Review of Methods to Build Personalized Recommendation Systems Based on User Portraits

Zujie Shao

Ulink College of Shanghai, 559 South Laiting Road, Jiuting Town, Songjiang District, Shanghai, 201615, China
Corresponding Author: Zujie Shao, Email: 1543224485@qq.com

Abstract

The advancement in Internet and communication technology increases the amount of information. To meet the rapid growth of the demand to extract valuable information from such backgrounds, the utilization of a personalized recommendation system based on user portraits becomes increasingly popular. The number of products in this domain has increased fast in recent years. Therefore, a comprehensive review and summary of both previous and current works are needed to give a better understanding of state-of-the-art approaches to building the systems with potential challenges and promising directions in the future. In this paper, I will discuss some commonly used approaches, such as long short-term memory, convolutional neural network, support vector machine and random forest. I will discuss these approaches through several aspects from the basic concept to general architecture to implement the systems to their advantages and challenges. Finally, I will sum up the potential problems of the systems and give a possible direction of a solution to the problems.

Keywords

User portrait; Recommendation system; Convolutional neural networks (CNN); Long short-term memory (LSTM); Support vector machine (SVM); Random forest

Introduction

The advances in Internet technology facilitate the generation, storage and processing of information. The volume, diversity and complexity of information increase fast to satisfy the strong demands of users for knowledge. This brings opportunities and challenges for users to extract valuable information. When information comes is too dense in forms of complexity and amount, it starts to act like noise when reaching overload. Thus, information overload could be a challenge for service providers. For example, in an internet shopping environment, information overload may lead consumers to a worse subject state (Chen, Shang, & Kao, 2009). Moreover, users may fail to process information efficiently without distraction, and anxiety with high rate of receiving information (Edmunds & Morris, 2000). One widely used method to solve this problem is to use search engines. Due to the

Citation: Zujie Shao. (2022) Review of Methods to Build Personalized Recommendation Systems Based on User Portraits. The Journal of Young Researchers 4(19): e20220630

Received on May 7, 2022; Accepted on May 10, 2022; Published on June 30, 2022

nature of the limited ability to store and process information, existing search engines (e.g. Google) extract relevant information online based on keywords input by users are not sufficient enough to meet users' demands (Alaoui, Idrissi, & Ajhoun, 2015). As current search engines give access to countless websites and infinite information which allow users to browse them for eternity without strong query formulation, it takes a lot of time and effort for users to distinguish relevant information from irrelevant ones. Moreover, responses given by search engines are monotonous while users are complicated in form of backgrounds such as genders, locations, careers, etc. In other words, search engines answer queries mainly depending on content without taking into consideration of individual differences. For instance, the same queries made by two different users may produce similar results despite their past behaviours being different.

To take advantage of the characteristics of different users and output personalized information, systems that can deliver information based on users' previous behaviours and users' interests are developed. Such systems that can collect and analyse users' characteristic features and activities to personalize the outcome of recommendations are referred to as personalized recommendation systems. For example, the majority of videos recommended by YouTube are based on a personalized recommendation system considering the activities of users (Davidson et al., 2010).

One common way to build such a system is by using user portraits. Here user portrait is a concept that tries to label users with characteristic features from attributes, interests and previous behaviours. Those labels may include online social behaviours, history of browsing, ages, genders, etc. User portrait allows service providers to analyse their users better to supply services of a higher quality. It also enables users to have an objective image of themselves. User portraits have been used in various domains such as e-commerce, social media, personality analysis and the medical field. Previous research showed that user portraits can be used in monitoring and recognizing customers that are likely to give a negative opinion on items. This was implemented by analysing users' past activities and comments (T. Chen et al., 2021). Social media companies can predict the topics that a user will in future by collecting users' posting history and participation history (Huang, Zhang, Wu, & Huang, 2017) Similarly, user portraits can also cluster users based on their history of communication via the social media (Gu, Wang, Wang, Zhuang, & Su, 2018; Gu, Wang, Wang, Zhuang, Bian, & Su, 2018). Moreover, user portraits are of vital importance in the healthcare domain. Studies have been conducted using medical big data and user portraits to accurately recommend medical specialists online (Fedushko, Shakhovska, & Syerov, 2018).

