of information 'leaked' by getting access to its prediction. We plot the variation of KL divergence with ϵ for various values of latent factor dimensions. We have ϵ on x-axis and KL divergence on y-axis, plotted for each latent factor dimension $d \in \{8, 16, 32\}$. Figure 4.8 shows that KL divergence increases along with the targeted level of privacy ϵ. The KL divergence of regular factorization without any differential privacy mechanism is higher for most of the cases. This supports our approach. It is also clear from the figure that compact models (higher γ or lower d) produce recommendations leaking less information compared to bigger one. The KL divergence which is a measurement on average is indeed much smaller than the target ϵ which is a guaranty on the worst case.

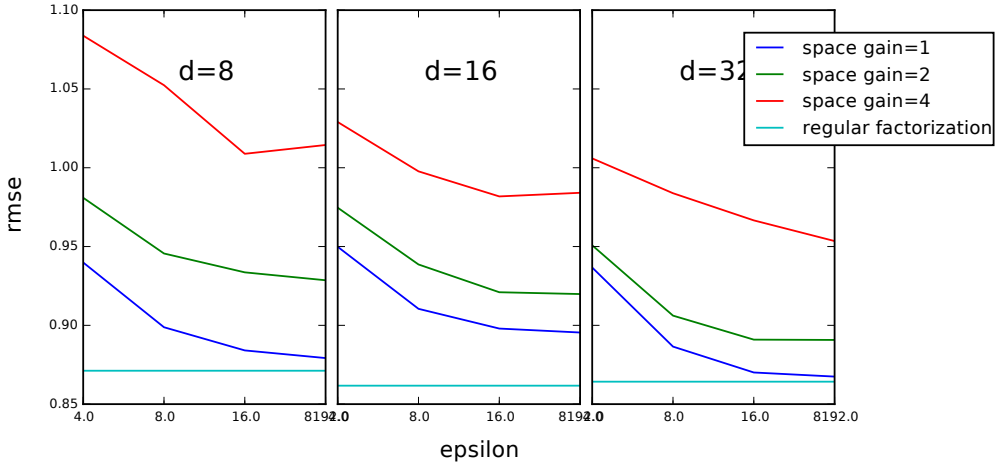

Figure 4.7: RMSE w.r.t. ϵ, Movielens

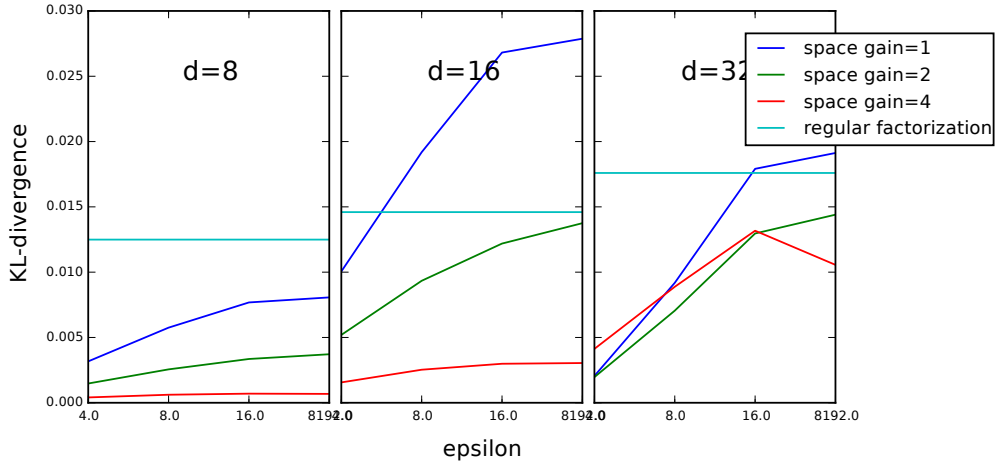

Figure 4.8: KL divergence w.r.t. ϵ, Movielens

4.3 Conclusion

The memory intensive nature of matrix factorization techniques calls for efficient representations of the learned factors. This work investigated the use of count sketch for storing the latent factors. Its compact and controllable representation makes it a good candidate for efficient storage of these parameters. We show that the optimization of the latent factors through the count sketch storage is indeed equivalent to finding the optimal count sketch structure for predicting the observed ratings. Experimental evaluations show the trade-off between performance and space and also reveal that count sketch factorization needs less data for training. This property is very useful in dynamic setting.

We also demonstrate that sketching techniques can be used to preserve privacy by taking advantage of their inherent randomness. This is in contrast to conventional techniques which uses special mechanism to achieve the same. We experimentally validate our approach using standard datasets and also define a new privacy measure using Kullback-Leibler divergence. We conclude that the scalability and privacy preserving nature of our sketch based factorization are more suited to applications which deal with large scale data of sensitive nature.

Chapter 5

Improving retrieval efficiency of recommender systems using LSH

Outline

Recommender systems require efficient retrieval methods to find relevant items for a given user based on his preferences and past consumption. The retrieval is achieved by searching the user profile against item profiles to provide relevant suggestions, where the profiles are learnt from past consumption data. The search process is often modeled as nearest neighbor problem. Many nearest neighbor search algorithms rely on encoding real vectors into binary vectors. The most common strategy projects the vectors onto random directions and takes the sign to produce so-called sketches. This chapter discusses the sub-optimality of this choice, and proposes a better encoding strategy based on the quantization and reconstruction points of view. The section 5.2 describes the sub-optimality of project-sign methods, followed by section 5.3, that describes our approach. An optimal coding scheme from reconstruction, is to search for the best binary code on the binary space, which quickly becomes infeasible for higher dimensions. Our quantization optimized LSH (qoLSH), iteratively optimizes the binary code obtained from project-sign method to minimize the reconstruction error in a realistic setup. We observe that it strikes a nice balance between efficiency and utility making it a practical technique of choice. We also propose a novel asymmetric estimator for the cosine similarity. Similar to previous asymmetric schemes, the query is not quantized and the similarity is computed in the compressed domain. Our contributions lead to improve the quality of nearest neighbor search with binary codes. Our experimental results, described in section 5.4 substantiates the improvement against a recent encoding technique called anti-sparse encoder.

The growing popularity of recommender systems posed new requirements that have to be solved to enhance utility. Two such broad requirements are effective modeling and prediction. The modeling challenges are addressed by customized machine learning techniques, some of which are discussed in the previous chapters. The other unaddressed problem is prediction, which is not given its due importance, when compared with modeling. Prediction often involves retrieval of relevant items from a huge pool of available item space and ranking them according to the relevance. The prominent latent factor model based recommender systems embed user and items in a common real vector space such that the distance/similarity defined in the vector space approximates the similarity relationship among user and items. Hence finding relevant items is reduced to performing a nearest neighbor search on this latent vector space, which is a well studied problem with many approaches. Nearest neighbors search has wide range of application in many domains other than recommender systems, such as multimedia indexing, pattern recognition, computational geometry, etc. On very large high dimensional datasets, finding exact nearest neighbors is time consuming and impractical, leading to approximation techniques. From a large collection of vectors in a high dimensional space, the approximate most similar search aims at extracting the most similar vectors to a query. Locality Sensitive Hashing is one of the popular family of approximate nearest neighbor techniques, with well defined properties and sound theoretical justification. Locality sensitive sketches are defined as an approximation for some of the popular distance measures and one such is Cosine. A technical background of LSH was presented in section 2.3.3.2 and Cosine sketches in section 2.3.3.5.

These approximate nearest neighbor search techniques are broadly classified into two categories: partition based and distance approximation based. The first class of methods work on to reduce the search space by partitioning the vectors into clusters, so that the query vector can be searched only on a subset of data. Often the size of the subset is very low compared to the size of the dataset, which improves the retrieval efficiency. To improve the retrieval chances, these techniques maintain multiple independent indices, multiplying the space requirements in favor of better recall. As the dimensionality increases, the partitioning is inefficient which worsens the retrieval complexity to linear. Also, finding optimal partitions becomes a non-trivial problem in high dimensional space. Many times, the original vectors have to be stored separately for detailed re-ranking, which increases the query latency and space utilization drastically.

An alternative approach is to approximate the similarity, to speedup the calculation. It means the query complexity is linear to the database size, but similarity computation is done much faster, so that queries can be answered in realistic time. One such method called Hamming Embedding, designs a function that maps vectors in $\mathbf{R}^d$ to binary sketches in $\mathbf{B}^L$ such that the Hamming distance between sketches estimates the similarity between vectors. In recent papers [TFW08a, JDS08, WTF09a, PLSP10, JB11], LSH is no longer considered in the context of probe algorithms, but employed as a Hamming

Embedding. To the best of our knowledge, Charikar [Cha02] was the first to estimate the angle between two Euclidean vectors based on their LSH sketches. Project and sign based method is one of the simplest encoding scheme that approximates cosine distance in hamming space by projecting on random hyperplane and retaining the sign information of the projection. Section 2.3.3.5 provides the necessary technical background to understand Cosine sketches. Since the individual sign information is stored as a single bit, cosine sketches are very compact, making storage and retrieval much easier. Also they are simple in construction and easy to encode and decode. Another advantage of project-sign based cosine sketches is that original vector can be reconstructed from the binary code, unnecessitating separate storage of original vectors. Cosine sketches are often used along with asymmetric similarity estimation.

