UNIVERSITY OF TORONTO

HONOURS THESIS

An Exploration of Machine Learning Algorithms for Photo Recommendation

Author:
Yu Qing ZHOU

Supervisor:
Dr. Scott SANNER

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Applied Science

in the

Data Driven Decision Making Lab
Department of Mechanical and Industrial Engineering

April 11, 2018

Declaration of Authorship

I, Yu Qing ZHOU, declare that this thesis titled, "An Exploration of Machine Learning Algorithms for Photo Recommendation" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Ivan Zhou

Date: April 11th, 2018

UNIVERSITY OF TORONTO

Abstract

Faculty of Applied Science and Engineering
Department of Mechanical and Industrial Engineering

Bachelor of Applied Science

An Exploration of Machine Learning Algorithms for Photo Recommendation

by Yu Qing ZHOU

This thesis uses the dataset from the photography community 500px to develop photo recommender systems. We believe the human perception of photos should be a critical component in their reaction towards photos. Therefore, we propose a novel algorithm to quantify photos' visual features and embed them into the recommender system. Through experiment, the new algorithm shows a performance increase of 24% from the item-based nearest neighbourhood algorithm and 2.84 times better than our baseline algorithm in precision@10. We also evaluate the existing one-class recommender system algorithms in the same dataset, including but not limited to: SVD matrix factorization, LRec, and Linear FLow. We assess the benefit of several techniques to improve the performance of these existing algorithms. We find that the User-based nearest neighbourhood (U-NN) algorithm performs better than other existing algorithms in this dataset. Beyond that, through the technique of embedding time decay effect into the models, we successfully improve the performance of U-NN even further. It shows the users' taste does shift across time, and thus should be captured by the recommender system models.

Acknowledgements

I'd like to thank my thesis supervisor Professor Scott Sanner for his great guidance and dedication to help me through my research. Great appreciation to Zimu Liu and other members in 500px team for their support through this research partnership. Thank you to Wuga, Vincent Chio, Sam Talasila, Putra Manggala, and Chen Karako for their kind suggestions and mental support for this thesis. Thank you to Shopify team to provide me a very enjoyable and resourceful internship during the time I am writing this thesis. I'd like to thank Annie Luo for her valuable support and understanding for my work. Finally, I'd like to thank my parents for their trust and care on me throughout all the years in my undergraduate.

Contents

Declaration of Authorship iii

Abstract v

Acknowledgements vii

1 Introduction 1
- 1.1 Background: Personalized Recommendation . . . 1
- 1.2 Motivation . . . 2
- 1.3 Contribution . . . 2

2 Background 5
- 2.1 Formal Definition of the Photo Recommendation . . . 5
- 2.2 Approaches of Photo Recommendation . . . 5
  - 2.2.1 General Approaches in Traditional Recommender Systems . . . 5
  - 2.2.2 Literature Review: Relevant Work in Photo Recommendation . 6
- 2.3 Algorithms in Recommender Systems . . . 7
  - 2.3.1 Nearest Neighbourhood models (NN) . . . 7
    - Similarity Metrics . . . 8
    - Use of Metadata in Cold-Start CF . . . 9
  - 2.3.2 Temporal Decay Models . . . 9
    - Defining Time Difference . . . 10
  - 2.3.3 Matrix Factorization Models (MF) . . . 11
    - Embedding of the Meta Data . . . 11
    - Choice of the Weighting Matrix . . . 11
  - 2.3.4 Linear Models . . . 12
    - The LRec Model . . . 12
    - The Linear-FLow Model . . . 13
- 2.4 Summary . . . 13

3 Novel Photo Recommendation Algorithms 15
- 3.1 Visual Content Embedding . . . 15
  - 3.1.1 HSV Color Embedding . . . 15
  - 3.1.2 Style Embedding . . . 16
    - Validation on Style Similarity . . . 17
    - Style Embedding Matrix . . . 17
- 3.2 Summary . . . 18

4 Data 23
- 4.1 Exploratory Data Analysis . . . 23
  - 4.1.1 Likes Data . . . 23
  - 4.1.2 Photos and Users Data . . . 23
- 4.2 Data Sampling . . . 24

4.2.1 Temporal Data Splitting 24
4.2.2 User-Photo Matrix 26

5 Empirical Evaluation **29**
5.1 Methodology 29
5.1.1 Methodology of Parameter Tuning 29
5.1.2 Methodology of Evaluation 30
5.1.3 Evaluation Metrics 30
5.2 Models' Performance on User-Photo Interaction Data 31
5.2.1 Models' Configuration 32
Nearest Neighbourhood Algorithm 32
Matrix Factorization Algorithm 32
Linear Recommendation Algorithm 33
5.2.2 Models' Performance 33
5.3 Addition of Temporal Effect to the Collaborative Filtering Algorithms 34
5.3.1 Models' Configuration 34
5.3.2 Models' Performance 35
5.4 Embedding of Users' and Photos' Metadata 35
5.4.1 Models' Configuration 36
5.4.2 Models' Performance 37
5.5 Embedding of Visual Features 38
5.5.1 Models' Configuration 38
5.5.2 Models' Performance 38

6 Conclusion **41**
6.1 Summary of Contributions 41
6.2 Directions for Future Work 42

Bibliography **43**

List of Figures

1.1 Examples of Photo Recommended on 500px Platform 3

2.1 Photo Categories in 500px Platform 9
2.2 TimeDecayExample 10

3.1 Example of HSV Feature Extraction 16
3.2 HSV Embedding Process 16
3.3 Example Outputs from Style Transfer 19
3.4 Architecture of VGG19 Convolutional Neural Network 20
3.5 Sample Images in the Style Extraction Experiment 21
3.6 Outputs from Style Extraction Experiment 22

4.1 Statistics of Likes across Time 24
4.2 Distribution of Likes across Time 24
4.3 LikeStats 26
4.4 Distribution of Likes in Sample Data 27

List of Tables

1.1 Real Life Applications of Recommender Systems 1

2.1 Three Approaches in Recommender Systems 6

3.1 Top-5 configuration in identify images with similar styles 18

4.1 Data Schema of the User Dataset 25
4.2 Data Schema of the Photo Dataset 25
4.3 Statistics of a Sample Set 27
4.4 Timeframe of a Sample Set 27
4.5 Table of constants for the User-Photo Matrix 27

5.1 Optimal Configuration - NN 32
5.2 Optimal Configuration - SVD 33
5.3 Optimal Configuration - Linear Models 33
5.4 Performance - U-I Matrix 34
5.5 Optimal Configuration - Temporal Decay Models 35
5.6 Performance - Temporal Decay Models 36
5.7 Optimal Configuration - Models on Metadata 37
5.8 Performance - Models on Metadata 37
5.9 Configuration - Visual Feature Embedded Models 38
5.10 Performance - Visual Feature Embedded Models 39

Chapter 1

Introduction

1.1 Background: Personalized Recommendation

In the past 20 years, we have seen an explosive growth of the online platforms like Amazon (E-Commerce), Netflix (Entertainment), and Facebook (Social networks). All these platforms feature tremendous amount of items [1] for their users, and the number continues to grow with their user base. The abundant and diverse selections of items attract users to come to the platform. At the same time, it also creates great challenges to make those items more discoverable. For example, for Netflix, it wants to help users discover movies that they have not yet watched but would be interested in. Otherwise, the tremendous amount of selection can lead to the information overload, which drives users away.

Recommender systems have long been used in different industries to provide automated and personalized recommendations for customers. Instead of having users browsing through one catalog after another or searching through queries, companies use recommender systems to suggest the products users would be interested and likely buy in the future. Recommender systems do so by learning users' preferences from their historical behaviors. For example, e-commerce platforms Amazon (Ekstrand, Riedl, and Konstan, 2011) and eBay (Zhang and Pennacchiotti, 2013) use effective recommender systems to suggest customers with the products that they may want to buy. The applications have also gone beyond the traditional domain of product recommendations. As we can see in the table 1.1, recommender systems have been applied extensively in the area of social network, media, and digital entertainment platforms. The items to recommend also go beyond products to include social connections, movies, and news stories, etc (Charu C. Aggarwal, 2016).

[1] The term "items" refer to products, movies, music, video, etc.

TABLE 1.1: Examples of items recommended by various platforms in real world (Charu C. Aggarwal, 2016)

Platforms	Industry	Recommended Items
Amazon.com	E-Commerce	Books, clothes, and other products
Tripadvisor	E-Commerce	Travel products
Netflix	Entertainment	Streaming Video
MovieLens	Entertainment	Movies
YouTube	Entertainment	Online Videos
Google News	News and Media	News
Facebook	Social Network	Friends, post, and advertisement
Linkedin	Social Network	Connections and post

1.2 Motivation

"A picture worths a thousand words" (Tess Flanders, 1911). Images play a very dominant role in influencing our perception of new objects. In the area of E-commerce, the pictures of products are as critical as, or even more important than, text description to help users to understand its appearance, texture, and so on. In media industry, a well-designed poster can effectively tell a good story and attract the target audiences.

Therefore, we see the value of developing algorithms that can provide personalized recommendation on pictures. To be more precise, we want to find algorithms that can learn from users' preference on pictures, identify similarity by visual features, and use those clues to provide right recommendation on photos.

To do that, we are proud to partner with 500px to explore the problem for photo recommendation. 500px is a public online community for photographers to share their photos. Now it has 13 million photographers and produced over 120 million high-quality images. Users can follow photographers, browse others' pictures, and express their preference through "Like". By observing a user's historical likes, 500px recommends photos that the user may like on his or her timeline and a dedicated newly-released "Discover" page, as shown in the Figure 1.1. This is an ideal data source to observe the patterns of users' preference on images and explore different methods of photo recommender system.

Algorithms of this kind can not only be used in suggesting photos. Essentially, they may also be used to develop a recommender system that can visually examine items. Traditionally, recommender systems mainly rely on textual metadata of items, including tags, titles, and descriptions to infer users' potential interests on the products. However, if we can observe users' preference on the visual information, we can get additional and valuable clues on recommending items to users based on the display and outlook of items. Furthermore, it can also drive the decision on how to personalize the display of products for users. A great example is how Netflix is trying to personalized the artwork of movies for each user (Ashok Chandrashekar and Jebara, 2018).

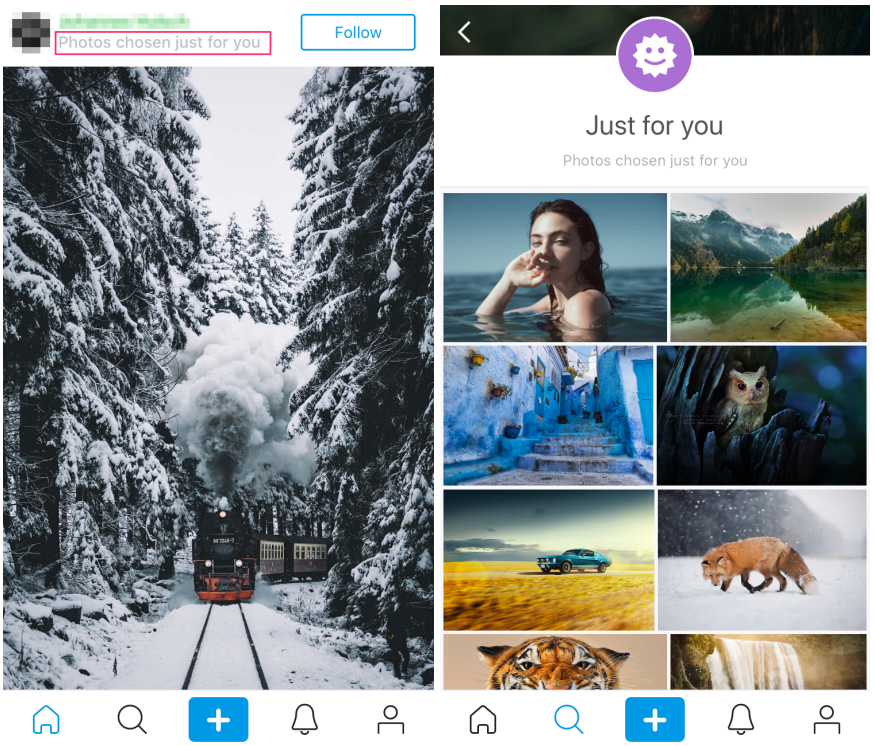
In summary, we want to create an effective machine learning algorithms to provide personalized photo recommendation. In the future, it will lead to a recommender system that can recommend all kinds of items based on their appearance.