With the prosperity of machine learning in recent years, many researchers choose to use these techniques to implement the concept of user portraits on the recommendation system. Compared with traditional statistical methods, machine learning methods are more flexible and require fewer assumptions on the user behaviours. According to previous studies (P. Zhang et al., 2022), Bidirectional Long Short-Term Memory (BiLSTM) could be used in order to account for contextual information from both forward and backward propagation (P. Zhang et al., 2022). In addition, a combined method through LSTM and Convolutional Neural Network (CNN) was introduced to remove local features before mining the relationship between terms and use a Support Vector Machine (SVM) for the usage of classification to separate users' attributes like age, education and gender from raw data (Y. Chen, He, Wei, Zhu, & Yu, 2021). A three-way recommendation system was developed based on the random forest to avoid the problem of overfitting (Zhang & Min, 2016). A recommendation system that performed recommendations by exploring the association between user portraits and commodity portraits using Association Rule Mining (ARM) was proposed (Wu, Yang, Zhang, Zhu, & Wan, 2020). In general, applying machine learning to implement a recommendation system based on user portraits has many advantages. First, with a great amount of information available, collecting data is much easier. It allows us to train complex models with many parameters. Second, models trained by machine learning are flexible to extract the patterns underlying the data.

With the great achievements of personalized recommendation systems based on user portraits, the number of works in this domain has been increasing dramatically over these years. Therefore, a systematic and comprehensive review and summary of influential works are needed to give a better understanding of state-of-the-art user portraits with potential challenges we may meet and promising directions in this area. Alternatively, amounts of secondary studies, which focus on works created by other researchers and analyse existing issues that are possible to be solved, are in demand.

I will introduce and compare commonly used machine learning techniques, including LSTM, CNN, SVM and random forest. I will give introductions on the implementation of each method on user portraits. I will compare the strengths and insufficient parts of each technique. Finally, I will describe open challenges and possible future solutions in the area of user portraits. I will provide a comprehensive review of the state-of-the-art machine learning techniques that were used to construct personalized recommendation systems based on user portraits. This will serve as guidance for new researchers who want to begin their study but without a clear understanding of this field.

Long Short-Term Memory

LSTM is a machine learning technique developed from the recurrent neural network (RNN). As the LSTM method perfectly inherits the advantages of RNN and improves the limitation of normal RNN, the architecture of the LSTM method is clearer if we start from a classical RNN structure to demonstrate the architecture of the LSTM method.

Recurrent Neural Network

RNNs (Rumelhart, Hinton, & Williams, 1986) are a family of neural networks which makes operation based on units of a neuron. This idea including name and structure was initially inspired by the human brain. RNNs are specialized for processing sequential data, a series of elements, like a series x1, x2. . . xn. The typical feature of the RNN architecture is the cyclic connection between and in each step Figure 1. This allows RNNs to update the current state. Here the state is a storage to store values calculated by current input elements combined with selective information from past elements. The calculated output of the current state can then be used as one part of the input as the past state in the next state. This recursive algorithm will be carried out through each cell. In this way, RNNs can store information about the history of all the past elements in a hidden layer by processing an input sequence one element per time. Thus, RNNs are able to capture latent relations in sequential data. This architecture can be illustrated in Figure 1.

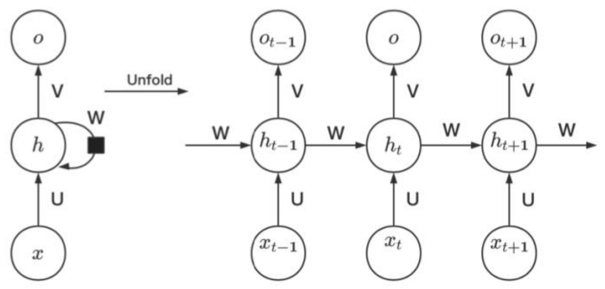

Figure 1. A classical RNN architecture

At each time t, the input is x_t, the hidden layer activation is h_t and the output is o_t. The W is the parameter of hidden to hidden connections. The *U* is the parameter of input to hidden connections. *V* is the parameter of hidden output connections. Each cell shares the same parameters (*U, V, W*). 'Unfold' here means that a single cell can be connected and expanded on the right in Figure 1. By this architecture, it can choose to put any information from the past into a hidden layer h and transmit h to the future. Figure 1 does not state the choice of activation function for the hidden layer. Typically the hyperbolic tangent is applied for the hidden layer. We assume the forward propagation of the RNN begins with an initial value. Thus the forward propagation equations for this RNN can be updated in Equation 1.