The asymmetric scheme computes similarity measurements from the query vector and the database sketches. In other words, the sketch of the query is not processed. We would like to find a new design with the following properties: (i) it is an Hamming embedding for the similarity based on the cosine between two vectors, (ii) it allows a simple reconstruction of the original vector from its sketch. The first property is crucial for efficiently finding a subset of the database containing similar vectors while the second property yields to an asymmetric scheme re-ranking these vectors by computing a better estimate from the query and their reconstructions. Yet, Section 5.2 outlines the suboptimalities of LSH from the viewpoint of reconstruction. In a previous paper, we have already proposed a design fulfilling the two properties but its complexity prevents its application to high dimensional space and/or large scale database. This is the reason why we propose in Section 5.3 a simple modification of LSH to boost its reconstruction ability while maintaining its efficiency. Section 5.4 shows experimental results demonstrating the good performances of our two-step approximate search which strikes a better trade-off between complexity and quality of search when compared to previous schemes.

5.1 Cosine sketches setup

This section briefly describes the setup of project-sign based cosine sketches, followed by the design of the project and sign hash function for cosine sketch and then asymmetric distance estimation using these sketches. Let us assume the query and database vectors are unit-normed and in $\mathbb{R}^d$ vector space. Let the L projection vectors $\{\mathbf{w}_j\}_{j=1}^{L}$ be iid sampled from unit sphere. For a vector $\mathbf{x}$, each projection $\mathbf{w}_j^\top \mathbf{x}$, followed by sign produces a bit:

$$b_j(\mathbf{x}) = \mathsf{sign}(\mathbf{w}_j^\top \mathbf{x}). \tag{5.1}$$

The sketch of $\mathbf{x}$ is just the concatenation of these bits:

$$\mathbf{b}(\mathbf{x}) = [b_i(x)]_{i=1}^{L} \tag{5.2}$$

Comparison of two binary codes $\mathbf{b}(\mathbf{x})$ and $\mathbf{b}(\mathbf{y})$ is nothing but hamming distance between the two bit vectors: $d_h(\mathbf{b}(\mathbf{x}), \mathbf{b}(\mathbf{y})) = \sum_{j=1}^{L} b(\mathbf{x})_j \neq b(\mathbf{y})_j$. Therefore, ranking vectors by increasing order of the hamming distance between their sketches and the sketch of a query approximates the ranking by increasing angle or decreasing cosine similarity.

5.1.1 Hash function design

The performance of sketches depends on the design of the hash functions. The random projections proposed by Charikar [Cha02] are widely used for cosine similarity, however they do not offer the best results. We distinguish two cases.

⋆ $L \leq d$**:** A set of orthogonal vectors yields better results than random projections [JDS08, JFF12]. The methods performing a PCA rotation learned in a training set, such as spectral hashing [WTF09a], implicitly use orthogonal vectors.

⋆ $L > d$**:** It is no longer possible to generate L orthogonal projections. The L projection vectors form an over-complete frame $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_L]$ [GVT98]. A tight frame satisfying $\mathbf{W}.\mathbf{W}^\top \propto \mathbf{I}_d$ is better than random projections [JFF12, SVZ13]. Another concurrent strategy [JLY+12] takes the union of subsets of orthogonal vectors (called *super-bits*). This construction has not been compared to an uniform tight frame.

Another track of research aims at optimizing the projection directions in order to better reconstruct the small distances [KD09a]. Similarly, a rotation matrix is optimized to balance the variance on the different components [JDSP10, GL11] so that each bit gives the same approximation error. These works mainly differ by the way the optimization is carried out.

5.1.2 Asymmetric scheme with sketches

The main interest of sketches is their compactness. In a typical scenario, they allow storing a representation of millions to billions vectors in memory. However, the memory constraint is not critical for the query, as this one is processed online.

This observation motivates the use of asymmetric methods [DCL08, PLSP10, JDSP10, JJG11, JDS11a], in which databases vectors are encoded into short sketches but the query is kept uncompressed to avoid quantization error. The first proposal considered the Euclidean distances from the query $\mathbf{y}$ to separating hyperplanes to weight the Hamming distance [DCL08]:

$$d_a(\mathbf{y}, \mathbf{b}(\mathbf{x})) = \sum_{j=1}^{L} (\mathbf{y}^\top \mathbf{w}_j).b_j(\mathbf{x}) \tag{5.3}$$

5.2 Suboptimality of project and sign

Instead of considering the analysis which maps $\mathbf{x}$ into $\mathbf{b}(\mathbf{x})$, we take a look at the synthesis, *i.e.* the reconstruction of the direction pointed by a vector from its sketch. From now on, we restrict to vectors on the hypersphere: $\|\mathbf{x}\| = 1$. We only consider a very simple reconstruction:

$$\hat{\mathbf{x}} \propto \sum_{j=1}^{L} b_j(\mathbf{x})\mathbf{w}_j = \mathbf{W}\mathbf{b}(\mathbf{x}). \tag{5.4}$$

The proportionality constant is set such that $\|\hat{\mathbf{x}}\| = 1$. In the sequel, we exclude degenerated cases s.t. $\sum_{j=1}^{L} b_j \mathbf{w}_j = \mathbf{0}$.

5.2.1 'project and sign' is not a good quantizer

The new point of view of reconstruction/quantization stems in an interesting question about the binarization strategy. Formally, we have defined a codebook $\mathcal{C}$ comprising at most 2^L distinct centroids over the hypersphere. Does the centroid $\mathbf{c} \propto \mathbf{W}\mathbf{b}(\mathbf{x})$, induced by the selected sketch $\mathbf{b}(\mathbf{x})$, provides the best possible choice from a reconstruction point of view? The best centroid is the one maximizing the co-linearity to the input unitary vector $\mathbf{x}$ as

$$\mathbf{c}^\star(\mathbf{x}) = arg - max_{\mathbf{c}\in\mathcal{C}} \mathbf{x}^\top \mathbf{c} \tag{5.5}$$

$$\propto \mathbf{W} arg - max_{\mathbf{b}\in\mathbb{B}^L} \frac{\sum_{j=1}^{L} b_j \ \mathbf{x}^\top \mathbf{w}_j}{\left\| \sum_{j=1}^{L} b_j \mathbf{w}_j \right\|}. \tag{5.6}$$

Let first consider the case of an orthonormal set of vectors: $\mathbf{W}^\top\mathbf{W} = \mathbf{I}_L$. The denominator is then constant and the optimum is therefore obtained when b_j and $\mathbf{x}^\top\mathbf{w}_j$ have the same sign. Therefore, the project and sign method is optimal for orthogonal frames with respect to quantization.

For the case $L > d$, the frame cannot be orthogonal and the above property does not hold, meaning that the best reconstruction may *not* take the sign of $\mathbf{x}^\top\mathbf{w}_j$.

⋆ **Example:** Consider the frame

$$\mathbf{W} = [\mathbf{w}_1\mathbf{w}_2\mathbf{w}_3] = \begin{bmatrix} 1 & 0 & \cos\frac{\pi}{3} \\ 0 & 1 & \sin\frac{\pi}{3} \end{bmatrix} \tag{5.7}$$

The vector $\mathbf{x} \propto \mathbf{w}_1 + \mathbf{w}_2 - \mathbf{w}_3$ happens to have a sketch $\mathbf{b}(\mathbf{x}) = [1, 1, 1]$, whereas the best centroid is obviously $\mathbf{c}^\star(\mathbf{x}) \propto \mathbf{W}.[1, 1, -1]^\top = \mathbf{x}$. In other terms, projecting and taking the sign is suboptimal in this case. Indeed, the function $\mathbf{x} \mapsto \mathbf{b}(\mathbf{x})$ is not necessarily surjective as some sketches might never be selected. This implies a loss of capacity in

the encoding scheme: although computed on L bits, the entropy of the sketches is lower than L bits.

At this stage, we mention that this problem is not solely due to the choice of the frame operator, but to the quantization procedure as well. Selecting the closest centroid in $\mathcal{C}$ to the input vector yields a better quantization as reported in Section 5.4. Yet this quantization is not possible for large values of L, for which browsing the whole set of centroids is not tractable.

5.2.2 Spread representations

These observations motivate a recent approach [JFF12] for a better encoding strategy based on spread representations [Fuc11]. It reduces the quantization error underpinning the sign function.