1.3 Contribution

This thesis uses the interaction data between users and photos from 500px to develop novel photo recommender systems. We evaluate multiple existing recommender sytem algorithms, then propose an algorithm to embed human perception of photos into the system and improve the performance of recommendation.

Currently there has been very few work on photo recommendation, and most of them are relying on the textual metadata of photos in recommendation. We believe the human perception of photos should be a critical component in their reaction towards photos. Therefore, we propose a novel algorithm to quantify photos' visual features and embed them into the recommender system. Through experiment, the new algorithm shows a performance increase of 24% from the item-based nearest neighbourhood algorithm and 2.84 times better than our baseline algorithm.

Other than that, we also evaluate the existing one-class recommender system algorithms in the same dataset, including but not limited to SVD matrix factorization,



(a) (b)

FIGURE 1.1: Photos are recommended by 500px in two forms: (a) in a user's timeline page, and (b) in a dedicated section named "Just for you"

LRec, and Linear FLow. We also assess the benefit of several techniques to improve the performance of these existing algorithms. We find that the User-based nearest neighbourhood (U-NN) algorithm performs better than other existing algorithms in this dataset. Beyond that, through the technique of embedding time decay effect into the models, we successfully improve the performance of U-NN even further. It shows the users' taste does shift across time, and thus should be captured by the recommender system models.

The rest of the thesis is organized as follows. chapter 2 goes over the existing literature on photo recommendation and the general recommender system algorithms. chapter 3 describes our novel photo recommendation algorithms that quantify and embed human's perception on photos into the recommender system. chapter 4 introduces the 500px dataset and does an exploratory data analysis on the data being used in this thesis. chapter 5 presents the empirical evaluation obtained through a series of experiments to compare multiple photo recommendation algorithms and quantify the advantages brought by our novel algorithm. chapter 6 summarizes the contributions of this thesis and suggested future work in the area of photo recommendation.

Chapter 2

Background

2.1 Formal Definition of the Photo Recommendation

Photo recommendation is defined as a task to recommend the top-k photos for a particular user u. We are given the *one-class feedback* data $R \in 0, 1^{m \times n}$, where $R_{ui} = 1$ indicates the user u has liked the photo i, and $R_{ui} = 0$ indicates the preference has not been observed. The goal of the photo recommendation task is to find the best possible ordering of a set of photos for a given user.

It is also referred as the *top-k recommendation problem* (Charu C. Aggarwal, 2016). Different with the prediction-version of the recommender system (Bennett and Lanning, 2007), the *top-k recommender system* does not necessarily need to predict the actual ratings of users for specific items. Instead, the system is optimized to find the top-k recommendations for each user, and all other predictions are ignored. This is more common in real life scenario: a user is more likely to pay attention to a few items at the top of the list than the lower-ranked items (Vargas and Castells, 2011).

2.2 Approaches of Photo Recommendation

In this section, we first provide an overview of the general approaches used in the traditional recommender systems, including their respective advantages and disadvantages. Then we have a literature review on photo recommendation problems. Since the research work dedicated to photo recommendation is very limited, we also take a look at relevant fields, including image retrieval and recommendations on visual contents.

2.2.1 General Approaches in Traditional Recommender Systems

In general, there are three main approaches used in personalized recommender system:

- **Content-Based Recommender Systems (CB)** (Pazzani and Billsus, 2007): it uses a user' or item' features (titles, descriptions, keywords, etc.) to make recommendations. CB takes in users' or items' descriptive features and observed preference as labels to train a classification or regression model for the corresponding user or item.

- **Collaborative Filtering (CF)** ((Koren and Yehuda, 2010; Resnick and Varian, 1997; Goldberg et al., 1992; Herlocker et al., 2004)): it assumes users' preferences are correlated. Therefore, CF exploits the wisdom of the crowd by predicting a user' preferences from not only his or her past behavior, but also the preferences of the similar users. It is the most popular recommendation algorithm.

TABLE 2.1: Comparison of three approaches in recommender systems.

Approaches	Inputs	Advantags	Disadvantages
Content-based	A user's ratings and items' attributes OR An item's ratings and users' attributes	Good at cold-start recommendation	- Lacks of diversity (serendipity) in recommendations. - Requires a large amount of indicative features on users' or items' side
Collaborative Filtering	Ratings from a community of users to a set of items.	- Many algorithms have been proposed. - Capable of making serendipitous recommendations	Incapable in cold-starting recommendation
Knowledge-based	User's specification, items' attributes, and domain knowledge.	Good at in cold-start recommendation	- Requires explicit specification of users' requirements - Requires a good amount of domain knowledge

- **Knowledge-Based Recommender Systems (KB)**(Burke, 2002): the recommendations are made by matching users' observed requirements with similar item descriptions. It requires explicit specification of both users' need and knowledge of items. It is particularly useful in the context where items are not purchased very often but there are experts who can provide domain knowledges.

Table 2.1 provides a comparison of advantages and disadvantages of the three approaches. Overall, the *Collaborative Filtering* approach is the most attractive to us out of the three for the following reasons:

1. For the problem we are solving, we don't have explicit specification of users' preference on photos, which makes *Knowledge-based* approach inapplicable in our case. The major dataset we have is the preference data (which we will describe in details in section 4), and that is suitable for the *Collaborative Filtering*.

2. Unlike the *Content-based* approach, *Collaborative Filtering* method is more capable to make serendipitous recommendations (Herlocker et al., 2004).

3. There are many algorithms proposed for the *Collaborative Filtering* method. We will cover them in the later part of the section.

However, most *Collaborative Filtering* algorithms still face challenges in providing recommendations in cold-start settings. In the later part of the section, we will discuss the solutions to the cold-start problems in detail.

2.2.2 Literature Review: Relevant Work in Photo Recommendation

There have been some work on photo-related recommendation problems. Most of them, however, aimed at recommending clothing and other fashion product in e-commerce settings. (Yamaguchi, Kiapour, and Berg, 2013) explored several visual features, such as RGB, Lab, and Skin-hair detection, to retrieve similar clothing items from a database. (Kiapour et al., 2014) extended the idea to identifying discriminating visual elements in order to do style classification for clothes. More efforts were spent on non-visual features: (Jagadeesh et al., 2014; Y. Kalantidis and L. Kennedy and L.-J. Li, 2013; Liu et al., 2012) developed fashion-product recommender systems through textual meta-data, descriptions, and keywords in photos' annotation. (Hu, Hall, and Attenberg, 2014) developed a method to identify users' unique interests through "likes" and factorized them into latent vectors to empower product recommendation.

In a larger scope, there are also several systems developed to recommending visual contents, but through non-visual features. (Koren, Bell, and Volinsky, 2009) demonstrated superiority of matrix factorization techniques in Netflix competition. Their system was built on users' ratings and other collaborative filtering techniques, including implicit feedback and temporal effects. (Shankar et al., 2017) explored the

image recommendation problem on Pinterest dataset, mainly by using deep neural network model to embed user-image social links into compact latent dimensions.

In an adjacent field of recommender system, information retrieval, there have been several work on Content-based image-retrieval. (Xin Jin et al., 2013; Carneiro et al., 2007; Kong, Li, and Guo, 2012; Makadia, Pavlovic, and Kumar, 2008; Xia et al., 2012) explored the methods to measure image similarity with query and use it to surface relevant images in searching. All these work provided good inspiration on leveraging visual similarity between photos and queries in ranking problems. To make it applicable in photo recommendation, we need to explore the correlation between visual similarity measure and users' preference.

When it comes to the photo-recommendation specifically, there has been no previous effort in exploring the use of visual features. Given the popularity of deep neural networks in computer vision field, we have seen awesome image-related application, including object detection in photos (Deng and Yu, 2014; LeCun, Bengio, and Hinton, 2015), style transfer (Gatys, Ecker, and Bethge, 2015a), image synthesis (Reed et al., 2016), caption generations (Karpathy and Fei-Fei, 2017), and so on. With all of those progresses, we believed there are great opportunity to explore the use of visual features in photo recommendation problem.

2.3 Algorithms in Recommender Systems

In this section, we will introduce different algorithms of the recommender systems proposed through the past research. Later on, we explore the application of these algorithms in our dataset, and their performance will be evaluated in the section 5.

2.3.1 Nearest Neighbourhood models (NN)

Nearest Neighbourhood-based Collaborative Filtering models (Herlocker et al., 1999; Bell and Koren, 2007; Sarwar et al., 2001; Linden, Smith, and York, 2003) are based on the fact that users with similar tastes show similar patterns of rating or likes, and similar items share similar patterns in the ratings or likes received. In general, there are two approaches to the nearest neighbour-based models.

1. **User-based Nearest Neighbourhood Model (U-NN)** (Herlocker et al., 1999; Bell and Koren, 2007) computes the similarity between users according to their user-item preference data, and recommends a user with items liked by the other users who share similar taste. The detailed computation can be found in equation 2.1.

2. **Item-based Nearest Neighbourhood Model (I-NN)** (Sarwar et al., 2001; Linden, Smith, and York, 2003) computes the similarity between items (photos in our case) according to their user-item preference data. It recommends a user with items that are similar to what he or she has liked before. The detailed computation can be found in equation 2.2.

$$R_{ui} = \sum_{v \in (S_{uv} \geq \alpha)} S_{uv} \cdot R_{vi} \tag{2.1}$$

$$R_{ui} = \sum_{j \in (S_{ij} \geq \alpha)} S_{ij} \cdot R_{uj} \tag{2.2}$$

In equation 2.1 and 2.2, there is a constant α that serves as a minimum similarity threshold. It is used to remove noises from irrelevant entities. Take the user-based nearnest neighbourhood model as an example: when predicting the recommendations for a user u, any user whose similiarity measure with the user u is below the threshold $alpha$ will be dropped. By doing so, we can remove the impacts from irrelevant users in the recommendation.

Similarity Metrics

The performance of the Nearest Neighbourhood models relies heavily on the the similarity metrics used. In this thesis, we used and evaluated four common similarity metrics, including:

Cosine Similarity, which measures the angle between two vectors. The cosine similarity between user u and v is computed as

$$S_{uv} = \frac{< R_{u:}, R_{v:} >}{||R_{u:}||_2 ||R_{v:}||_2}$$

where $R_{u:}$ indicates the vector of likes of the user u, $R_{v:}$ indicates the vector of likes of the user v, and $< ., . >$ denotes inner product.

Jaccard Similarity, which measures the similarity between two sets. The Jaccard similarity between user u and v is computed as

$$S_{uv} = \frac{|I_u \cap I_v|}{|I_u \cup I_v|}$$

where I_u indicates a set of reactions (observed and unobserved) on all the items by the user u, and I_v indicates a set of values of the user v.

Euclidean similarity, which is based on the euclidean distance between two vectors. Euclidean distance measures the straight-line distance between two vectors. The larger the distance, the less similiar the two vectors are. The Euclidean similarity between user u and v is computed as

$$D_{uv} = \sqrt{\sum_{i \in I_{uv}} (R_{ui} - R_{vi})^2}, \; S_{uv} = \frac{1}{1 + D_{uv}}$$

where R_{ui} indicates the reaction of the user u on the item i in the User-Item matrix, and R_{vi} indicates the reaction of the user v on the item i.

All of the functions above measure the similarity between users. The similarities between items S_{ij} can be computed in the same way.