$$\begin{aligned} a^{(t)} &= b + Wh^{(t-1)} + Ux^{(t)} \\ h^{(t)} &= \tanh\left(a^{(t)}\right) \qquad (1) \\ o^{(t)} &= c + Vh^{(t)} \end{aligned}$$

where *b* and *c* are the bias vectors used with the weight *W*, *U* and *V* in input to hidden connections, hidden to hidden connections and hidden to output connections respectively. RNNs are very powerful dynamic systems and can be trained by the backward propagation method. This method uses the chain rule to firstly calculate the derivative of output as an input to calculate the derivative of the prior step through the whole process. The common challenge of RNNs comes when learning long-term dependencies. Assume the self-multiplication of a scalar weight *W* by many times. The result, W^t, will either tend to zero or infinite through the multiplication of many times depending on the initial value of *W* chosen. If initial *W* is larger than 1, the result value will explode and if initial *W* is smaller than 1, the result value will vanish. The fundamental reason for the vanishing and exploding gradient problem was firstly analysed by researchers (Bengio, Simard, & Frasconi, 1994; Hochreiter, 1991). The simple method is to keep parameters in a suitable space so that neither vanishing nor exploding happens in place when propagating forward. This can be a possible solution but not for a general situation. In order to build a robust network, which can store enough past information to be resistant to small perturbations, the problem of gradient vanishing is required to be solved. Besides, whenever it can carry out learning of long-term dependencies, the magnitude of long-term interaction is getting rapidly smaller than short-term interaction. This means the long-term interaction may tend to be hidden by small fluctuations from short-term interaction (Bengio et al., 1994).

LSTM Architecture

To solve the problem of vanishing gradient, a self-loop in an LSTM model was introduced (Hochreiter & Schmidhuber, 1997). Instead of a unit cell that basically applies non-linear operation in the transformation of inputs and recurrent units, LSTM recurrent networks have "LSTM cells" with an internal recursive relation (self-loop) combining with the RNN's outer recurrence. By using another separate hidden layer, the weight of this self-loop can change.

Thus, the gradient can restrain at a reasonable value. Each 'LSTM' cell has a similar transformation on input as a traditional recurrent network, but it contains extra parameters and a set of gating units to govern the flow of information. Thus, the LSTM model is capable to capture long-term dependencies over a long period. Figure 2 (Yu, Si, Hu, & Zhang, 2019) depicts the inner connections of an 'LSTM' cell, which may be used to extract the mathematical formulas:

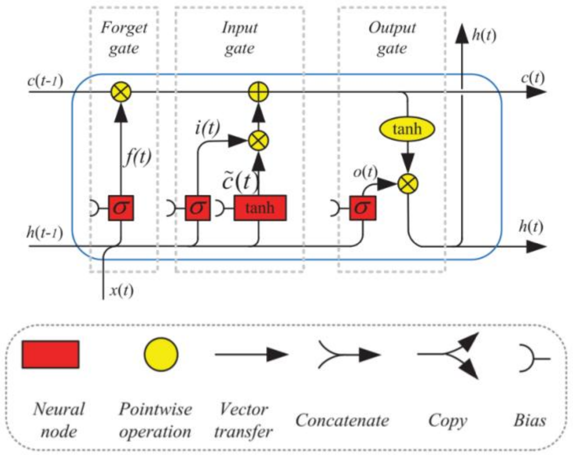

Figure 2. Architecture of an LSTM cell
Source: This graph is from '*A review of recurrent neural networks LSTM cells and network architectures*'

The LSTM cell can be mathematically described using the connections illustrated in Figure 2 (Yu et al., 2019) below:

$$f_t = \sigma\left(W_{fh}h_{t-1} + U_{fx}x_t + b_f\right)$$

$$i_t = \sigma\left(W_{ih}h_{t-1} + U_{ix}x_t + b_i\right)$$

$$\tilde{c}_t = \tanh\left(W_{\tilde{c}h}h_{t-1} + U_{\tilde{c}x}x_t + b_{\tilde{c}}\right)$$

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t$$

$$o_t = \sigma\left(W_{oh}h_{t-1} + U_{ox}x_t + b_o\right)$$

$$h_t = o_t \cdot \tanh\left(c_t\right)$$

Where x_t, h_t denotes the current input vector, and the current hidden layer vector of the cell respectively. σ denotes the activation function sigmoid used here. Each gate has separate b. $U_{\cdot x}$, $W_{\cdot h}$, for bias and weights of input and hidden layer vectors respectively. The most important component in the LSTM model is cell the state c_t. This is where a self-linear loop is added and can be used to update information in a cell combining the information from the hidden layer vector and external input. In other words, c_t. is determined by both forget gate and the external input gate. The forget gate f_t can determine what information from the cell state will be discarded. When the forget gate f_t turns to 1, it entirely preserves this information; otherwise, a value of 0 implies it throws away all of it. i_t denotes the external input gate which is calculated in a similar way as the forget gate. This gate decides what information from the input can be stored in the cell. o_t denotes the output gate deciding what information from the current cell state can be output. An RNN can be built by joining several LSTM cells together. To mine deeper contextual relations, extra hidden layers can also be used. For example, users' history of online communication is used to mine user tags to further predict the user's behaviours in the recommendation system.