The "anti-sparse coding" strategy first looks at

$$\mathbf{v}^\star(\mathbf{x}) = \arg \min_{\mathbf{v}\in\mathbf{R}^L:\mathbf{W}\mathbf{v}=\mathbf{x}} \|\mathbf{v}\|_\infty . \tag{5.8}$$

This goal resembles the objective of sparse coding, except that the ℓ_0 norm is replaced by ℓ_∞. As a result, instead of concentrating the signal representation on few components, anti-sparse coding has the opposite effect: It tends to spread the signal over all components, whose magnitude is comparatively less informative.

Interestingly, $L - d + 1$ components of $\mathbf{v}^\star(\mathbf{x})$ are stuck to the limit, *i.e.* , equal to $\pm\|\mathbf{v}^\star(\mathbf{x})\|_\infty$. As a result, this vector can be seen as a "pre-binarized version". The subsequent binarization to $\mathbf{b}(\mathbf{x}) = \mathsf{sign}(\mathbf{v}^\star(\mathbf{x}))$ introduces less quantization loss than with the regular "project and sign" approach.

The main problem of anti-sparse coding is its low efficiency: Encoding a vector requires several matrix inversions, and the complexity strongly depends on the vector dimensionality [JFF12]. Although this encoding step is done offline, it remains the bottleneck in practical setups involving billions of descriptors and is not tractable for high-dimensional vectors.

5.3 Our approach: quantization-optimized LSH (qoLSH)

This section explains how we improve the cosine sketch detailed in Section 2.3.3.5 by adopting a quantization point of view. The section 5.2 illustrated the suboptimality of the "project and sign", from a reconstruction point of view, and the prohibitive cost of the optimal strategy due to the exponential increase in the number of centroids $|\mathcal{C}|$ with L.

Matrix W: We only use tight frames, as they generally offer better performance in this context [JFF12, SVZ13]. We randomly draw a $L \times d$ matrix with i.i.d. Gaussian

entries. Then we compute its QR decomposition and set $\mathbf{W}$ as the first d rows of $\mathbf{Q}$, so that

$$\mathbf{W}.\mathbf{W}^\top = \mathbf{I}_D. \tag{5.9}$$

Computation of the sketch: Our approach is to alter the cosine sketch $\mathbf{b}$ of (5.1) in such a way that it decreases the reconstruction error. This way the Hamming distance between sketches still approximate the angle between their real vector counterparts. We alter the cosine sketch $\mathbf{b}$ by flipping sequentially individual bits that improves the reconstruction error.

For a given vector $\mathbf{x}$, its sketch $\mathbf{b}(\mathbf{x})$ and reconstructed vector $\hat{\mathbf{x}} = \mathbf{W}\mathbf{b}(\mathbf{x})$, we define the objective function as:

$$\mathcal{L}(\mathbf{b}) = \frac{\mathbf{x}^\top \hat{\mathbf{x}}}{\|\hat{\mathbf{x}}\|} \tag{5.10}$$

We start with sketch $\mathbf{b}_{(0)}$ of (5.1) and compute the reconstruction vector $\hat{\mathbf{x}}_{(0)}$. By flipping the j-th bit in $\mathbf{b}_{(0)}$, we get a new sketch $\mathbf{b}_{(1j)}$ and reconstruction vector

$$\hat{\mathbf{x}}_{(1j)} = \hat{\mathbf{x}}_{(0)} - 2b_j\mathbf{w}_j. \tag{5.11}$$

For L such bits in $\mathbf{b}$, we get L possible $\mathbf{b}_{(1j)}$ sketches. Out of L such sketches, we choose the one that maximizes the improvement in the objective function and call it $\mathbf{b}_{(1)}$, *i.e.*

$$\mathbf{b}_{(1)} = arg - max\mathcal{L}(\mathbf{b}_{(1j)}). \tag{5.12}$$

We now take $\mathbf{b}_{(1)}$ as the base sketch and find the next best bit to flip, which gives a new sketch $\mathbf{b}_{(2)}$, as described before. We continue this iteration until no bit flipping improves the objective function, or we reach a predefined number of iterations, say M.

The discrete nature of $\mathbf{b}$ makes exact optimization impossible. Changing a single coordinate at a time is suboptimal but it has a limited complexity.

Asymmetric scheme: The estimation of the similarity is based on the cosine of the angle between the query $\mathbf{y}$ and the reconstructed vector:

$$\cos(\mathbf{y}, \hat{\mathbf{x}}) = \frac{\mathbf{y}^\top \mathbf{W}\mathbf{b}(\mathbf{x})}{\|\mathbf{W}\mathbf{b}(\mathbf{x})\|} \tag{5.13}$$

$$= \frac{\sum_{j=1}^{L}(\mathbf{y}^\top \mathbf{w}_j)b_j(\mathbf{x})}{\|\mathbf{W}\mathbf{b}(\mathbf{x})\|}. \tag{5.14}$$

The major difference with (5.3) comes from the denominator.

5.4 Experiments

This section evaluates our approach against the popular LSH sketch for cosine estimation [Cha02] and a recent state-of-the-art search technique based on Anti-Sparse coding [JFF12].

5.4.1 Evaluation protocol

The methods are evaluated on both synthetic and real datasets.

Synthetic dataset. We draw i.i.d vectors uniformly on the d-dimensional unit sphere, $d = 8$. For this purpose, we draw the vectors with normal distribution and normalized them to Euclidean unit norm. We produce $N = 1$ million vectors as database (indexed) vectors and $10,000$ queries vectors. The ground-truth is the (exact) cosine similarity.

Real dataset: SIFT1M. We also use a public dataset [JTDA11][1] of SIFT descriptors [Low04]. This dataset, referred to as SIFT1M, consists of $1,000,000$ database and $10,000$ query vectors of dimensionality $d = 128$.

Evaluation metrics. For both datasets, we compare the different methods based on recall@R curves: For each rank R, we measure the proportion of queries for which the true NN (Nearest Neighbor) appears in a position lower or equal to R.

Re-ranking. We adopt a two-stage retrieval procedure for all the methods. The first stage computes the similarities based on the binary codes and produce a short-list of 1,000 vector candidates based on fast Hamming-based computation: we order the vectors based on (2.19). This short-list is subsequently re-ordered with the asymmetric cosine estimation in (5.3), *i.e.* , we use the un-approximated query vector and compare it with short-list vectors reconstructed from their binary codes.

Encoding parameters. All the binarization methods considered in this section produce L-dimensional binary sketches. We set $L = 16$ for the synthetic dataset, in order to get a tractable complexity for the exhaustive optimal quantizer. For SIFT1M, we set $L = 256$. Note the optimal quantizer $\mathbf{c}^\star$ is not tractable for the SIFT dataset, as it is not possible to exhaustively list set of 2^L possible reconstruction values. Similarly, we mention that anti-sparse coding is not tractable with this parameter. The comparison with these two approaches is therefore only performed on the synthetic dataset.

For our method, we set $M = 5$ for the synthetic dataset and $M = 10$ for SIFT1M. The reconstruction quality is always better with higher values of M, however large values of M (*e.g.*, $M = L/2$) suffer the same problem as the optimal quantizer: the sketch is less stable w.r.t. perturbations of the input vector, yielding inferior results w.r.t. binary comparison.

[1] `http://corpus-texmex.irisa.fr`

Table 5.1: Comparison of the properties of different binary sketch constructions on the synthetic dataset.

	MSE	entropy	Query time μs/vector
LSH	0.434	11.39	0.12
LSH+frame	0.207	12.47	0.12
Anti-sparse	0.142	14.23	1,307.40
Optimal	0.075	15.75	324.40
qoLSH	0.107	15.43	3.89

5.4.2 Encoding analysis

Table 5.1 compares several sketch encoding methods based on (1) the quantizer performance measured by mean square error (MSE), (2) the empirical entropy and (3) the encoding time. We use the same tight frame for all the methods except "LSH": for the others, including LSH+Frame, only the encoding strategy differs. The optimal quantizer, by construction, offers the best quantization performance, see (5.6). Our method qoLSH is the best among the tractable encoding strategies. In particular the reconstruction is much better than LSH encoding (with the same frame). The encoding cost of qoLSH is larger than that of LSH, however it remains very efficient: encoding 1,000 vectors takes less than 4 ms. As a reference, computing 1,000×1 million Hamming distances takes 15.5 seconds. For larger datasets, the binary sketch computation associated with the query is negligible compared to Hamming distance computation.

Figure 5.1 plots some statistics (mean, 5% and 95% quantiles) of the difference between the true and estimate cosine similarities, *i.e.* , we show $(\cos(\mathbf{y}, \hat{\mathbf{x}}) - \cos(\mathbf{y}, \mathbf{x}))$ as a function of $\cos(\mathbf{y}, \mathbf{x})$. Compared to LSH, qoLSH decreases the bias and the estimation noise.