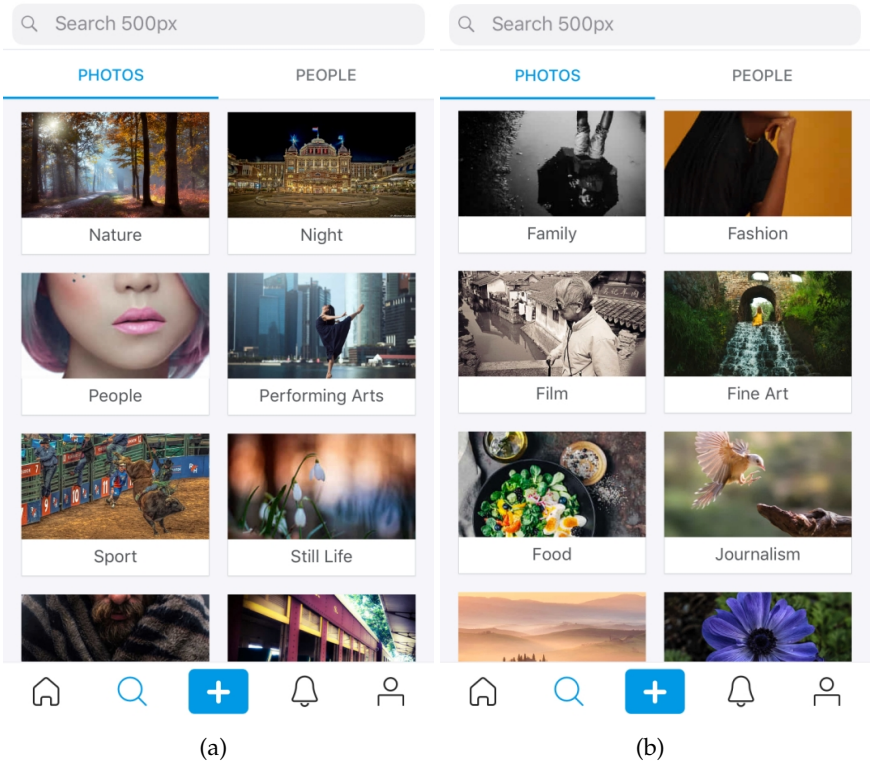


FIGURE 2.1: Examples of Photo Categories in 500px Platform

Use of Metadata in Cold-Start CF

In addition to the User-Item matrix, we can also add metadata as extra clues to measure similarity. The additional metadata is useful especially in the cold-start problems, when there are not enough ratings or likes being observed, so we can leverage the user's metadata to find the similar users, or item's metadata for similar items.

In our problem of photo recommendation, an example of the metadata for the photos is *Category*. In 500px's website, there are 30 pre-defined photo categories, such as Sport, Fashion, and Travel (examples can be found in the Figure 2.1). When a user upload a photos, the user can select the category that the photo belongs to. The complete list of metadata we used in the user and item-based Nearest Neighbourhood methods can be found in table 4.1 and 4.2, where the rows for the selected metadata are highlighted.

2.3.2 Temporal Decay Models

Users' preferences on products may change over time. The preferences can be impacted by products' popularity and seasonality, or users' own changes in taste. (Koren and Yehuda, 2009) incorporated the temporal dynamics into collaborative filtering models and demonstrated that capturing time drifting patterns in user behavior

is essential to improving the performance of the recommender system. In this thesis we will explore the incorporating the effect of temporal decay into the nearest neighbourhood models.

Take the item-based nearest neighbourhood model as an example. The hypothesis is: when predicting whether the photo i should be recommended to the user u, the weight of a photo j_1 should decay by a larger degree than the weight of another photo j_2, given that the photo j_2 was liked by the user u more recently than j_1, as illustrated with the figure 2.2. The detailed computation can be found in equation 2.3:

$$R(u,i) = \sum_{j \in I} S_{i,j} \cdot e^{-\lambda * T_{\Delta}(u,i,j)} \cdot R_{u,j} \tag{2.3}$$

where $T_{\Delta}(u,i,j)$ is a pre-defined function to compute the time difference, which we are going to cover in the following section.

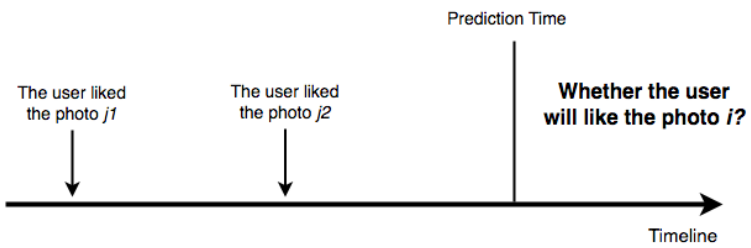


FIGURE 2.2: An example to illustrate the time decay

Defining Time Difference

The time difference function $T_{\Delta}(u,i,j)$ can be defined as the difference between time when the user u liked the item j and a given time when we need to predict for R_{ui}.

However, in the setting of our experiment, the likes in the training set, the validation set, and the test set are in the chronological order. Therefore, everytime when we make the prediction if a user u will like a photo i, the time of R_{ui} is unknown to us. In such a case, the reference time in the time difference function becomes arbitrary. In this thesis, we use two ways to define the time for R_{ui}:

1. The time in the beginning of the test phase. For example, if we are predicting for the likes in the validation set, we take the beginning time of the validation set as our reference time.
2. The time in the middle of the test phase. Still the example of the validation set, we take the middle time-point of the validation set as our reference time

Both of the approaches will be evaluated and the performance results will be covered in the Section 5.

2.3.3 Matrix Factorization Models (MF)

In a typical recommender system problem, the User-Item matrix is very sparse; therefore, it is possible to use matrix factorization models to embed the users' and items' vectors into lower-rank latent space. Assuming the user-item preference data is correlated, matrix factorization models aim to express a given user's or item's vector with a linear combination of factor vectors (Fawcett, Mishra, and Knowledge Discovery & Data Mining (2003 : Washington, 2003; Rusian Salakhutdinov, 2009; Koren, Bell, and Volinsky, 2009).

The general matrix factorization can be expressed as an optimization process, whose goal is to

$$min_\theta \sum_{u \in U, i \in I} J_{ui} \cdot \mathcal{L}(R_{ui}, \hat{R}_{ui}(\theta)) + \Omega(\theta) \tag{2.4}$$

where $J \in R_+^{m \times n}$ is a pre-defined weighting matrix, $\mathcal{L} : R \times R \rightarrow R_+$ is some loss function, typically squared loss $\mathcal{L}(y, \hat{y}) = (y - \hat{y})^2$, and $\Omega(\theta)$ is the regulariser, usually the L-2 regulariser:

$$\Omega(\theta) = \frac{\lambda}{2} \cdot (||A||_F^2 + ||B||_F^2)$$

After the optimization, we can get the matrices $A \in R^{k \times m}, B \in R^{k \times n}$ as the latent representation of m users and n users, with $k \in \mathbb{N}_+$ being the latent dimension. Then the recommendation matrix $\hat{R}_\theta$ is computed with

$$\hat{R}(\theta) = A^T B$$

for $\theta = \{A, B\}$.

Embedding of the Meta Data

A photo's metadata can also be correlated with the user-photo interaction; therefore, we can also embed a photo's metadata into the lower-rank latent vector to describe the photo. The same for the user's metadata (Stern, Herbrich, and Graepel, 2009).

Take the photos' metadata as an example. We can concatenate the photos' meta data $P \in R_+^{n \times q}$ to the User-Photo preference matrix R and get a new matrix $R' \in R_+^{m \times (n+q)}$. Then we can factorize the new matrix R' with equation 2.5.

Choice of the Weighting Matrix

Due to the nature of the dataset from 500px, we are dealing with the One-Class Collaborative Filtering problem (OCCF), where the negative samples are unobserved from the data. As a result, we cannot simply set $J_{ui} = [\![R_{ui} > 0]\!]$, or the model can only learn from the observed positive preference (likes).

In this thesis we set $J_{ui} = 1$ uniformly, which essentially treats all unobserved likes as indications of a negative preference. The benefit of this approach is that we are transforming the collaborative filtering problem into a standard matrix factorization problem:

$$min_{\theta} \sum_{u \in U, i \in I} \mathcal{L}(R_{ui}, \hat{R_{ui}}(\theta)) + \Omega(\theta) \tag{2.5}$$

We can solve the equation 2.5 with existing techniques like Singular-value decomposition.

2.3.4 Linear Models

For one-class collaborative filtering problem, (Sedhain et al., 2016a; Sedhain et al., 2016b; Sedhain, 2016) proposed linear models that formulate recommendation as a classification problem.

The LRec Model

LRec converts the one-class collaborative filtering into a number of logistic regression subproblems. First we define a matrix $X \in \mathbb{R}^{n \times m}$ by transposing the preference matrix:

$$X = R^T$$

Then for each user $u \in U$, we define a binary target vector $y^{(u)} \in \{\pm 1\}^n$ by:

$$y^{(u)} = 2R_{u:} - 1$$

.

Then, LRec solves

$$\min_{\theta} \sum_{u \in U} \sum_{i \in I} \mathcal{L}(y_i^{(u)}, X_{i:} w^{(u)}) + \Omega(\theta) \tag{2.6}$$

where $\mathcal{L}$ is a pre-defined convex loss function, and $\theta = \{w^{(u)}\}_{u \in U}$.

From the model we can learn a weight vector $w^{(u)}$ for each user u. We can interpret an element $w_v^{(u)}$ from the weight vector as the relevance of the likes of one user v in predicting the likes of the user u. The recommendation vector for a user u can be computed as:

$$(w^{(u)})^T \cdot R$$

Based on the formulation, we can found that due to the design of LRec model, the training process for individual users is highly parallelisable. Furthermore, we

can define the $\mathcal{L}$ as logistic loss function and leverage the Logistic Regression solver from scikit-learn library in Python to train all the models.

The Linear-FLow Model

The LRec method still requires solving a large number of regression subproblems; thus, it cannot be easily scalable to large collaborative filtering problems. To make linear models usable for large scale problems, (Sedhain et al., 2016b; Sedhain, 2016) also proposed a new algorithm Linear-FLow to speed up the linear recommendation through lower-dimensional factorization. Linear-FLow model uses latent factors to learn the similarity among users or items (in our problem, photos) and then express the user-item matrix with the latent factors.

Take item-based Linear-FLow model as an example. It embeds the item-item similarity into the matrix Q_kY. Q_k is latent matrix with shape $n \times k$ ($k \ll n$) and is an orthogonal basis of the row space of the user-item matrix R. Therefore, it can be computed with SVD algorithm. In order to compute the other latent matrix Y, we use the equation **??**:

$$\underset{Y}{argmin}||R - RQ_kY||_F^2 + \lambda||Y||_F^2 \tag{2.7}$$

The analytical solution is written as:

$$Y = (Q_k^T R^T R Q_k + \lambda I)^{-1} Q_k^T R^T R$$

After optimization, we find the photo-photo similarity through Q_kY, and the recommendation matrix can predicted as RQ_kY.

Similarly, the user-based Linear-FLow model is expressed in the equation 2.8:

$$\underset{Y}{argmin}||R - YP_k^T R||_F^2 + \lambda||Y||_F^2 \tag{2.8}$$

The corresponding analytical solution is written as:

$$Y = R(P_k^T R)^T (P_k^T R P_k R^T + \lambda I)^{-1}$$

2.4 Summary

This chapter reviewed the past work in recommender systems research. We formally defined our photo recommendation problem and discussed the relevant work in the field. We also provided an overview on the existing algorithms that will be evaluated

in our dataset. In the next section, we will propose our novel photo recommendation algorithms.

Chapter 3

Novel Photo Recommendation Algorithms

In the section 2, we pointed out there is very few research work in the area of photo recommendation. In this section, we are going to introduce our novel photo recommendation algorithms to tackle the problem.