Applying LSTM to Build a Recommendation System Based on the User Portraits

LSTM is commonly used in text processing. For example, the paper (Y. Chen et al., 2021) designed an LSTM algorithm used to predict user tags by inputting feature matrices extracted from the CNN model. The feature matrices are coming from user query terms in a search engine. User tags include age, gender and education. Here, LSTM is used to discover latent relations in contextual information and user tags. As input data is a set of short terms from a user, this is sequential data which an LSTM model is suitable to use on it. This paper does not include the classical LSTM model, it uses the gated recurrent unit (GRU) model, a variant of the LSTM model instead. One apparent difference between a classical LSTM model and a GRU model is GRU uses a single gate to work as the forget gate and make decisions to update the hidden layer vector in LSTM simultaneously. The formula in the paper can be expressed mathematically in Equation 2:

$$\begin{aligned} z_t &= \sigma\left(W_z \cdot [h_{t-1}, x_t]\right) \\ r_t &= \sigma\left(W_r \cdot [h_{t-1}, x_t]\right) \\ \tilde{h}_t &= \tanh\left(W \cdot [r_t \cdot h_{t-1}, x_t]\right) \\ h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \hat{h}_t \end{aligned} \quad (2)$$

The feature vector x_t and the hidden layer vector h_{t-1} comprise the input of GRU at time *t*. The output h_t is then used as input of the full link layer to reduce the number of dimensions. Finally, the softmax activation function is used to obtain the probability of each user tag that may be suitable for a specific user.

Moreover, some of tags may not only relate to the past performance of users, but also possibly be influenced by performance in the future. A Bi-LSTM architecture can be used to solve such a relation between tags and performance. The Bi-LSTM architecture is a combination of two simple LSTM architectures. One is forward propagation and another is backward propagation. In this way, the system is capable to generate user tags both from previous performance and later performance of users.

Advantage and Challenges

LSTM makes it possible for us to extend deep learning to sequential data. This allows systems to analyse contextual information from users' past behaviours and discover term dependencies, especially long-term dependencies. Compare with CNN, which does computations based on the prior layer, LSTM can build a model which can connect all cells even two not adjacent cells. Besides, LSTM can handle unknown length of sequences due to the sharing weights through each 'LSTM' cell.

Due to the architecture of LSTM, a large number of parameters needed to set up to build a powerful model, like numbers of hidden layers and cells and the initial value of weights. Because of backward propagation used in training, we cannot obtain the error in a certain mid-cell during training, the output value is hard to explain. This will influence the reliability of the model. As increasing complexity of the model, the training will take more time and with increase of one hidden layer, the cost of computation will dramatically increase. Since commonly used LSTM architectures cannot do the parallel computation, this will increase its computation cost in a further step.

Convolutional Neural Network

CNNs (LeCun et al., 1989) are a specialized type of neural network for processing data that can be arranged in grids. For example, a sentence can be seen as a 1-D grid composed of every single word; a image data can be expressed into a 2-D grid that each unit in the grid represents a specific colour. The colours are made up of three pixels: red, green, and yellow. This indicates multiple layers of grid can be used as input of data in CNNs. The name 'convolutional neural network' explicitly shows that the network applies a mathematical operation 'convolution' during the processing of data. CNNs are powerful to be used for feature extraction. In order to apply CNNs in the recommendation system, I am going to talk about the structure of CNNs

Basic CNN Components

A classical CNN is mainly made of three layers: convolutional layer, pooling layer and fully-connected layer. It uses data in forms of grids as input and extract features from the input by passing data through convolutional layers and pooling layers. Fully-connected layer is used to complete tasks based on these features, involving image recognition (Szegedy et al., 2015), video analysis (Karpathy et al., 2014) and etc.