5.4.3 Search quality

Figure 5.2 compares the search performance of our algorithm with LSH and anti-sparse on both synthetic and real data with a 2-stage retrieval procedure (short-listing with binary codes and then asymmetric computation). Again, the optimal quantizer achieves the best results on the synthetic dataset, which confirms the importance of improving the quantizer. However, it is slow for $L \geq 20$, typically.

Our approach outperforms the optimal quantizer for large values of R. This is because the sketch comparison based on binary codes is better with our method than with this optimal quantizer, for which two nearby vectors may have very different sketches. This explains the saturation effect of the optimal quantizer observed in the figure. Note

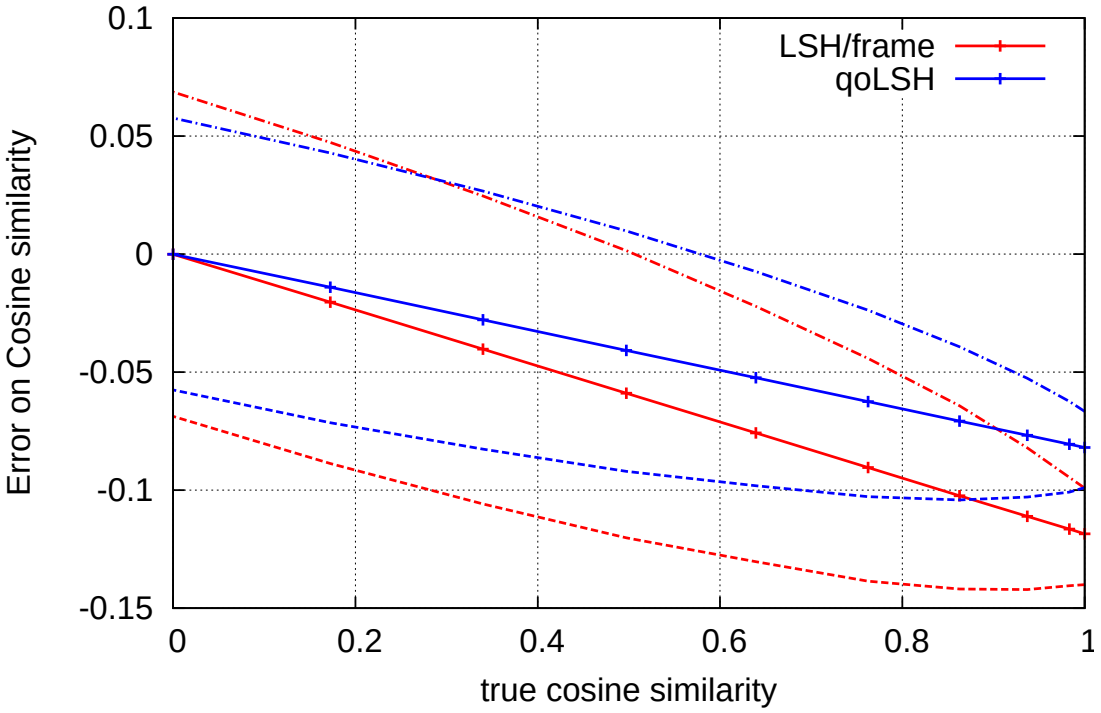

Figure 5.1: Statistics about the difference between estimated and cosine similarities ($d = 128, L = 256$): mean (plain), 5% quantile (dash), 95% quantile (dot-dash).

that all techniques gives different trade-offs from this point of view: anti-sparse coding is also appealing as the binary codes are even more stable than in our approach for large values of R.

For lower R values the performance of our algorithm is much better than LSH and anti-sparse. This assures higher chances of finding nearest neighbors in the top positions. The performance deteriorates after $R = 100$, because of the first binary filter.

Overall, our approach gives a competitive trade-off: For typical values of M, the binary comparison is significantly better than that of regular LSH and slightly better than that of LSH+Frame and anti-sparse coding. After re-ranking with asymmetric distance computation, qoLSH exhibits a large gain over the other tractable methods.

5.5 Conclusion

This chapter discusses the "project and sign" sketch construction method commonly used to estimate the cosine similarity in the compressed domain, and evidences that the method is sub-optimal when seen as a spherical quantizer. This is problematic in a context where the search is refined by considering the explicit reconstruction of a short-list of database vectors.

This leads us to define an alternative encoding strategy that offers significantly better performance both from quantization and approximate search points of view. Compared to other reconstruction focussed encoding schemes, the encoding time is much faster, which makes our solution more useful in practice. We surmise that the improvement we bring to Cosine sketches not just improve the retrieval efficiency of recommender systems, but also other domains which make use of approximate nearest neighbors.

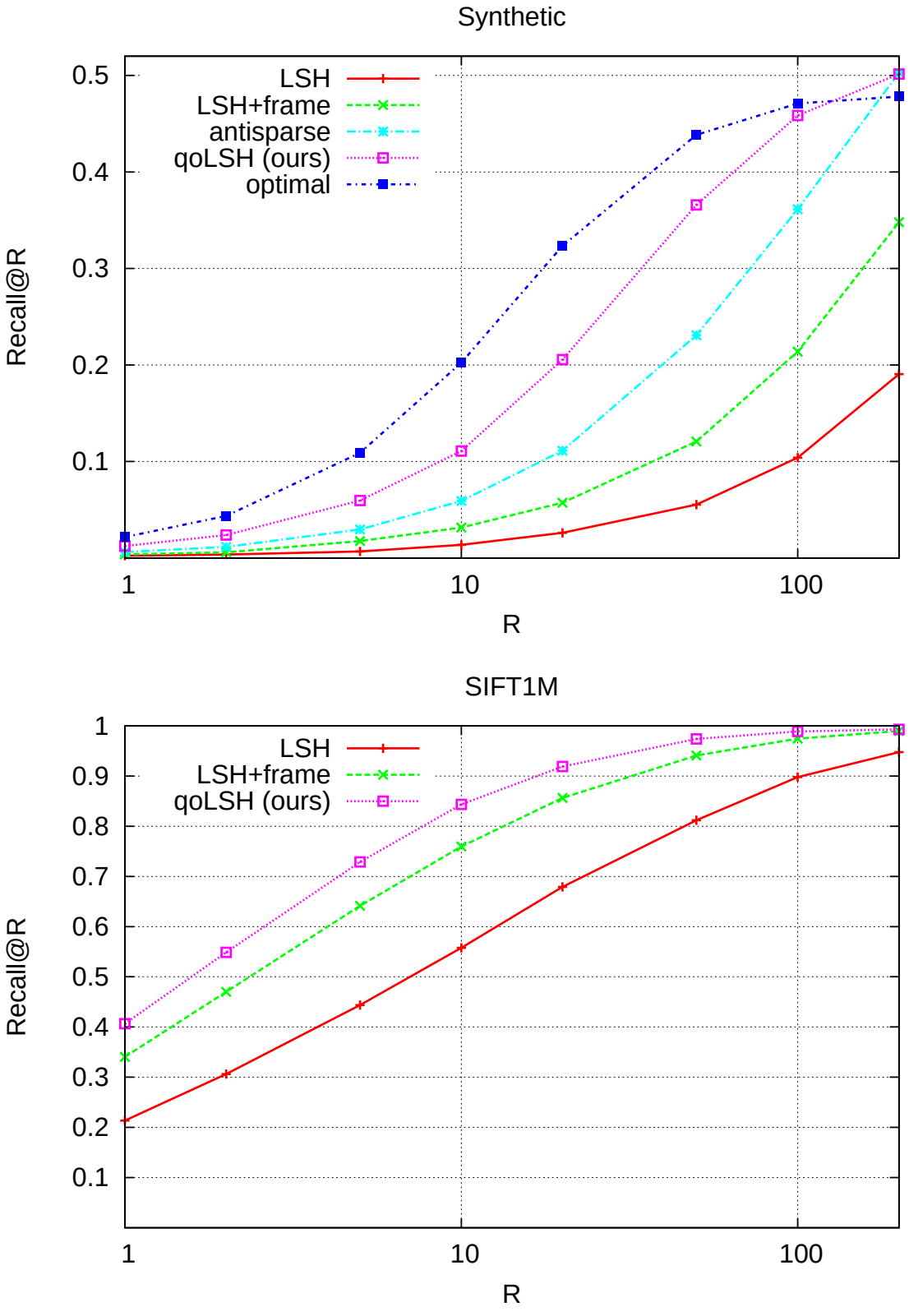

Figure 5.2: Performance comparison: Recall@R on synthetic (top) and SIFT1M (bottom) dataset.

Chapter 6

Conclusion

Our central theme of focus was to study and evaluate the privacy and scalability properties of recommender systems using sketching techniques and propose scalable privacy preserving personalization mechanisms. To that end, we attempted to answer some of prominent issues pertaining to the topic of interest.