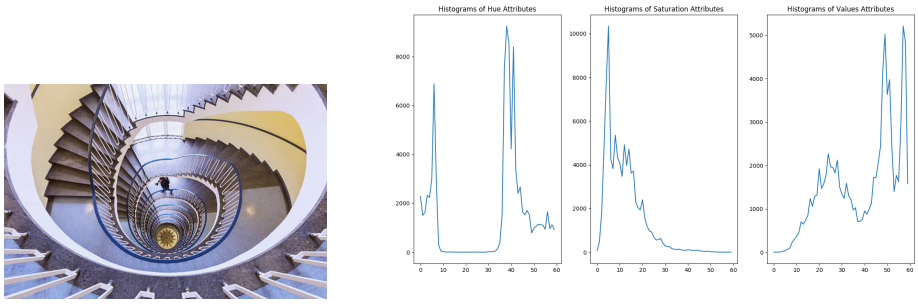
3.1 Visual Content Embedding

In the section 2.2.2, we described that there have been efforts to use metadata of photos, including the categories, tags and titles of the photos, to make photo recommendation. In reality, since we are dealing with photos, we have more features available than those metadata: the visual content in the photos. In this thesis, we proposed the embedding of the HSV color space and the style feature of the photos in the recommender systemi.

3.1.1 HSV Color Embedding

The HSV color space was designed to in order to more closely capture the human perception on color-making attribute (Cheng et al., 2001; Sural, Gang Qian, and Pramanik, 2002). Different with the traditional RGB color space, HSV color space separates out the luminance, or light intensity, from the color information. In addition to the colorfulness, HSV color space is able to describe the brightness and whiteness of an image. Therefore, the measurement of HSV space of an image can represent our view of the image from multiple aspects.

We did experiment to see if the embedding of HSV color space of photos can improve the performance of our photo recommender system. In implementation, for each photo, we generate a histogram with 60 bins from each of the HSV color channel and concatenate them to create a one-dimensional feature vector. By joining these feature vectors from all the photos, we get a HSV embedded matrix. Then we concatenate the User-Photo interaction matrix with the HSV embedded matrix.



(a) (b)

FIGURE 3.1: Example of HSV Feature Extraction: (a) shows the original photo (b) shows the histograms from each of the HSV color channel with 60 bins

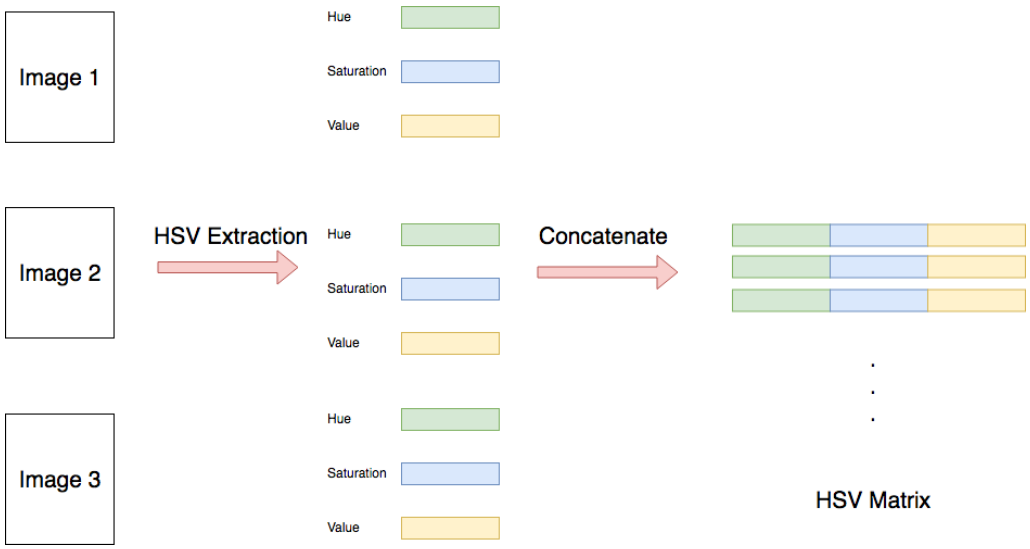


FIGURE 3.2: The Process of creating a HSV embedded matrix: for each photo, we generate a histogram from each of the HSV color channel. Then we concatenate the histograms into one feature vector. After joining these feature vectors from all the photos, we get an HSV embedded matrix

Finally, we solve it with the collaborative filtering methods and linear models described in section 2.3. Figure 3.1 shows an example photo with its HSV histograms, and the Figure 3.2 shows the process to create an HSV embedded matrix.

3.1.2 Style Embedding

The style of a photo is a strong visual perception, but it is very difficult to described with text. Therefore, we are not able to extract style-relevant information based on the tags or descriptions of photos. We have to find other ways to quantify the style information of photos.

(Gatys, Ecker, and Bethge, 2015b) shows an interesting application of Deep Neural Networks in artistic style transfer. Through a deep convolution neural network, we can separate the content and style of arbitrary images, and recombine them to create a new image. For example, In figure 3.3, we successfully extracted the style attributes from Van Gogh's *Starry Night* and Pablo Picasso's *Ma Jolie* respectively, and transfered that to a photo taken in University of Toronto.

(Gatys, Ecker, and Bethge, 2015b) extracted the style of a given image by applying the Gram matrix to the output from a hidden layer in the deep convolutional neural network. If we can use this method to quantify the style information of a given photo, and take one step further, to measure the style similarity between different photos, then we can embed it into the collaborative filtering model.

Validation on Style Similarity

In order to validate the idea, we need to show if a convolutional neural network can successfully identify photos that are similar in style, and find the best similarity metric. We use a pre-trained VGG network that was used in the paper (Gatys, Ecker, and Bethge, 2015b) to process each photo. The architecture of the VGG-19 network is shown in the figure 3.4. It contains 16 convolutional layers and these layers are divided into 5 blocks. We also want to find out which block's output is optimal to measure style similarity between photos.

We set up an experiment with multiple sets of paintings and photos that are similar in style. Each set contains 3 images. The sample image sets are shown in the figure 3.5. In the experiment, we want to evaluate which output block from VGG-19 and which similarity metric can best find the images in the same set. We use VGG-19 Network to process all of the images, and vectorize the output from each block in VGG-19. Then we apply each of the three similarity metrics listed in 2.3.1 to compute the similarity between the output vectors. In the end, for each similarity metric on the output sets from each block, we return the top k pairs of photos that have the highest similarity score. We want to find out the configuration that returns the maximum number of correct pairs. We use *Precision@k* to measure the results, which we will explain in detail in section 5.1.3.

The top-5 configurations are listed in the Table 3.1. Clearly the Euclidean Similarity metric on the Block No.3 gives us the highest precision at top 10 and top 15 results. To take a closer look at its actual performance, we also print out the top 5 pairs returned from the configuration in Figure 3.6. As we can see from the actual outputs, it gets all the top 5 pairs correct. This experiment validates the idea of quantifying style similarity between photos and provides us the optimal configuration to measure the similarity.

Style Embedding Matrix

After the experiment, we get the optimal configuration to extract the photo's style information from the Convolutional Neural Network. In actual implementation, we

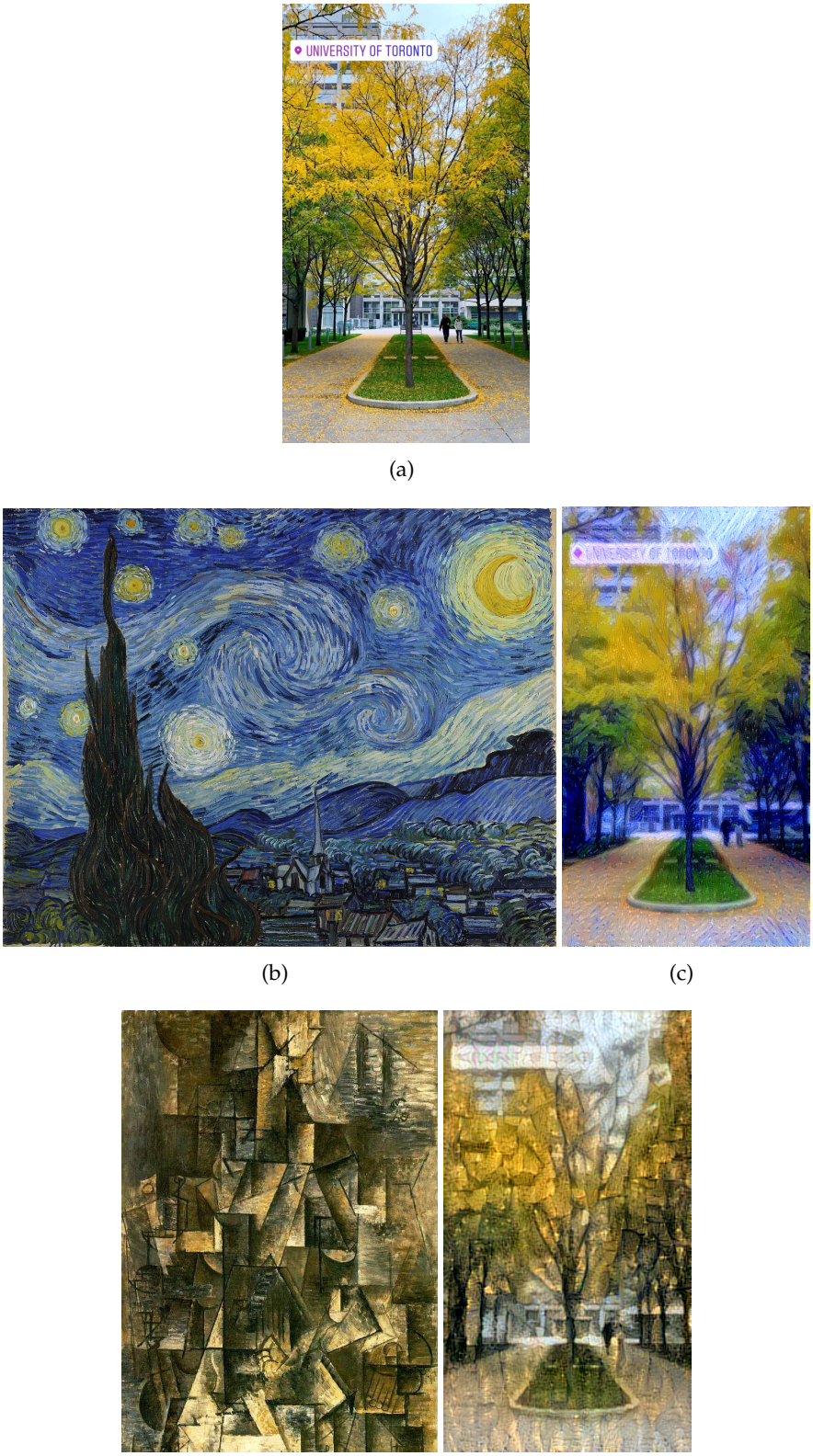
TABLE 3.1: Top-5 configuration in identify images with similar styles

Rank	Similarity Metric	Block Index	Prec@10	Prec@15
1	Euclidean	3	0.7	0.53
2	Euclidean	2	0.7	0.47
3	Jaccard	4	0.6	0.47
3	Jaccard	2	0.6	0.47
4	Jaccard	3	0.6	0.4

use the VGG-19 network to process each photo, select the output of a block No.3 from the model and vectorize it. Similar to the HSV matrix in 3.1.1, we join these feature vectors from all the photos to form a style embedded matrix. In the result section, we will show how we select the output layer and use the style embedded matrix in the collaborative filtering method.

3.2 Summary

In this chapter, we presented the approaches to quantify visual contents from photos and embed them into matrices. The visual contents including the HSV color space and style information of photos. We did an experiment to validate the idea that we can quantify the style information of different photos and thus measure the style similarity between photos. Later on, we will do experiment on the real world dataset from 500px and show the performance output from the experiment.



(a)

(b) (c)

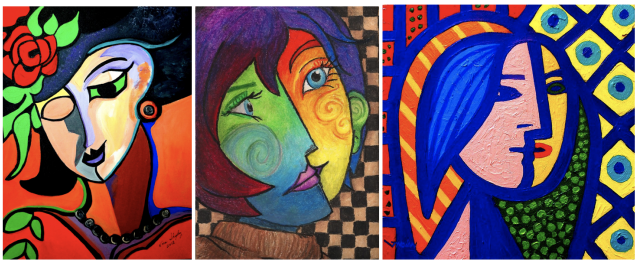
(d) (e)

FIGURE 3.3: Examples of Style Transfer: the image (c) combines the content from (a) and style attributes from the image (b), while the image (e) combines the content from the image (a) and style attributes from the image (d)

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

FIGURE 3.4: The architecture of the VGG-19 Convolutional Neural Network. Its 16 convolutional layers are divided into 5 blocks.