Convolutional Layer

The convolutional layer is the core of a CNN that carry out most of computation. The parameters in this layer are mainly made up of a set of filters which can do self-learning by backward propagation method. Each of these filters is spatially small in size compared to the volume of input (along height and width) but spread along the whole depth of input. Filters will slide over the width and height of the input volume and we will generate various 2-dimensional feature map including connections of filters and each spatial points in the input. This connection is a linear operation that can be expressed by $\sum_i w_i x_i + \varepsilon$. Here w_i denotes the weight given by the neuron and x_i denotes the element of input data. ε denotes the bias. The result of this weighted sum is given non-linear transformation through an activation function like ReLU (f(x) = max (0,x)). The network will learn filters that activate when they meet desirable specific visual features from input, like the edge of some lines and changes of colours at some regions. Thus entire patterns of some features will preserve in feature maps. As we have a set of filters and each of them will produce a distinct feature map. By stacking these maps along the depth dimension, we will finally get output volume from the convolutional layer.

Pooling Layer

Pooling layer is commonly inserted between successive convolutional layers in CNN. It is used to merge similar features into one in order to reduce the spatial size of the representation to reduce the number of parameters and cost of computation in the network. Hence, it can control overfitting. This can be done by combining the output at a specific position from the prior layer with nearby values into a single element. For example, max pooling operation chooses the largest value as a new output from the corresponding outputs involved. These corresponding outputs are selected from rectangles across feature maps given by the prior layer. Average pooling operation which sums up outputs in rectangles and takes average value as a new output. The pooling operation can make the representation robust to small changes of the input. It means that if the input get small change, most of the pooled outputs remain almost constant. This property is particularly useful when we only focus on if one feature exists in the input instead of the exact location.

Fully-Connected Layer

Fully-connected layer allows a connection between all feature maps from the previous layer to every neurons in fully-connected layer. Thus all outputs from the previous layer can be combined together. The combined output can then be used in other function or model for further calculation.

Applying CNN to Build Recommendation System Based on User Portraits

Commonly, CNN is used as to extract desirable features from the input. Previous studies proposed a multi-model approach to build the recommendation system (Y. Chen et al., 2021). It uses both CNN and LSTM. A CNN model is used to extract features of short text and LSTM is used to predict user tags from these extracted features. Here we focus on the CNN part as LSTM model has been discussed above in Equation 2.

Firstly, words in short text data are converted to word vectors by using Word2vec, a technique to convert a single word to a vector. For example, if each word is placed into an m-word vector

$(v_j \in \mathbb{R}^m)$ and a sentence $(Z_i \in \mathbb{R}^{m\times n})$ is limited to n words here. A sentence can be represented by a matrix of the size m × n. Thus each sentence can form a two dimensional matrix Z of size n × m where $Z_i = [v_1, \ldots, v_n]$, $v_i = [u_{i1}, \ldots, u_{im}]$.

Secondly, the matrices of sentences are put into convolutional layer with multiple filters. To extract various features of sentences, the operation of convolution followed by non-linearity can be given by Equation 3:

$$f(x) = \max(0, x) \qquad (3)$$

$$S = f(W * Z + \varepsilon)$$

where S denotes the feature matrix extract from convolution operation, W denotes weights used and ε denotes bias vector. The paper is using two convolutional layers successively to extract deeper feature. Finally, the paper used K-max pooling, which selects the top K-maximum values from the feature map given out from the latest convolutional layer. The number of K can be determined by Equation 4:

$$K = \left\lfloor \frac{\ell - f_S + 1}{2} \right\rfloor \qquad (4)$$

where ℓ denotes the length of sentence vector and f_S denotes receptive field of the kernel. Thus, these well extracted feature matrices can be used as input of LSTM model to predict user tags. Moreover, the combination of CNN and LSTM are widely used in building recommendation system (Lee, Ahn, Lee, Ha, & Lee, 2016; Wang, Ye, Yang, & Miao, 2020; Pradhan, Kumar & Pal, 2021; Zhou, Li, & Liang, 2020)

Advantage and Challenge

CNNs can be considered as a regularized version of multi-layer perceptions which is normally fully-connected network. Each neuron in one layer is fully connected to all neurons in another layer. As every connections between neurons need a weight, which will make the complexity of the network rapidly increase and cause overfitting. Besides, a huge amount of computation is required to train the network. CNNs reduce the issues by leveraging three important ideas: sparse connection, parameter sharing and equivariant representation (Goodfellow, Bengio, & Courville, 2016). By using filters with sizes smaller than input data to output feature maps that go through pooling layer, we can selectively preserve valuable features from the entire of input data. This can efficiently reduce parameters we need to store and take into account for calculation, especially for a deep model. Since each member of filters is used in every single element in input volume, only one set of parameters are needed to learn in filters rather than different sets of parameters need to learn for different elements in the input volume. This can reduce cost of training and memory requirement. The property shows out by the idea of parameter sharing is Equivariant representation. This allows CNNs to change its output when input is changed. Thus, CNN can detect a specific feature no matter its location in the grid. This help solve the problem of overfitting.