On the privacy aspects, we were interested in both new privacy preserving mechanisms and the evaluation of such mechanisms. We observed that ϵ in differential privacy is a control parameter and motivated to find measures that can *assess* the privacy guarantees. We were also interested in proposing new mechanisms that are privacy preserving and get along well with the evaluation metrics.

On the scalability aspects, we were motivated to solve the challenges arising in user modeling and item retrieval. User modeling with evolving data poses new difficulties in storage and adapting to new data, that has to be addressed. Also, addressing the retrieval aspects finds applications in various domains other than recommender systems.

What amplified the motivation is the intersection of these two goals, making the problem more challenging. The thesis being at the intersection of three different domains: recommender systems, differential privacy and sketching techniques; provided ample opportunities for exploring various possibilities and answering many other concerns in combination of these domains. Some of the interesting combinations are: differential privacy and sketching, sketching and modeling, modeling and differential privacy. We conclude on our contributions that stemmed to answer these questions and address the privacy scalability challenges.

6.1 Summary of contributions and observations

In figure 6.1, we graphically present a high level overview of our contributions and how they fit in the big picture of topic of interests. Blocks colored in blue are our contributions and green shaded blocks are existing works and grey shaded blocks are scope

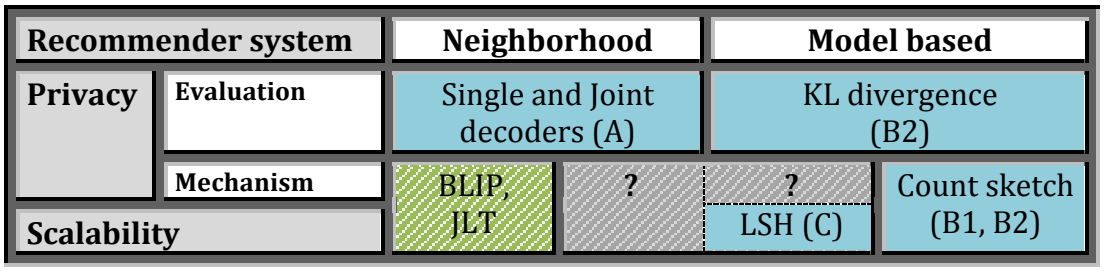

Figure 6.1: Overview of our contributions

for future work. We categorize the contributions into three groups: **contribution A** focuses on neighborhood based recommender systems, **contribution B** deals with scalability and privacy aspects of model based recommender systems and **contribution C** focuses on retrieval aspects. We now list the contribution wise summary, observation and future scope, followed by overall perspectives.

Contribution A

In chapter 3, we assessed the privacy guarantees of differentially private systems. We quantified the amount of information that we can learn from a sanitized database by playing the role of the attacker. We first described a simple attack using single decoder and we improved it by using a more sophisticated joint decoder. We tested our attack models on two sketch based differentially private recommender system techniques: BLIP and Johnson-Lindenstrauss transform. We studied the problem theoretically and also designed practical decoders to work on the before mentioned schemes. Through our theoretical analysis and experimental validation on public datasets, we proved the superiority of joint decoders over single decoder in performing inference attacks on sanitized data.

The superiority of joint decoder over single decoder is huge for large ϵ (low privacy), tapering down with decreasing ϵ (increasing privacy). The observation is a consistent pattern on all datasets supporting the guarantees of differential privacy. Nevertheless, we should not rule out the possibility of existence of more sophisticated inference attacks, which can derive more information. Our MCMC based practical implementation of joint decoder is a compromise between usability and accuracy. We acknowledge the possibility of using other approximate inference techniques such as Belief propagation and variational inference.

We used existing techniques and only contributed on the evaluation aspects of differentially private similarity estimation mechanisms. We observed that the two existing techniques: BLIP and JL Transform use Bloom filter and random projection respectively, which are not the best known space efficient similarity estimation techniques in

Technique	Representation size	Prediction error
Count sketch factorization	Fixed	Variable
Regular factorization	Variable	Fixed

Table 6.1: Count sketch vs regular factorization

the literature. Hence, we suggest devising new mechanisms using compact and efficient similarity estimation techniques such as LSH. LSH has proved its efficacy in similarity approximation and the trend might follow in differentially private estimation too.

Contribution B1 & B2

In chapter 4, we described our matrix factorization approach using count sketch. We also demonstrated the scalability and dynamicity properties of sketch based factorization through experimental results on benchmark public rating datasets. We also described a modification to our system to learn the models in a differentially private setup. We then described a privacy evaluation metric using Kullback-Leibler divergence that is more relevant to our setup and evaluated it on standard datasets.

Our first observation is that our count sketch based factorization has contrasting trade-off properties when compared against regular matrix factorization. With increasing data, the prediction accuracy of our approach goes down, without impacting the representation size. In case of regular factorization, the properties are quite opposite: the representation size increases with data, without much impacting the prediction accuracy. We tabulate and highlight the difference in table 6.1.

Our second observation is that the regularization and privacy capabilities are inherent, thanks to the hash-based randomization, which contrasts to the conventional methods. In conventional approaches, regularization is incorporated by limiting the parameter search and privacy by explicit perturbation of data. A future scope would be to find and explore other similar sketching techniques to represent parameters, so that regularization and privacy capabilities are implicitly provided at storage level.

Contribution C

In chapter 5, we improved the retrieval efficiency of latent factor based recommender systems using LSH. We described an instance of LSH called cosine sketches which can approximate normalized inner product estimation. We motivated the problem from a reconstruction point of view and briefly described the existing anti-sparse encoder technique. We also proposed a new technique, from a reconstruction point of view, to improve the estimation efficiency of such cosine sketches. We finally validated our technique on synthetic and benchmark real-vector datasets.

We observe that our sequential improvement of reconstruction error is suboptimal from a reconstruction point of view. Hence there is a room for devising sophisticated quantization based encoders that optimizes multiple bits together. The catch is that such an encoder should obey time and space complexities like our technique.

Our contribution is primarily on the later stage of encoding and we make simple assumptions about the projection matrix: it is over-complete and a tight-frame. Not optimizing to the input data distribution makes our approach simple, generic and robust. Nevertheless, optimizing the projection matrix to the input data distribution might reduce the reconstruction error arising from quantization and thereby improve the retrieval accuracy. Hence there is a scope to explore the possibilities using machine learning techniques.

We observe that quantization step in the encoding introduces error on reconstruction and anticipate that the introduced error might enable privacy as a side effect. But the privacy properties of LSH based quantization is hard to evaluate, leaving room for finding efficient and retrieval friendly privacy preserving mechanisms using LSH (shaded grey block in figure 6.1).

Publications

The thesis work resulted in the following list of publications, presented in various international conferences.

- **Contribution A**: *Challenging differential privacy: the case of non-interactive mechanisms*, Raghavendran Balu, Teddy Furon, Sébastien Gambs. ESORICS European Symposium on Research in Computer Security, Sep 2014, Wroclaw, Poland. **Best student paper award**.
- **Contribution B1**: *Sketching techniques for very large matrix factorization*, Raghavendran Balu, Teddy Furon, Laurent Amsaleg. ECIR 38th European Conference on Information Retrieval, Mar 2016, Padoue, Italy.
- **Contribution B2**: *Differentially Private Matrix Factorization using Sketching Techniques*, Raghavendran Balu, Teddy Furon. IH&MMSec ACM workshop on Infomation Hiding and MultiMedia SECurity, Jun 2016, Vigo, Spain.
- **Contribution C**: *Beyond "project and sign" for cosine estimation with binary codes*, Raghavendran Balu, Teddy Furon, Hervé Jégou. IEEE ICASSP International Conference on Acoustics, Speech, and Signal Processing, May 2014, Florence, Italy.

6.2 Perspectives

We present some new directions on the topics related to our thesis work, that are worth exploring by the research community in the future.

Retrieval friendly recommender systems

Our contribution C focused on improving retrieval of recommender systems and other similar problems. The number of items to be recommended is enormous these days in applications like music, video, and keeps increasing. This increasing pattern hints the necessity to concentrate on the retrieval challenges. A promising direction is to model the user and items using retrieval friendly representations such as compact binary codes and optimize. But, the discrete nature of such representations makes optimization cumbersome. Nevertheless, there are some promising existing works such as [ZZ12, WWY15, LHDL14]. We suggest exploring more in this perspective of retrieval focused recommender systems.

Sketching techniques for machine learning

A brief summary of the interaction between sketching and machine learning is presented in section 2.3.4. We observe that the symbiotic relationship between these two techniques are significant to explore more in the future. Sketching is a subtype of randomization techniques and it is known that machine learning complements well with these techniques.

In problems like dimensionality reduction, randomization techniques such as random projection are seen as a simpler alternative to linear and non-linear machine learning techniques such as principal component analysis, linear discriminant analysis, manifold learning, etc. In similarity approximation, LSH is a simpler randomization based alternative to sophisticated learning based techniques such as learning binary codes [SH09, WTF09b] and quantization codes [JDS11b]. On other cases, randomization is incorporated well into the learning algorithm itself such as stochastic gradient descent, random sampling, drop out [SHK+14] and input perturbation [Bis95], to name a few.