(a)



(b)

(c)

(d)

(e)

(f)

FIGURE 3.5: Sample images used to evaluate how effective is the VGG-19 Network in identifying images that are similar to each other. Each group contains three paintings or photos that are similar to each other in style.

FIGURE 3.6: Top 5 pairs of images by similarity found through the Euclidean Similarity metric on the Block No.3 from VGG-19 Network.

Chapter 4

Data

We are honored to partner with 500px to explore our photo recommendation algorithms on their datasets. In this chapter, we will first do an exploratory data analysis on 500px dataset. Then we will explain how we do data sampling as part of our experiment setup.

4.1 Exploratory Data Analysis

An exploratory data analysis is conducted in order to understand the dataset better. The dataset is from the partner of this thesis research project, 500px.com. There are three datasets: the likes data, the photos data, and the users data.

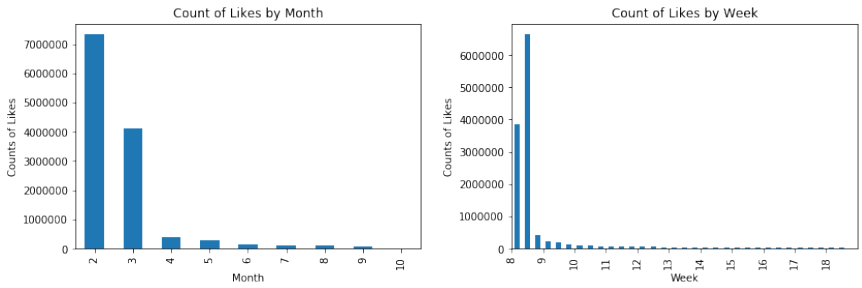
4.1.1 Likes Data

The most important dataset, likes dataset, contains three columns: the "photo_id" column that maps to the same column in the photo dataset, the "user_id" column that maps to the same column in the user dataset, and the "timestamp" column that records the time when a user likes a photo.

Through the distribution of likes by date (Figure **??**), the likes mostly happened in the 8th and 9th week of 2017. The upload time of photos is missing from all the datasets. With confirmation from 500px team, the photos are uploaded from the timeframe of February 24, 2017 to March 2, 2017. If we look at the average distribution of likes by photos after the first like (Figure 4.2), we can find most of the likes happened in the first three days of upload.

4.1.2 Photos and Users Data

There are 225,922 users in the user dataset and 300,000 photos in the photo dataset. Each user listed in the likes dataset can be found in the user dataset, and it is the same to the photos dataset. A summary of the dataset schema of the two datasets



(a) (b)

FIGURE 4.1: (a) The count of likes by months (b) The count of likes by weeks

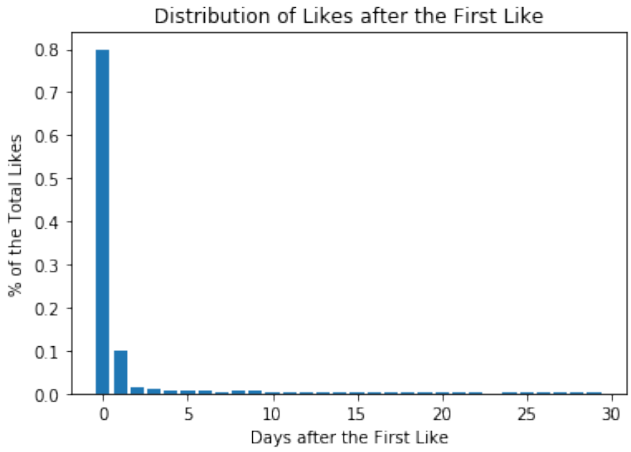


FIGURE 4.2: Distribution of likes after the first like (Average across all photos)

can be found in Table 4.1 and Table 4.2., along with the descriptions of each column and the percentage of completeness.

4.2 Data Sampling

4.2.1 Temporal Data Splitting

In our experiment, we need to tune the hyperparameters for each model and then evaluate their performance. As pointed out by (Cawley and Talbot, 2010), we will risk the overfitting problem if we use the same test set in both parameter tuning and performance evaluation. For this reason, we used nested cross validation method so that we can get an unbiased evaluation of our models.

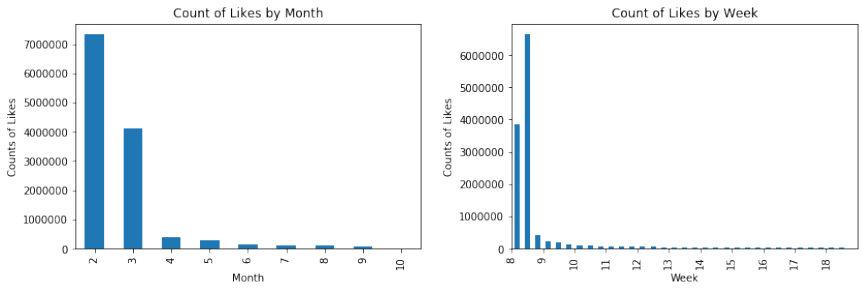
We prepared 5 data samples for nested cross validation. Each sample comes with a training set, a validation set, and a test set. All the datasets in the same sample have the same set of users and items and they were splitted by time. The likes in the

TABLE 4.1: Data Schema of the User Dataset. The highlight columns are used as users' metadata in the models.

Column	Data Type	Description	% NAN
User ID	String	Unique ID for each user	0.0
Gender	**Binary**	**A binary value to indicate the gender of the user**	**0.0**
Birth Year	Integer	The birth of the year of the user	64.8
Country	String	The country where the user lives currently	30.5
Equipments	String	Gears the user is using, including brands and models	78.2
# of followers	**Integer**	**Number of the followers the user has**	**0.0**
# of friends	**Integer**	**Number of the accounts the user is following**	**0.0**

TABLE 4.2: Data Schema of the Photo Dataset. The highlight columns are used as metadata in the models.

Column	Data Type	Description	% NAN
Photo ID	String	Unique ID for each photo	0.0
User ID	String	ID of the user who took the photo	0.0
thumbnail url	String	URL of the photo's thumbnail	0.0
Category	**String**	**One of the 30 pre-defined photo categories on the website**	**0.0**
Keywords	**String**	**Keywords that the user put for the photo**	**21.9**
Width	Integer	Width of the photo	0.0
Height	Integer	Height of the photo	0.0
Editor's Choice	**Binary**	**Whether the photo has been picked by the editors in 500px**	**0.0**
Camera	String	Gear used to take the photo	24.3
Lens	String	Type of the lens to take the photo	58.7
Aperture	Double	Aperture used to take the photo	27.5
Shutter Speed	String	Shutter speed used to take the photo	27.5
ISO	Integer	ISO used to take the photo	26.2
Focal Length	Integer	Focal length used to take the photo	26.9
Location	String	The longitude and latitude where the photo was taken	53.5



(a) (b)

FIGURE 4.3: (a) The count of likes by months (b) The count of likes by weeks

validation set comes later than those in the training set, and the test set is after the validation set. This can be shown through the Table 4.4, which lists the time frames of a sample. As all the photos were uploaded in one week and majority of the likes happened within the first three days, the timespan between the samples is really narrow. Therefore, there are certain overlap between the data samples.

4.2.2 User-Photo Matrix

For each dataset, we created a User-Photo matrix for the purpose to implement the Collaborative Filtering methods. The matrix is described with constants defined in the Table 4.5. Each user represents a row in the matrix and each photo represents a column. The value in the matrix is based on if a given user likes a photo: it will be 1 if the user liked the photo during the timeframe of the dataset, and 0 represents either the user has not seen the photo, or disliked it. The definition is formalized in Equation 4.1 and Equation 4.2, and the Table (sampleui) shows an example of the matrix.

Table 4.3 gives a statistics of a sample and Figure 4.4 shows detailed distribution through histograms. The plots on the histograms are very steep logarithmic curve, so it is very sparse matrix.

$$X \in R^{M \times N} \tag{4.1}$$

$$X_{(u,i)} = \begin{cases} 0, \text{ if user } u \text{ disliked or has not seen the photo } i \text{ during the defined timeframe} \\ 1, \text{ if user } u \text{ liked the photo } i \text{ during the defined timeframe} \end{cases} \tag{4.2}$$

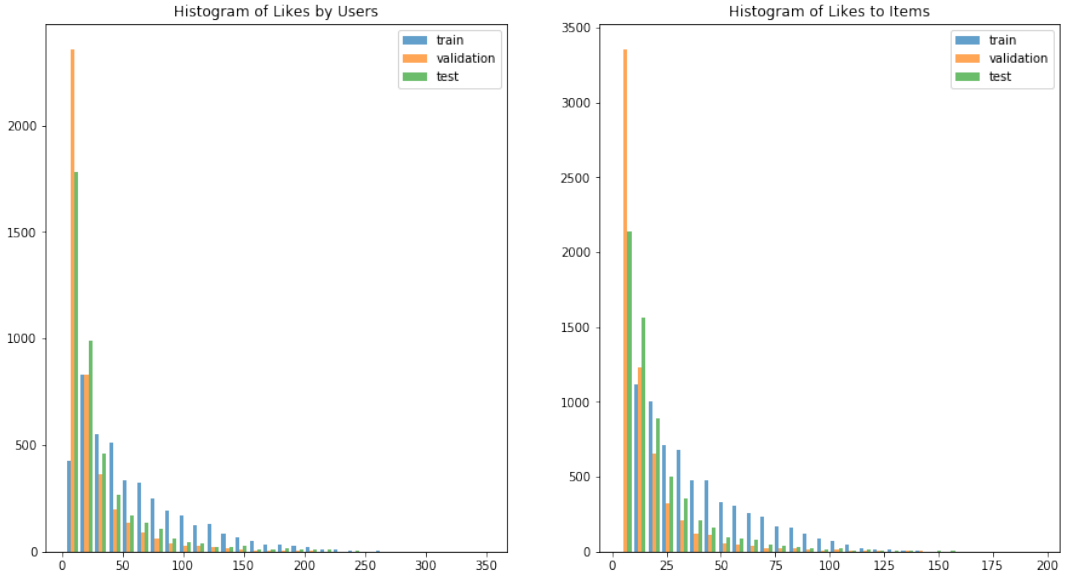


FIGURE 4.4: Distribution of Likes of Users (left) and Items (right) in a Sample

TABLE 4.3: Statistics of a Sample Set

Dataset	Measurement	Min	Mean	Max
Train	# Likes by a user	10	57	299
Train	# Likes to a photo	10	40	162
Validation	# Likes by a user	3	20	199
Validation	# Likes to a photo	3	14	181
Test	# Likes by a user	5	27	234
Test	# Likes to a photo	5	19	167

TABLE 4.4: Time Frame of a Sample Set

Dataset	Time of the first like	Time of the last like
Train	2017-02-24 14:47:26	2017-02-28 23:59:58
Validatiion	2017-03-01 00:00:04	2017-03-07 23:59:46
Test	2017-03-08 00:00:27	2017-10-03 12:49:44

TABLE 4.5: Table of constants for the User-Photo Matrix

Constant	Description
M	Number of unique users
N	Number of unique photos

Chapter 5

Empirical Evaluation

When developing machine learning systems, the evaluation of a given model is important. The evaluation helps us quantify the performance of the model and make side-to-side comparison with other models. In this chapter, we will first explain our methodology and metrics used in evaluation, then we will present out findings from model evaluation.

We introduced the existing recommender system algorithms in section 2.3 and our novel photo recommendation algorithms in chapter 3. In this section, we want to compare these algorithms through a series of experiments. In each experiment, we will introduce the questions we want to answers and the insights from the results.

5.1 Methodology

As described in (**datasampling**), we use the nested cross validation in order to achieve an unbiased evaluation of our models. In this section, we will explain our parameter tuning and model evaluation methods.

5.1.1 Methodology of Parameter Tuning

For each model, we have one or more parameters to tune. We use grid search method to find all possible sets of parameters and evaluated the result from each of them with our metrics. The set of parameters that yield to the optimal metrics measurements are selected as the optimal parameter values.

We tuned the parameters through the validation sets. As mentioned in the section 4.2, we have altogether five data samples. In each of the five data samples, there is one training set and one validation set. For parameter tuning, we train a given model with one training set, and evaluate the prediction against the validation set in the same sample. The objective of the prediction is to find what photos a given user will like next; in other words, the prediction should be as similar to the validation set as possible, since the likes in the validation set happens after the training set.

The evaluation was based on a set of pre-defined metrics (refer to metrics), and the results were averaged across all the samples for the purpose of cross-validation.

5.1.2 Methodology of Evaluation

In each data samples there is also a test set. In the test phase, we train the well-tuned models with both the training and validation set, and evaluate their results side-by-side with the test set in the same sample. Similar to the validation phase, the same set of metrics are used and the results from all the samples are collected to compute the average performance and the confidence interval.

5.1.3 Evaluation Metrics

We use 4 ranking metrics throughout the thesis to evaluate the models, including *precision@k*, *precision@k*, $R - Precision$, and *Average Precision* as the target metrics. Ranking metrics are widely used to evaluate the performance of recommender system.

To define the 4 ranking metrics, let I_u be the set of items (photos) that were liked by a user u, and I_u^{rec} be the sorted list of recommended items for a user u.

Precision@k (P@k) measures the fraction in the top-k recommended list that are correctly identified as positives. In this thesis, we define the *precision@k* for a user u as the fraction of the top k recommended photos that were liked by the user u:

$$precision@k(u) = \frac{|I_u \cap I_u^{rec}[:k]|}{k}$$

$$precision@k = \sum_{u=1}^{m} \frac{precision@k(u)}{m}$$

Mean Average Precision (MAP) measures the precision of the recommendation with consideration on the order of the item in the recommended list. It is a widely-used ranking metric in recommender system because the single figure can reflect the performance across all recall levels. It is defined as:

$$AP(u) = \sum_{i=1}^{n} \frac{precision@i(u) \cdot relevance(u,i)}{|I_u|}$$

$$MAP = \sum_{u=1}^{m} \frac{AP(u)}{m}$$

where $relevance(u,i) = 1$ if the top i^{th} recommended photo is liked by the user u, else 0.

R-Precision (R-Prec) is a good supplement for the *Precision@k*. A disadvantage of *Precision@k* is that the total number of relevant items may differ significantly across users and can largely influence the *Precision@k*, making it very unstable. $R - Precision$ is another form of *Precision@k* by dynamically adjusting the k according to the number of the relevant items. It is defined as:

$$R - Precision(u) = \frac{|I_u \cap I_u^{rec}[: |I_u|]|}{|I_u|}$$

$$R - Precision = \sum_{u=1}^{m} \frac{R - Precision(u)}{m}$$

Recall@k (R@k) measures the fraction of all positives that are correctly identified in the top-k recommended list. In this thesis, we define the *recall@k* for a user u as the fraction of all the photos being liked by the u that are correctly recommended in the top-k list:

$$recall@k(u) = \frac{|I_u \cap I_u^{recl}[: k]|}{|I_u|}$$

$$recall@k = \sum_{u=1}^{m} \frac{recall@k(u)}{m}$$

5.2 Models' Performance on User-Photo Interaction Data

In this experiment, we applied our algorithms on pure user-photo interaction data. We want to assess each algorithm's capability on learning from the user-photo interaction alone and use that for prediction, without embedding of any categorical data or visual attributes. Later on, we will use the performance recorded from this experiment as benchmark scores for the following experiments, so that we can assess the value of the additional techniques to photo recommendation.

We will compare the performance of the following algorithms:

1. Two forms of Nearest Neighbourhood Algorithms (NN), as discussed in subsection 2.3.1: the item-based nearest neighbourhood algorithm (I-NN) and the user-based nearest neighbourhood algorithm (U-NN)
2. SVD Matrix Factorization Algorithm (SVD), as discussed in subsection 2.3.3
3. LRec (LRec), as discussed in section 2.3.4
4. Two forms of Linear FLow (LFlow), as discussed in section 2.3.4: the item-based Linear FLow (LFlow-I) and the User-based Linear FLow (LFlow-U)

TABLE 5.1: The optimal configuration of the nearest-neighbour methods after the parameter tuning

Model	Similarity Metric	Parameter(s)
I-NN	Cosine	Similarity Threshold: 0.26
U-NN	Cosine	Similarity Threshold: 0.06

Different algorithms are learning different aspects from the same user-photo interaction data: for nearest-neighbourhood algorithms (NN), they are learning similarity between users or between photos. For latent factorization algorithm (SVD), it is learning each user and photos in latent space. For LRec algorithm (LRec), it is learning the relevancy between users. For Linear FLow algorithms (LFlow), they are learning the relevancy between users or photos in latent space.

We also evaluate the performance of the following two benchmark algorithms to compare with algorithms listed above:

1. Random: it will randomly select photos and recommend them to users.
2. Popularity-based algorithm (popular): it will select the most popular photos from the training set and recommend them to all the users. Photos will be ranked by the number of likes they received in the training set.

5.2.1 Models' Configuration

Nearest Neighbourhood Algorithm

As described in subsection 2.3.1, for the Nearest-Neighbourhood based algorithms, they have a parameter on the minimum threshold of similarity. In addition, we also need to evaluate the performance on different similarity metrics introduced in section 2.3.1. After tuning the parameters with the validation sets, we get the optimal configuration of all the models, as shown in the table 5.1.

Out of the three similarity metrics, Cosine Similarity shows superior performance than other metrics, Jaccard and Euclidean Similarity, so we use Cosine Similarity metric in each model.

Matrix Factorization Algorithm

As introduced in 2.3.3, we use singular-value decomposition method to solve the equation 2.5. Therefore, we only have one parameter to tune: the length of the latent vector k.

However, for a user-photo interaction matrix R of different size, the optimal latent vector size should be different. Therefore, instead of tuning for a constant k, we

TABLE 5.2: The optimal configuration of the Matrix Factorization models after parameter tuning

Model	Parameter
SVD	k factor: 0.09

TABLE 5.3: The optimal configuration of the Linear Recommendation models after the parameter tuning

Model	Parameter(s)
LRec	Parameter of Regularization: 0.1
LFlow-I	k factor: 0.53; Parameter of Regularization: 75
LFlow-U	k factor: 0.33; Parameter of Regularization: 75

tune a *factor of k*: the ratio between the length of the latent vector vs. the length of the input user vector, or the number of items n.

After tuning the parameters with the validation sets, we get the optimal configuration of all the models, as shown in the Table 5.2.

Linear Recommendation Algorithm

As described in section 2.3.4, the *LRec* model can be solved through logistic regression. The only hyperparameter in the model is the penalty parameter in the logistic regression.

As for the *Linear-FLow* model, there are two variations: item-based (equation 2.7) and user-based equation (2.8). Both of the models the parameters include the size of the latent dimensions and the parameter of regularization. For the latent dimension size, similar to the Matrix Factorization model in section **??**, we tune the ratio to the number of items, *k-factor*, instead of the constant k.

After parameter tuning on the validation sets in our five sample data, we get the optimal model configuration shown in the Table 5.3.

5.2.2 Models' Performance

All of the algorithms are evaluated together with the two benchmark models. The results of the performance can be found in the Table 5.4.

First, we can see the performance of all the target algorithms exceeds the benchmark algorithms by more than 100%. Among all the models, U-NN model has the best performance across all the metrics, and it is **3.6 times** better than the item average benchmark model in *Precision@10* and *Average Precision*.

TABLE 5.4: Performance of Models trained on pure user-photo interaction data. The results come from the test sets from 5 data samples. The best measurement in each column is highlighted in bold.

Model	Precision@10	Avg Precision	R-Precision	Recall@10
Random	0.0058 ± 0.0021	0.0060 ± 0.0023	0.0059 ± 0.0021	0.0024 ± 0.0016
Item Average	0.0207 ± 0.0037	0.0206 ± 0.0049	0.0183 ± 0.0040	0.0082 ± 0.0031
I-NN@Cosine	0.0476 ± 0.0088	0.0468 ± 0.0103	0.0405 ± 0.0080	0.0192 ± 0.0089
U-NN@Cosine	**0.0748 ± 0.0088**	**0.0738 ± 0.0121**	**0.0622 ± 0.0100**	**0.0299 ± 0.0111**
SVD	0.0597 ± 0.0092	0.0582 ± 0.0103	0.0512 ± 0.0093	0.0240 ± 0.0097
LRec	0.0546 ± 0.0047	0.0524 ± 0.0067	0.0439 ± 0.0037	0.0217 ± 0.0074
LFlow-I	0.0602 ± 0.0111	0.0579 ± 0.0116	0.0518 ± 0.0105	0.0243 ± 0.0113
LFlow-U	0.0556 ± 0.0101	0.0535 ± 0.0109	0.0472 ± 0.0094	0.0224 ± 0.0104

The superiority of the U-NN model shows that it captures the similarity between users well. However, for the other two user-based models, LRec and LFlow-U algorithms does not perform as well as the U-NN algorithm.

Despite the success of the U-NN, the nearest neighbourhood algorithm does not work well on item-side. On the other hand, item-side Linear FLow model performs the second best among all the algorithms in Precision@10, R-Precision, and Recall@10. It shows Linear FLow algorithm has more advantage to identify the relevancy between items.

As stated before, the goal of the experiment is to build a baseline for each algorithm on pure user-photo interaction data. Later on, we will add additional techniques to these algorithms and assess if there will be any additional value to the performance.

5.3 Addition of Temporal Effect to the Collaborative Filtering Algorithms

In the subsection 2.3.2, we introduced the idea of encoding the time drifting patterns into the Collaborative Filtering model to improve its performance. In this section, we evaluated the performance of the nearest-neighbourhood models with Temporal Decay and compared with the results recorded through the last experiment in section 5.2. The goal is to find if the addition of the temporal effect will add value to our algorithms.

5.3.1 Models' Configuration

In the section 2.3.2, we explained two different approaches to define the time of our predicted action in equation 2.3, aka when the user u will like or dislike the photo i:

1. ***TD-I***: The time in the beginning of the test phase

TABLE 5.5: The optimal configuration of the time decay models after the parameter tuning

Model	Similarity Metrics	Parameter(s)
I-NN-TD-I	Cosine	Decay Parameter: 100; Similarity Threshold: 0.31
I-NN-TD-II	Cosine	Decay Parameter: 10; Similarity Threshold: 0.31
U-NN-TD-I	Cosine	Decay Parameter: 1000; Similarity Threshold: 0.06
U-NN-TD-I	Cosine	Decay Parameter: 1000; Similarity Threshold: 0.06

2. ***TD-II***: The time in the middle of the test phase

In the validation phase, we tried both the *U-NN* and *I-NN* with all the similarity metrics on all our five data samples. After parameter tuning, we get a list of models with the optimal parameters, as shown in the table 5.5.

5.3.2 Models' Performance

All of the models in Table 5.5 are evaluated and their performance are compared to the U-NN and I-NN models from the table 5.4. The results of the performance can be found in the Table 5.6.

Clearly from the table, the *U-NN-TD-I* and *U-NN-TD-II* models show the best performance in all the four evaluation metrics. It shows the addition of time decay factor **further pushes the advantage of the User-based NN model forward**, which was the best performing models among all in the last experiment. On the other hand, in photo-based Nearest Neighbourhood models, we do not see performance improvement brought by either *I-NN-TD-I* or *I-NN-TD-II*. The results indicates that in the domain of photography, **users' taste on photos does shift as time goes and thus should be taken into account when developing a recommender system**, while for photos, the similarity may not change quickly. Considering the time span of the dataset is still small (the timespan of the training data is within two weeks), further experiment on a larger time horizon may show more significant improvement in performance.