All these suggest that sketching techniques have deeper connections with machine learning, which is unexplored as of now. We positioned our contribution on the scalability and privacy aspects. A systematic exploration on other aspects might be beneficial to both sketching and machine learning domains.

Privacy: Control or Estimate?

We presented two classes of attack models: Single and Joint decoders and KL divergence based inference. Our Single and joint decoders are theoretically bounded by the mutual

information between sanitized profile and "items in suspect", as described in section 3.3. On the other hand, KL divergence is nothing but a measure of information gain in the error distributions of observed (training) and unobserved (testing) data. We perceive the similarity among the two techniques and concur that information theoretic approach is a strong contender as a privacy evaluation measure.

Observing that differential privacy is nothing but maximal information gain among neighboring datasets supports our argument. The difference being, ϵ is a control measure, where as our contributions are estimated measures. We suggest to explore in this direction in the future, for new potential privacy measures. An evaluation measure is practical, easy to implement, and complements well with the control parameter ϵ, which gives only theoretical security assurance.

Differentially private machine learning

We described the interaction of differential privacy and machine learning in section 2.4.8. We observe that privacy is often seen as a constraint that has to be incorporated with other objectives. Recent advances in efficient differentially private mechanisms such as Bayesian sampling [WFS15] paint a different and promising picture. Bayesian methods are widely popular in the machine learning community and the equivalence of privacy mechanisms and bayesian posterior estimation means that privacy is not alien to the community and encourages incorporating privacy into the learning objectives. It is imperative to see that the objective of both privacy and machine learning is one and same: *"generalize well and don't overfit to/remember individual instances"*. Hence we suggest to devise more privacy enabling mechanisms in the future that benefits both domains.

Privacy, Regularization and Randomized storage

Our previous perspective implied the commonality of privacy and generalization requirements. The usual approach to achieve both is through explicit mechanisms. We demonstrated through contribution B that providing these two properties can be achieved implicitly by means of a randomized storage structure. We suggest exploring along the lines, in finding new randomized storage structures that guarantee privacy and regularization implicitly, by taking advantage of randomized nature of such structures.

On a surface level, these topics may look like remote, as it arose from different communities. However, the deeper connections among these topics cannot be ruled out, owing to the play of randomness factor on these disparate objectives. It is also worthwhile to see if there is a latent "topic of interest" that is still unexplored and waiting to be discovered.

Bibliography

[AA01] Dakshi Agrawal and Charu C. Aggarwal. On the design and quantification of privacy preserving data mining algorithms. In *20th ACM SIGMOD-SIGACT-SIGART Symp. on Principles of database systems*, pages 247–255, 2001.

[AACP11] Mário S. Alvim, Miguel E. Andrés, Konstantinos Chatzikokolakis, and Catuscia Palamidessi. On the relation between differential privacy and quantitative information flow. In *Int. Conf. on Automata, Languages and Programming*, pages 60–76, 2011.

[ABK+13] Armen Aghasaryan, Makram Bouzid, Dimitre Kostadinov, Mohit Kothari, and Animesh Nandi. On the use of lsh for privacy preserving personalization. In *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pages 362–371. IEEE, 2013.

[ABPH07] Paulo Sérgio Almeida, Carlos Baquero, Nuno Preguia, and David Hutchison. Scalable bloom filters. *Information Processing Letters*, 101(6):255–261, 2007.

[AC06] Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform. *STOC '06 - Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 557–563, 2006.

[Ach01] Dimitris Achlioptas. Database-friendly random projections. In *Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 274–281. ACM, 2001.

[AGK12] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. BLIP: Non-interactive Differentially-Private Similarity Computation on Bloom Filters. In *14th Int. Symp. on Stabilization, Safety, and Security of Distributed Systems*, Toronto, Canada, 2012.

[AGK15] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. Heterogeneous differential privacy. *arXiv preprint arXiv:1504.06998*, 2015.

[AMS96] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. In *STOC*, pages 20–29. ACM, 1996.

[AS00] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *ACM Sigmod Record*, volume 29, pages 439–450. ACM, 2000.

[BCFM98] Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. Min-wise independent permutations. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 327–336, New York, NY, USA, 1998. ACM.

[BFK+15] Arnaud Berlioz, Arik Friedman, Mohamed Ali Kaafar, Roksana Boreli, and Shlomo Berkovsky. Applying differential privacy to matrix factorization. In *RecSys*, RecSys '15, pages 107–114, New York, NY, USA, 2015. ACM.

[BHK98] John S Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52. Morgan Kaufmann Publishers Inc., 1998.

[Bis95] Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.

[BL07] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

[Blo70] Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[BMP+06] Flavio Bonomi, Michael Mitzenmacher, Rina Panigrahy, Sushil Singh, and George Varghese. An improved construction for counting bloom filters. In *Algorithms–ESA 2006*, pages 684–695. Springer, 2006.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[BNO08] Amos Beimel, Kobbi Nissim, and Eran Omri. Distributed private data analysis: on simultaneously solving *how* and *what*. In *Proc. of Advances in Cryptology*, pages 451–468, 2008.

[Bou12] P. T. Boufounos. Universal rate-efficient scalar quantization. *IEEE Trans. Inform. Theory*, 58(3), March 2012.

[BP98] Daniel Billsus and Michael J Pazzani. Learning collaborative information filters. In *Icml*, volume 98, pages 46–54, 1998.

[BPR09] Yoram Bachrach, Ely Porat, and Jeffrey S Rosenschein. Sketching techniques for collaborative filtering. In *IJCAI*, pages 2016–2021, 2009.

[Bro97] Andrei Z Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences 1997. Proceedings*, pages 21–29. IEEE, 1997.

[CCFC02] Moses Charikar, Kevin Chen, and Martin Farach-Colton. Finding frequent items in data streams. In *ICALP*. 2002.

[Cha02] M. S. Charikar. Similarity estimation techniques from rounding algorithms. In *STOC*, pages 380–388, May 2002.

[CKN+11] J.A. Calandrino, A. Kilzer, A. Narayanan, E.W. Felten, and V. Shmatikov. "you might also like:" privacy risks of collaborative filtering. In *SP*, pages 231–246, May 2011.

[CKRT04] Bernard Chazelle, Joe Kilian, Ronitt Rubinfeld, and Ayellet Tal. The bloomier filter: an efficient data structure for static support lookup tables. In *Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 30–39. Society for Industrial and Applied Mathematics, 2004.

[CL08] Keke Chen and Ling Liu. A survey of multiplicative perturbation for privacy-preserving data mining. In *Privacy-Preserving Data Mining*, pages 157–181. Springer, 2008.

[CMS11] Kamalika Chaudhuri, Claire Monteleoni, and Anand D. Sarwate. Differentially private empirical risk minimization. *J. Mach. Learn. Res.*, 12:1069–1109, July 2011.

[Cor11] Graham Cormode. Sketch techniques for approximate query processing. *FnTD. NOW publishers*, 2011.

[DCL08] W. Dong, M. Charikar, and K. Li. Asymmetric distance estimation with sketches for similarity search in high-dimensional spaces. In *SIGIR*, pages 123–130, July 2008.

[DDGR07] Abhinandan S Das, Mayur Datar, Ashutosh Garg, and Shyam Rajaram. Google news personalization: scalable online collaborative filtering. In *WWW*, pages 271–280. ACM, 2007.

[DIIM04] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, SCG '04, pages 253–262, New York, NY, USA, 2004. ACM.

[DK04] Mukund Deshpande and George Karypis. Item-based top-n recommendation algorithms. *ACM Transactions on Information Systems (TOIS)*, 22(1):143–177, 2004.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[DKS10] Anirban Dasgupta, Ravi Kumar, and Tamás Sarlós. A sparse johnson: Lindenstrauss transform. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 341–350. ACM, 2010.

[Dob00] Adrian Dobra. Measuring the disclosure risk for multi-way tables with fixed marginals corresponding to decomposable log-linear models. Technical report, 2000.

[DR06] Fan Deng and Davood Rafiei. Approximately detecting duplicates for streaming data using stable bloom filters. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 25–36. ACM, 2006.

[DR16] Cynthia Dwork and Guy N Rothblum. Concentrated differential priacy. *arXiv preprint arXiv:1603.01887*, 2016.

[DS98] Persi Diaconis and Bernd Sturmfels. Algebraic algorithms for sampling from conditional distributions. *The Annals of Statistics*, 26(1):363–397, Feb. 1998.

[Dwo06a] Cynthia Dwork. Differential privacy. In *Int. Conf. on Automata, Languages and Programming*, volume 4052 of *LNCS*, pages 1–12. Springer, 2006.