When comparing two approaches of time definition, the performance of *ID-I* and *TD-II* is almost identical in both User-based models and photo-based models. As mentioned above, as the time span of the dataset is small, it may be not sufficient to compare the these two approaches.

5.4 Embedding of Users' and Photos' Metadata

In addition to the user-photo interaction data, we also have users' and photos' metadata, as introduced in the subsection 4.1.2. In this experiment, we want to assess if the addition of these metadata will help improve the performance of our algorithms.

TABLE 5.6: Performance Metrics of Time Decay Models. The results come from the test sets from 5 data samples. The best measurement in each column is highlighted in bold. All of the NN models are using Cosine Similarity metric

Model	Precision@10	Avg Precision	R-Precision	Recall@10
I-NN	0.0476 ± 0.0088	0.0468 ± 0.0103	0.0405 ± 0.0080	0.0192 ± 0.0089
I-NN-TD-I	0.0430 ± 0.0057	0.0419 ± 0.0068	0.0357 ± 0.0046	0.0172 ± 0.0069
I-NN-TD-II	0.0429 ± 0.0056	0.0419 ± 0.0068	0.0358 ± 0.0047	0.0172 ± 0.0069
U-NN	0.0748 ± 0.0088	0.0738 ± 0.0121	0.0622 ± 0.0100	0.0299 ± 0.0111
U-NN-TD-I	**0.0750 ± 0.0088**	**0.0744 ± 0.0121**	**0.0627 ± 0.0100**	**0.0299 ± 0.0109**
U-NN-TD-II	**0.0750 ± 0.0088**	**0.0745 ± 0.0120**	**0.0627 ± 0.0100**	**0.0299 ± 0.0110**

A detailed list of metadata can be found in the highlighted rows in the table 4.1 and 4.2. We are using the same algorithms evaluated in the section 5.2, but at this time, we will train them on both the user-photo interaction data as well as the metadata. We will evaluate their performance and compare them to the performance in the table 5.4.

Please note that there are two methods to embedding the metadata to the Nearest Neighbourhood Model. Take the item-based Nearest Neighbourhood Model as an example:

1. *Weighted Matrices*, which concatenate matrices and gives different weights to each part. Then the similarity metric will be applied to the concatenated matrix to compute the similarity.
2. *Weighted Similarity*, which compute the similarity of the user-item matrix and that of the metadata separately, and then compute the weighted sum of two similarity matrices.

In this section, we will evaluate both of the embedding methods.

5.4.1 Models' Configuration

After tuning the parameters with the validation sets, we get the optimal configuration of all the models, as shown in the Table 5.7.

Please note that for U-NN model, through our parameter tuning process, we find the optimal weight on users' metadata is 0, so it means that the addition of the users' metadata will not help with the model. Therefore, we excluded it from the evaluation.

TABLE 5.7: The optimal configuration of the models trained on both User-Photo Data and Metadata

Model	Parameter
I-NN-Meta I@Cosine	Similarity Threshold: 0.31; Weight on Metadata: 0.1
I-NN-Meta II@Cosine	Similarity Threshold: 0.31; Weight on Metadata: 0.2
SVD-I-Meta	k factor: 0.09
SVD-U-Meta	k factor: 0.15
LFlow-I	k factor: 0.53; Parameter of Regularization: 75
LFlow-U	k factor: 0.45; Parameter of Regularization: 150

TABLE 5.8: Performance of models trained on both user-photo interaction data and metadata (noted with 'meta' in model name) compared with the models trained on user-photo interaction alone. The results come from the test sets from 5 data samples. The best measurement in each column is highlighted in bold.

Model	Precision@10	Avg Precision	R-Precision	Recall@10
I-NN@Cosine	0.0476 ± 0.0088	0.0468 ± 0.0103	0.0405 ± 0.0080	0.0192 ± 0.0089
I-NN-Meta I@Cosine	0.0437 ± 0.0053	0.0425 ± 0.0064	0.0367 ± 0.0044	0.0175 ± 0.0068
I-NN-Meta II@Cosine	0.0427 ± 0.0056	0.0418 ± 0.0066	0.0357 ± 0.0045	0.0171 ± 0.0068
SVD	0.0597 ± 0.0092	0.0582 ± 0.0103	0.0512 ± 0.0093	0.0240 ± 0.0097
SVD-I-Meta	0.0612 ± 0.0088	0.0591 ± 0.0108	0.0522 ± 0.0094	0.0245 ± 0.0097
SVD-U-Meta	0.0537 ± 0.0137	0.0518 ± 0.0147	0.0453 ± 0.0133	0.0219 ± 0.0120
LFlow-I	0.0602 ± 0.0111	0.0579 ± 0.0116	0.0518 ± 0.0105	0.0243 ± 0.0113
LFlow-I-Meta	$\mathbf{0.0643 \pm 0.0121}$	$\mathbf{0.0619 \pm 0.0130}$	$\mathbf{0.0561 \pm 0.0115}$	$\mathbf{0.0260 \pm 0.0122}$

5.4.2 Models' Performance

All of the models in Table 5.7 are trained with both user-photo interaction data and users' or photos' metadata and evaluated with the test sets. The results of the performance can be found in the Table **??**.

First, we can see that the SVD model and LFlow-I model get improved with the additional photos' metadata. Their performance measures are the best in the table, especially the LFlow-I model. Both of the models are learning the photos' information in latent space, which is more convenient as we do not need to tune the weight on the metadata. The results also show the two methods based latent factorization can embed the photos' attributes better.

In comparison, I-NN model does not see any improvement from either of the embedding methods. Its performance gets even worse with the additional metadata. It shows the Nearest Neighbour is not very capable to extract insights from the photos' metadata, including their categories and tags.

TABLE 5.9: The configuration of the models trained on both User-Photo Data and visual embedded data

Model	Parameter
HSV-NN	Similarity Threshold: 0.31; Weight on Metadata: 0.01
HSV-LRec	Parameter of Regularization: 0.9
HSV-LFlow-I	k factor: 0.33; Parameter of Regularization: 75
Style-NN	Similarity Threshold: 0.31; Weight on Metadata: 0.3

5.5 Embedding of Visual Features

In this final experiment, we will evaluate the performance of the novel algorithm we proposed in chapter 3 to embed visual features, including HSV color space and style information of photos. We will first describe the configuration the HSV and Style-embedded models, and then show the performance results.

5.5.1 Models' Configuration

The configuration of the models we evaluated is listed in the table 5.9.

Please note that, for each photo, the length of the style vector, which is 262,144, is much longer than the users' preference vector. Therefore, in practice, the only algorithm we can use is the item-based nearest neighbourhood, by computing the photo-photo similarity on user-photo interaction data and style matrix separately, and compute the weighted sum of two similarity matrix. For the other algorithms, like SVD and Linear FLow, the impact of user-photo interaction will be very trivial with the dominance of the style matrix. In our validation process, we also find the results from SVD and Linear FLow algorithms were very poor. Therefore, we only used the item-based NN algorithm to evaluate the value of style feature.

5.5.2 Models' Performance

All of the models listed in the table 5.9 are evaluated together. The results of the performance can be found in the Table **??**.

First let's look at the HSV-embedded models. If we compare them with baseline models trained on pure user-photo interaction data, including *I-NN-Meta-I*, *LRec*, and *LFLow-I* in table 5.4, we don't see the HSV-embedded models show any advantages. It shows the embedded HSV color space does not help with the photo recommendation.

On the other hand, the *Style-NN* model is the best-performing model across all the metrics in the Table 5.10. Furthermore, if we compare it with the *I-NN* in table 5.4 and two *I-NN* models with metadata in 5.8, it clearly shows a large improvement in performance: it is 24% better than the *I-NN* model and 2.84 times better than the

TABLE 5.10: Performance of Visual-Feature-Embedded Models. The results come from the test sets from 5 data samples. The best measurement in each column is highlighted in bold.

Model	Precision@10	Avg Precision	R-Precision	Recall@10
HSV-LRec	0.0520 ± 0.0041	0.0503 ± 0.0057	0.0427 ± 0.0041	0.0207 ± 0.0069
HSV-LFlow	0.0549 ± 0.0094	0.0527 ± 0.0105	0.0463 ± 0.0089	0.0221 ± 0.0100
HSV-SVD	0.0381 ± 0.0041	0.0366 ± 0.0021	0.0300 ± 0.0023	0.0148 ± 0.0024
HSV-NN	0.0426 ± 0.0056	0.0414 ± 0.0064	0.0355 ± 0.0044	0.0171 ± 0.0068
Style-NN	$\mathbf{0.0588 \pm 0.0122}$	$\mathbf{0.0566 \pm 0.0138}$	$\mathbf{0.0498 \pm 0.0108}$	$\mathbf{0.0238 \pm 0.0116}$

popularity baseline algorithm in precision@10. This is very promising result. It supports our hypothesis that the addition of the style information can help us discover the photos that are similar to each other from humans' perception, even when the metadata is not helping with the algorithm, as shown in the table 5.4. Therefore, it is really valuable in the context of the photo recommendation.

Chapter 6

Conclusion

6.1 Summary of Contributions

In this thesis we evaluated the performance of multiple existing recommender system algorithms in the context of photo recommendation and proposed our novel algorithm that improves the performance. Through the literature review we found there has been very few research work on photo recommendation, and most of them are relying on the textual metadata of photos to improve the recommendation. In real life, we have more features from the photos themselves. We proposed the novel algorithms to embed visual attributes from photos to improve the performance of the recommender system. By partnering with 500px, we use their user-photo interaction dataset to evaluate our algorithms.

Through experiment, we proved that our proposed style feature embedding method on item-based nearest neighbourhood algorithm shows great improvement on existing models. It shows 24% increase than the *I-NN* model and 2.84 times better than our *popularity* baseline algorithm in our precision@10 metric. Despite the fact that the embedding of photos' metadata does not improve the I-NN model, style feature embedding method still makes a large improvement over the I-NN model. It means that our style embedding method successfully capture human perceived similarity between photos. Therefore, it is a valuable technique to help solving the photo recommendation problem.

We also evaluated the existing one-class recommender system algorithms in the same dataset. We found that the User-based nearest neighbourhood (U-NN) algorithm performs better than other existing algorithms in this dataset. With cosine similarity metric, this algorithm successfully captured the similarity among users' taste in order to provide effective photo recommendation. Beyond that, through the technique of embedding time decay effect into the models, we successfully improve the performance of U-NN even further. It shows the users' taste does shift across time, and thus should be captured by the recommender system models.

Lastly, through our third experiment on users' and items' metadata, we find users' metadata does not help in improving the U-NN model. In order to pushing its performance forward, we need to explore more meaningful and indicative features on users. On the other hand, items' metadata helps improves the performance of

two algorithms that are based on latent factorization, the SVD matrix factorization algorithm and item-based Linear FLow algorithm.

6.2 Directions for Future Work

The way we quantified style feature of photos in this thesis is using the style transfer technique on the VGG-19 Convolutional Neural Network. Future work should explore the use of other computer vision models and compare the performance in style embedding. It is especially valuable to find new way to quantify style features into smaller vectors, so that the computation can be more scalable.

The ultimate goal of this photo recommendation work is not only to improve the effectiveness of predicting the photos that you may like, but more broadly, to discover high performing image recommendation technique and apply it in product recommendation problem. Potential applications includes recommendation of products based on the outlook and personalized display of product advertisement. Future researchers can extend this work to develop systems that use more visual features in personalized recommendation.

Bibliography

Ashok Chandrashekar Fernando Amat, Justin Basilico and Tony Jebara (2018). *Artwork Personalization at Netflix*. URL: https://medium.com/netflix-techblog/artwork-personalization-c589f074ad76.

Bell, Robert M and Yehuda Koren (2007). "Improved Neighborhood-based Collaborative Filtering". In: URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.129.4662&rep=rep1&type=pdf.

Bennett, James and Stan Lanning (2007). "The Netflix Prize". In: URL: https://www.cs.uic.edu/~liub/KDD-cup-2007/NetflixPrize-description.pdf.

Burke, Robin (2002). "Hybrid Recommender Systems: Survey and Experiments". In: *User Modeling and User-Adapted Interaction* 12.4, pp. 331–370. ISSN: 09241868. DOI: 10.1023/A:1021240730564. URL: http://link.springer.com/10.1023/A:1021240730564.

Carneiro, Gustavo et al. (2007). "Supervised Learning of Semantic Classes for Image Annotation and Retrieval". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.3, pp. 394–410. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2007.61. URL: http://ieeexplore.ieee.org/document/4069257/.

Cawley, Gavin C. and Nicola L. C. Talbot (2010). "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation". In: *Journal of Machine Learning Research* 11.Jul, pp. 2079–2107. ISSN: ISSN 1533-7928. URL: http://www.jmlr.org/papers/v11/cawley10a.html.

Charu C. Aggarwal (2016). *Recommender Systems: the Textbook*.

Cheng, H.D. et al. (2001). "Color image segmentation: advances and prospects". In: *Pattern Recognition* 34.12, pp. 2259–2281. ISSN: 0031-3203. DOI: 10.1016/S0031-3203(00)00149-7. URL: https://www.sciencedirect.com/science/article/pii/S0031320300001497?via%3Dihub#BIB30.

Deng, Li and Dong Yu (2014). "Deep Learning: Methods and Applications". In: *Foundations and Trends® in Signal Processing* 7.3-4, pp. 197–387. ISSN: 1932-8346. DOI: 10.1561/2000000039. URL: http://nowpublishers.com/articles/foundations-and-trends-in-signal-processing/SIG-039.

Ekstrand, Michael D., John T. Riedl, and Joseph A. Konstan (2011). "Collaborative Filtering Recommender Systems". In: *Foundations and Trends® in Human–Computer Interaction* 4.2, pp. 81–173. ISSN: 1551-3955. DOI: 10.1561/1100000009. URL: http://www.nowpublishers.com/article/Details/HCI-009.

Fawcett, Tom., Nina. Mishra, and D.C.) International Conference on Knowledge Discovery & Data Mining (2003 : Washington (2003). "Weighted low-rank approximations". In: *Weighted low-rank approximations*. AAAI Press, p. 940. ISBN: 1577351894. URL: https://dl.acm.org/citation.cfm?id=3041929.

Gatys, Leon A., Alexander S. Ecker, and Matthias Bethge (2015a). "A Neural Algorithm of Artistic Style". In: URL: http://arxiv.org/abs/1508.06576.

— (2015b). "A Neural Algorithm of Artistic Style". In: URL: http://arxiv.org/abs/1508.06576.

Goldberg, David et al. (1992). "Using collaborative filtering to weave an information tapestry". In: *Communications of the ACM* 35.12, pp. 61–70. ISSN: 00010782. DOI: 10.1145/138859.138867. URL: http://portal.acm.org/citation.cfm?doid=138859.138867.

Herlocker, Jonathan L. et al. (1999). "An algorithmic framework for performing collaborative filtering". In: *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '99*. New York, New York, USA: ACM Press, pp. 230–237. ISBN: 1581130961. DOI: 10.1145/312624.312682. URL: http://portal.acm.org/citation.cfm?doid=312624.312682.

Herlocker, Jonathan L. et al. (2004). "Evaluating collaborative filtering recommender systems". In: *ACM Transactions on Information Systems* 22.1, pp. 5–53. ISSN: 10468188. DOI: 10.1145/963770.963772. URL: http://portal.acm.org/citation.cfm?doid=963770.963772.

Hu, Diane J., Rob Hall, and Josh Attenberg (2014). "Style in the long tail". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '14*. New York, New York, USA: ACM Press, pp. 1640–1649. ISBN: 9781450329569. DOI: 10.1145/2623330.2623338. URL: http://dl.acm.org/citation.cfm?doid=2623330.2623338.

Jagadeesh, Vignesh et al. (2014). "Large Scale Visual Recommendations From Street Fashion Images". In: URL: http://arxiv.org/abs/1401.1778.

Karpathy, Andrej and Li Fei-Fei (2017). "Deep Visual-Semantic Alignments for Generating Image Descriptions". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.4, pp. 664–676. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2016.2598339. URL: http://ieeexplore.ieee.org/document/7534740/.

Kiapour, M. Hadi et al. (2014). "Hipster Wars: Discovering Elements of Fashion Styles". In: Springer Verlag, pp. 472–488. DOI: 10.1007/978-3-319-10590-1{_}31. URL: http://link.springer.com/10.1007/978-3-319-10590-1_31.

Kong, Weihao, Wu-Jun Li, and Minyi Guo (2012). "Manhattan hashing for large-scale image retrieval". In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '12*. New York, New York, USA: ACM Press, p. 45. ISBN: 9781450314725. DOI: 10.1145/2348283.2348293. URL: http://dl.acm.org/citation.cfm?doid=2348283.2348293.

Koren, Yehuda, Robert Bell, and Chris Volinsky (2009). "Matrix Factorization Techniques for Recommender Systems". In: *Computer* 42.8, pp. 30–37. ISSN: 0018-9162. DOI: 10.1109/MC.2009.263. URL: http://ieeexplore.ieee.org/document/5197422/.

Koren, Yehuda and Yehuda (2009). "Collaborative filtering with temporal dynamics". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '09*. New York, New York, USA: ACM Press, p. 447. ISBN: 9781605584959. DOI: 10.1145/1557019.1557072. URL: http://portal.acm.org/citation.cfm?doid=1557019.1557072.

— (2010). "Factor in the neighbors". In: *ACM Transactions on Knowledge Discovery from Data* 4.1, pp. 1–24. ISSN: 15564681. DOI: 10.1145/1644873.1644874. URL: http://portal.acm.org/citation.cfm?doid=1644873.1644874.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton (2015). "Deep learning". In: *Nature* 521.7553, pp. 436–444. ISSN: 0028-0836. DOI: 10.1038/nature14539. URL: http://www.nature.com/articles/nature14539.

Linden, G., B. Smith, and J. York (2003). "Amazon.com recommendations: item-to-item collaborative filtering". In: *IEEE Internet Computing* 7.1, pp. 76–80. ISSN:

1089-7801. DOI: 10.1109/MIC.2003.1167344. URL: http://ieeexplore.ieee.org/document/1167344/.

Liu, Si et al. (2012). "Hi, magic closet, tell me what to wear!" In: *Proceedings of the 20th ACM international conference on Multimedia - MM '12*. New York, New York, USA: ACM Press, p. 619. ISBN: 9781450310895. DOI: 10.1145/2393347.2393433. URL: http://dl.acm.org/citation.cfm?doid=2393347.2393433.

Makadia, Ameesh, Vladimir Pavlovic, and Sanjiv Kumar (2008). "A New Baseline for Image Annotation". In: Springer, Berlin, Heidelberg, pp. 316–329. DOI: 10.1007/978-3-540-88690-7{_}24. URL: http://link.springer.com/10.1007/978-3-540-88690-7_24.

Pazzani, Michael J. and Daniel Billsus (2007). "Content-Based Recommendation Systems". In: *The Adaptive Web*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 325–341. DOI: 10.1007/978-3-540-72079-9{_}10. URL: http://link.springer.com/10.1007/978-3-540-72079-9_10.

Reed, Scott et al. (2016). "Generative Adversarial Text to Image Synthesis". In: URL: http://arxiv.org/abs/1605.05396.

Resnick, Paul and Hal R. Varian (1997). "Recommender systems". In: *Communications of the ACM* 40.3, pp. 56–58. ISSN: 00010782. DOI: 10.1145/245108.245121. URL: http://portal.acm.org/citation.cfm?doid=245108.245121.

Rusian Salakhutdinov, Andriy Mnih (2009). "Probabilistic Matrix Factorization". In: *Probabilistic Matrix Factorization*. Neural Information Processing Systems Foundation, pp. 1257–1264. ISBN: 9781605603520. URL: https://dl.acm.org/citation.cfm?id=2981720.

Sarwar, Badrul et al. (2001). "Item-based collaborative filtering recommendation algorithms". In: *Proceedings of the tenth international conference on World Wide Web - WWW '01*. New York, New York, USA: ACM Press, pp. 285–295. ISBN: 1581133480. DOI: 10.1145/371920.372071. URL: http://portal.acm.org/citation.cfm?doid=371920.372071.

Sedhain, Suvash (2016). "A Unified Approach to Collaborative Filtering via Linear Models and Beyond". PhD thesis. Australian National University. URL: https://openresearch-repository.anu.edu.au/handle/1885/118270.

Sedhain, Suvash et al. (2016a). "On the Effectiveness of Linear Models for One-Class Collaborative Filtering". In: *Proceedings of the 30th Conference on Artificial Intelligence (AAAI 2016)*.

Sedhain, Suvash et al. (2016b). "Practical linear models for large-scale one-class collaborative filtering". In: *IJCAI International Joint Conference on Artificial Intelligence*.

Shankar, Devashish et al. (2017). "Deep Learning based Large Scale Visual Recommendation and Search for E-Commerce". In: URL: http://arxiv.org/abs/1703.02344.

Stern, David H., Ralf Herbrich, and Thore Graepel (2009). "Matchbox". In: *Proceedings of the 18th international conference on World wide web - WWW '09*. New York, New York, USA: ACM Press, p. 111. ISBN: 9781605584874. DOI: 10.1145/1526709.1526725. URL: http://portal.acm.org/citation.cfm?doid=1526709.1526725.

Sural, S., Gang Qian, and S. Pramanik (2002). "Segmentation and histogram generation using the HSV color space for image retrieval". In: *Segmentation and histogram generation using the HSV color space for image retrieval*. Vol. 2. IEEE, pp. II–589–II–592. ISBN: 0-7803-7622-6. DOI: 10.1109/ICIP.2002.1040019. URL: http://ieeexplore.ieee.org/document/1040019/.

Tess Flanders (1911). *Speakers Give Sound Advice*.

Vargas, Saúl and Pablo Castells (2011). "Rank and relevance in novelty and diversity metrics for recommender systems". In: *Proceedings of the fifth ACM conference on*

Recommender systems - RecSys '11. New York, New York, USA: ACM Press, p. 109. ISBN: 9781450306836. DOI: 10.1145/2043932.2043955. URL: http://dl.acm.org/citation.cfm?doid=2043932.2043955.

Xia, Hao et al. (2012). "Boosting multi-kernel locality-sensitive hashing for scalable image retrieval". In: *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '12*. New York, New York, USA: ACM Press, p. 55. ISBN: 9781450314725. DOI: 10.1145/2348283.2348294. URL: http://dl.acm.org/citation.cfm?doid=2348283.2348294.

Xin Jin et al. (2013). "Reinforced Similarity Integration in Image-Rich Information Networks". In: *IEEE Transactions on Knowledge and Data Engineering* 25.2, pp. 448–460. ISSN: 1041-4347. DOI: 10.1109/TKDE.2011.228. URL: http://ieeexplore.ieee.org/document/6081863/.

Y. Kalantidis and L. Kennedy and L.-J. Li (2013). "Getting the Look: Clothing Recognition and Segmentation for Automatic Product Suggestions in Everyday Photos". In: *in Proceedings of International Conference on Multimedia Retrieval (ICMR) (ICMR 2013)*. Dallas, TX: ACM.

Yamaguchi, Kota, M. Hadi Kiapour, and Tamara L. Berg (2013). "Paper Doll Parsing: Retrieving Similar Styles to Parse Clothing Items". In: *2013 IEEE International Conference on Computer Vision*. IEEE, pp. 3519–3526. ISBN: 978-1-4799-2840-8. DOI: 10.1109/ICCV.2013.437. URL: http://ieeexplore.ieee.org/document/6751549/.

Zhang, Yongzheng and Marco Pennacchiotti (2013). "Predicting purchase behaviors from social media". In: *Proceedings of the 22nd international conference on World Wide Web - WWW '13*. New York, New York, USA: ACM Press, pp. 1521–1532. ISBN: 9781450320351. DOI: 10.1145/2488388.2488521. URL: http://dl.acm.org/citation.cfm?doid=2488388.2488521.