[Dwo06b] Cynthia Dwork. *ICALP 2006*, chapter Differential Privacy, pages 1–12. Berlin, Heidelberg, 2006.

[Dwo08] Cynthia Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008.

[FCAB00] Li Fan, Pei Cao, Jussara Almeida, and Andrei Z Broder. Summary cache: a scalable wide-area web cache sharing protocol. *IEEE/ACM Transactions on Networking (TON)*, 8(3):281–293, 2000.

[FGC12] T. Furon, A. Guyader, and F. Cerou. Decoding fingerprints using the Markov Chain Monte Carlo method. In *IEEE Int. Work. on Information Forensics and Security (WIFS)*, pages 187–192, 2012.

[FM87] P. Frankl and H. Maehara. The johnson-lindenstrauss lemma and the sphericity of some graphs. *J. Comb. Theory Ser. A*, 44(3):355–362, June 1987.

[FS10] Arik Friedman and Assaf Schuster. Data mining with differential privacy. In *SIGKDD*, KDD '10, pages 493–502, New York, NY, USA, 2010. ACM.

[Fuc11] J. J. Fuchs. Spread representations. In *ASILOMAR*, November 2011.

[GL11] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *CVPR*, June 2011.

[GRGP01] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A constant time collaborative filtering algorithm. *Information Retrieval*, 4(2):133–151, 2001.

[GVT98] V. K. Goyal, M. Vetterli, and N. T. Thao. Quantized overcomplete expansions in $\mathcal{R}^N$: Analysis, synthesis, and algorithms. *IEEE Trans. Inform. Theory*, 44(1):16–31, January 1998.

[GW06] Songtao Guo and Xintao Wu. On the use of spectral filtering for privacy preserving data mining. In *ACM Symp. on Applied Computing*, pages 622–626, 2006.

[GW07] Songtao Guo and Xintao Wu. Deriving private information from arbitrarily projected data. In *Advances in Knowledge Discovery and Data Mining*, volume 4426 of *LNCS*, pages 84–95. Springer, 2007.

[HDC05] Zhengli Huang, Wenliang Du, and Biao Chen. Deriving private information from randomized data. In *ACM SIGMOD Int. Conf. on Management of data*, pages 37–48. ACM, 2005.

[HH05] Thomas Hofmann and D Hartmann. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 2005 ACM symposium on applied computing*, pages 791–795, 2005.

[Hof04] Thomas Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems (TOIS)*, 22(1):89–115, 2004.

[HSRF95] Will Hill, Larry Stead, Mark Rosenstein, and George Furnas. Recommending and evaluating choices in a virtual community of use. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 194–201. ACM Press/Addison-Wesley Publishing Co., 1995.

[IM98] Piotr Indyk and Rajeev Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *STOC*, pages 604–613. ACM, 1998.

[JB11] A. Joly and O. Buisson. Random maximum margin hashing. In *CVPR*, 2011.

[JDS08] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, October 2008.

[JDS11a] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Trans. PAMI*, 33(1):117–128, January 2011.

[JDS11b] Herve Jegou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE transactions on pattern analysis and machine intelligence*, 33(1):117–128, 2011.

[JDSP10] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *CVPR*, June 2010.

[JFF12] H. Jégou, T. Furon, and J. J. Fuchs. Anti-sparse coding for approximate nearest neighbor search. In *ICASSP*, March 2012.

[JJG11] M. Jain, H. Jégou, and P. Gros. Asymmetric hamming embedding. In *ACM Multimedia*, October 2011.

[JKT11] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. 09 2011.

[JL84] William B Johnson and Joram Lindenstrauss. Extensions of lipschitz mappings into a hilbert space. *Contemporary mathematics*, 26(189-206):1, 1984.

[JLE14] Zhanglong Ji, Zachary Chase Lipton, and Charles Elkan. Differential privacy and machine learning: a survey and review. *CoRR*, abs/1412.7584, 2014.

[JLY+12] J. Ji, J. Li, S. Yan, B. Zhang, and Q. Tian. Super-bit locality-sensitive hashing. In *NIPS*, December 2012.

[JTDA11] H. Jégou, R. Tavenard, M. Douze, and L. Amsaleg. Searching in one billion vectors: re-rank with source coding. In *ICASSP*, Prague Czech Republic, 2011.

[KBV09] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

[KD09a] B. Kulis and T. Darrell. Learning to hash with binary reconstructive embeddings. In *NIPS*, December 2009.

[KD09b] Brian Kulis and Trevor Darrell. Learning to hash with binary reconstructive embeddings. In *Advances in neural information processing systems*, pages 1042–1050, 2009.

[KKMM12] Krishnaram Kenthapadi, Aleksandra Korolova, Ilya Mironov, and Nina Mishra. Privacy via the johnson-lindenstrauss transform. *arXiv preprint arXiv:1204.2606*, 2012.

[KLN+11] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[KN10] Daniel M Kane and Jelani Nelson. A derandomized sparse johnson-lindenstrauss transform. *arXiv preprint arXiv:1006.3585*, 2010.

[KN14] Daniel M Kane and Jelani Nelson. Sparser johnson-lindenstrauss transforms. *Journal of the ACM (JACM)*, 61(1):4, 2014.

[KST96] E Knill, A Schliep, and D C Torney. Interpretation of pooling experiments using the Markov chain Monte Carlo method. *J. Comput. Biol.*, 3(3):395–406, 1996.

[KT13] Michael Kapralov and Kunal Talwar. On differentially private low rank approximation. In *ACM-SIAM*, pages 1395–1414. SIAM, 2013.

[KWS10] Alexandros Karatzoglou, Markus Weimer, and Alex J Smola. Collaborative filtering on a budget. In *AISTATS*, 2010.

[LAS08] Edo Liberty, Nir Ailon, and Amit Singer. Dense fast random projections and lean walsh transforms. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 512–522. Springer, 2008.

[LC11] Jaewoo Lee and Chris Clifton. How much is enough? Choosing ϵ for differential privacy. In *Information Security*, volume 7001 of *LNCS*, pages 325–340. Springer, 2011.

[LCH06] Ping Li, Kenneth W Church, and Trevor J Hastie. Conditional random sampling: A sketch-based sampling technique for sparse data. In *Advances in neural information processing systems*, pages 873–880, 2006.

[LGK08] Kun Liu, Chris Giannella, and Hillol Kargupta. A survey of attack techniques on privacy-preserving data perturbation methods. In *Privacy-Preserving Data Mining*, volume 34 of *Advances in Database Systems*, pages 359–381. Springer, 2008.

[LHDL14] Xianglong Liu, Junfeng He, Cheng Deng, and Bo Lang. Collaborative hashing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2139–2146, 2014.

[Lib13] Edo Liberty. Simple and deterministic matrix sketching. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 581–588. ACM, 2013.

[LK11] Ping Li and Arnd Christian König. Theory and applications of b-bit minwise hashing. *Communications of the ACM*, 54(8):101–109, 2011.

[Low04] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[LSMK11] Ping Li, Anshumali Shrivastava, Joshua L Moore, and Arnd C König. Hashing algorithms for large-scale learning. In *NIPS*, pages 2672–2680, 2011.

[LSY03] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *IEEE Internet computing*, 7(1):76–80, 2003.

[LWS15] Ziqi Liu, Yu-Xiang Wang, and Alexander Smola. Fast differentially private matrix factorization. In *RecSys*, RecSys '15, pages 171–178, New York, NY, USA, 2015. ACM.

[LZWY11] Yang D. Li, Zhenjie Zhang, Marianne Winslett, and Yin Yang. Compressive mechanism: utilizing sparse representation in differential privacy. *CoRR*, abs/1107.3350, 2011.

[Mat08] Jirí Matousek. On variants of the johnson–lindenstrauss lemma. *Random Structures & Algorithms*, 33(2):142–156, 2008.

[MDDC15] Luca Melis, George Danezis, and Emiliano De Cristofaro. Efficient private statistics with succinct sketches. *arXiv preprint arXiv:1508.06110*, 2015.

[MF11] P. Meerwald and T. Furon. Group testing meets traitor tracing. In *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, pages 4204–4207, 2011.

[MF12] P. Meerwald and T. Furon. Toward practical joint decoding of binary Tardos fingerprinting codes. *IEEE Trans. on Inf. Forensics and Security,*, 7(4):1168–1180, 2012.

[MM09] Frank McSherry and Ilya Mironov. Differentially private recommender systems: Building privacy into the net. In *SIGKDD*, KDD '09, pages 627–636, New York, NY, USA, 2009. ACM.

[Mou08] Pierre Moulin. Universal fingerprinting: capacity and random-coding exponents. arXiv:0801.3837, January 2008.

[mov] http://grouplens.org/datasets/movielens/.

[MS03] Jirí Matousek and Milos Stojaković. On restricted min-wise independence of permutations. *Random Structures & Algorithms*, 23(4):397–408, 2003.

[MS06] Nina Mishra and Mark Sandler. Privacy via pseudorandom sketches. In *ACM SIGMOD-SIGACT-SIGART*, pages 143–152. ACM, 2006.

[MS07] Andriy Mnih and Ruslan Salakhutdinov. Probabilistic matrix factorization. In *Advances in neural information processing systems*, pages 1257–1264, 2007.

[MT07] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *FOCS*, pages 94–103, Oct 2007.

[Mut05] Shanmugavelayutham Muthukrishnan. *Data streams: Algorithms and applications*. Now Publishers Inc, 2005.

[NAB11] Animesh Nandi, Armen Aghasaryan, and Makram Bouzid. P3: A privacy preserving personalization middleware for recommendation-based services. In *Hot Topics in Privacy Enhancing Technologies Symposium*, 2011.

[NB11] Mohammad Norouzi and David M Blei. Minimal loss hashing for compact binary codes. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 353–360, 2011.

[NRS07] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *STOC*, STOC '07, pages 75–84, New York, NY, USA, 2007. ACM.

[NS08] Arvind Narayanan and Vitaly Shmatikov. Robust de-anonymization of large sparse datasets. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*, pages 111–125. IEEE, 2008.

[PJA10] Loïc Paulevé, Hervé Jégou, and Laurent Amsaleg. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters*, 31(11):1348–1358, 2010.

[PLSP10] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier. Large-scale image retrieval with compressed Fisher vectors. In *CVPR*, June 2010.

[PP13] Ninh Pham and Rasmus Pagh. Fast and scalable polynomial kernels via explicit feature maps. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 239–247. ACM, 2013.

[PSS07] Felix Putze, Peter Sanders, and Johannes Singler. Cache-, hash-and space-efficient bloom filters. In *Experimental Algorithms*, pages 108–121. Springer, 2007.

[RC04] C.P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer Verlag, 2004.

[RFGST09] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pages 452–461. AUAI Press, 2009.

[RIS+94] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186. ACM, 1994.

[RKK14] Ori Rottenstreich, Yossi Kanizo, and Isaac Keslassy. The variable-increment counting bloom filter. *IEEE/ACM Transactions on Networking (TON)*, 22(4):1092–1105, 2014.

[RL10] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *NIPS*, 2010.

[RR07a] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *NIPS*, 2007.

[RR07b] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems*, pages 1177–1184, 2007.

[RS05] Jasson DM Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning*, pages 713–719. ACM, 2005.

[SC13] A.D. Sarwate and K. Chaudhuri. Signal processing and machine learning with differential privacy: Algorithms and challenges for continuous data. *Signal Processing Magazine, IEEE*, 30(5):86–94, Sept 2013.

[SH09] Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009.

[SHK+14] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[SJ+03] Nathan Srebro, Tommi Jaakkola, et al. Weighted low-rank approximations. In *ICML*, volume 3, pages 720–727, 2003.

[SJ10] D. Sejdinovic and O. Johnson. Note on noisy group testing: asymptotic bounds and belief propagation reconstruction. In *Proc. 48th Allerton Conf. on Commun., Control and Computing*, Monticello, IL, USA, October 2010. arXiv:1010.2441v1.

[SKKR00] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system-a case study. Technical report, DTIC Document, 2000.

[SKKR01] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295. ACM, 2001.

[SLH12] Malcolm Slaney, Yury Lifshits, and Junfeng He. Optimal parameters for locality-sensitive hashing. *Proceedings of the IEEE*, 100(9):2604–2623, 2012.

[SM95] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating âĂIJword of mouthâĂİ. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217. ACM Press/Addison-Wesley Publishing Co., 1995.

[SM08a] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. *Proceedings of the 25th international conference on Machine learning - ICML '08*, pages 880–887, 2008.

[SM08b] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems*, volume 20, 2008.

[SMH07] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th international conference on Machine learning*, pages 791–798. ACM, 2007.

[SRJ04] Nathan Srebro, Jason Rennie, and Tommi S Jaakkola. Maximum-margin matrix factorization. In *Advances in neural information processing systems*, pages 1329–1336, 2004.

[SVZ13] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning local feature descriptors using convex optimisation. Technical report, Department of Engineering Science, University of Oxford, 2013.

[TFW08a] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large databases for recognition. In *CVPR*, June 2008.

[TFW08b] Antonio Torralba, Rob Fergus, and Yair Weiss. Small codes and large image databases for recognition. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[UF98] Lyle H Ungar and Dean P Foster. Clustering methods for collaborative filtering. In *AAAI workshop on recommendation systems*, volume 1, pages 114–129, 1998.

[VZT15] S. J. Vollmer, K. C. Zygalakis, and Y.W. Teh. (non-) asymptotic properties of stochastic gradient langevin dynamics. *arXiv:1501.00438*, 2015.

[War65] Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

[WDL+09] Kilian Weinberger, Anirban Dasgupta, John Langford, Alex Smola, and Josh Attenberg. Feature hashing for large scale multitask learning. In *ICML*. ACM, 2009.

[WFS15] Yu-Xiang Wang, Stephen E Fienberg, and Alex Smola. Privacy for free: Posterior sampling and stochastic gradient monte carlo. *arXiv preprint arXiv:1502.07645*, 2015.

[WKLS07] Markus Weimer, Alexandros Karatzoglou, Quoc Viet Le, and Alex Smola. Maximum margin matrix factorization for collaborative ranking. *Advances in neural information processing systems*, pages 1–8, 2007.

[WKS08] Markus Weimer, Alexandros Karatzoglou, and Alex Smola. Improving maximum margin matrix factorization. *Machine Learning*, 72(3):263–276, 2008.

[WM10] Oliver Williams and Frank McSherry. Probabilistic inference and differential privacy. In *Advances in Neural Information Processing Systems*, pages 2451–2459, 2010.

[WT11] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *ICML*, pages 681–688, Bellevue, WA, USA, 2011.

[WTF09a] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *NIPS*, December 2009.

[WTF09b] Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in neural information processing systems*, pages 1753–1760, 2009.

[WWY15] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1235–1244. ACM, 2015.

[ZZ12] Ke Zhou and Hongyuan Zha. Learning binary codes for collaborative filtering. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 498–506. ACM, 2012.

List of Tables

List of Figures

Résumé

Cette thèse étudie les aspects passage à l'échelle et respect de la vie privée des systèmes de recommandation grâce à l'emploi d'algorithmes à base de *sketchs*. Les contributions techniques liées à cette étude nous permettent de proposer un système de recommandations personnalisées capable de passer à l'échelle tant en nombre d'utilisateurs qu'en nombre de produits à recommander, tout en offrant une bonne protection de la confidentialité de ces recommandations. La thèse se situe ainsi à la croisée de trois domaines qui sont les systèmes de recommandation, la confidentialité différentielle et les techniques à base de *sketchs*. Concernant la confidentialité, nous nous sommes intéressés à définir de nouveaux mécanisme garantissant une bonne confidentialité mais aussi à les évaluer. Nous avons pu observer que c'est ϵ qui est le paramètre essentiel contrôlant le respect plus ou moins garanti de la confidentialité différentielle. Par ailleurs, le besoin de fonctionner à grande échelle demande de relever les défis liés à la modélisation de très nombreux utilisateurs et à la prise en compte de très nombreux produits à recommander. Ces défis sont particulièrement difficiles à relever dans un contexte où les préférences des utilisateurs et le catalogue de produits évoluent dynamiquement. Cette évolution complexifie les techniques de stockage des profils des utilisateurs, leur mise à jour et leur interrogation. Nos contributions sur cet aspect intéressent non seulement le domaine de la recommandation, mais ont une portée plus générale. Globalement, nous avons mené de nombreuses campagnes d'évaluation de nos propositions, sur des jeux de données réels de très grande taille, démontrant ainsi la capacité de nos contributions à passer à l'échelle tout en offrant de la dynamicité et des garanties sur la confidentialité.

Abstract

In this thesis, we aim to study and evaluate the privacy and scalability properties of recommender systems using sketching techniques and propose scalable privacy preserving personalization mechanisms. Hence, the thesis is at the intersection of three different topics: recommender systems, differential privacy and sketching techniques. On the privacy aspects, we are interested in both new privacy preserving mechanisms and the evaluation of such mechanisms. We observe that the primary parameter ϵ in differential privacy is a control parameter and motivated to find techniques that can assess the privacy guarantees. We are also interested in proposing new mechanisms that are privacy preserving and get along well with the evaluation metrics. On the scalability aspects, we aim to solve the challenges arising in user modeling and item retrieval. User modeling with evolving data poses difficulties, to be addressed, in storage and adapting to new data. Also, addressing the retrieval aspects finds applications in various domains other than recommender systems. We evaluate the impact of our contributions through extensive experiments conducted on benchmark real datasets and through the results, we surmise that our contributions very well address the privacy and scalability challenges.