In order to build an efficient CNN model, various hyper-parameters are needed to be selected, like size and number of layers of the kernel, stride and number of convolutional and pooling layers. Tuning these hyper-parameters is time-consuming and needed to be well experienced to find suitable hyper-parameters, especially for a deep model with a great amount of hyper-parameters. Besides, collecting labelled data for training and testing requires a lot of human efforts. As the model is going deeper and deeper, there is an increasing cost of computation.

Classification Approaches

There are other machine learning techniques can be used in building recommendation system based on user portraits. They may have different functions in recommendation system. Here we introduce several classification methods to build user tags.

Support Vector Machine

SVM is a robust two-class linear classifier. It can determine the category of a new sample based on which side from the hyperplane the sample is in. The location of this hyperplane is determined by the nearest data point on each side, namely 'support vectors'. Though there are many hyperplanes we can choose to use for classification task, we are trying to find the most suitable one to minimize generalization error. Generalization error is a measure of accuracy of an algorithm to predict outcome values for a new data. If there exists a hyperplane that maximizes the distance from support vectors, then generalization error is minimised. In other words, it is the most appropriate hyperplane we are looking for. A typical SVM can only solve linear classification task whereas by placing input data to a transformed feature space to make input data linearly separable, SVM can solve nonlinear classification as well.

SVM can be used for text classification in order to generate user tags (Chen et al., 2021). Firstly, term frequency-inverse document frequency model is used to extract important features from users' raw text. Then, these features are mapped to a vector of higher dimensional feature space from the original space. A higher dimensional feature space is used to find an alternative representation of the features that is linearly separable. Thus SVM can fit on the pattern of the features in this feature space to do binary classification. In this way, such a well-trained SVM can be applied to classify users based on their retrieval text data in forms of gender, education and age.

Random Forest

Random forest is an ensemble learning method that construct a collection of decision trees at training time and combine them together to form a single tree model. First of all, we are going to discuss a single decision tree. Decision tree is a commonly used in machine learning to do classification as well as regression. It works by dividing the set of source into several distinct subsets of the input source to do classification. This can be done by using binary splits based on the features used. A feature is like a filter. When input values passing through a feature, it gives responses to input values. All values with the target feature will be placed to a new subset. The remaining values in input values will be placed into another separate subset. This method is done recursively on each derived subset. When the subset has all the same values of the target variable, or when splitting adds no more values to each subset, the recursion is finished. A single decision tree may prone to overfitting which give output in a high variance with a low bias. In order to control overfitting, random forest (Ho, 1995) can be used to decrease variance. It randomly samples one part from the training data with replacement and selects numbers of features from the whole available features to train for numbers of decision trees. When we input values to this collection of decision trees to do classification, the most frequent result of trees will be the final result of input values. Different trees in the forest may involve different information. By using random forest to take advantages of various information, error of prediction will seldom happen.

Random forest is used to make classification in movie recommendation in this paper (Zhang & Min, 2016) to determine whether a movie should send to a user based on the user's portrait and item's portrait. Tags for users and for movies are randomly selected as features used in separate decision trees. When data input to the forest, each tree will give a result between three options: recommend, not recommend and wait. The most frequent result given by the forest will be the final result of the input data.

Advantage and Challenges

Both SVM and random forest can be used to do classification. They have own advantages and challenges. Compared with SVM which is only capable for binary classification, random forest can fit classification of multiple classes. SVM can apply on multiclass classification. SVM can convert a multiclass classification to several binary classifications. This transformation needs to be designed manually which is time-consuming and requires a lot of experience. Besides, as random forest selects features to use completely randomly, it is a non-parametric model which has little preset on the pattern of trees. SVM is trying to find a hyperplane that the pattern is fixed before training. Thus random forest is more flexible and is more suitable for various types of data.

In order to build up random forest, numbers of parameters are needed to set up. For example, the number of trees in the forest, the number of features in each tree, etc. Changing of parameters may influence the efficiency of the model. SVM has no parameter needed to set up for a linear classification and one for non-linear classification. In order to make a random forest robust, large amount of time may needed to tune parameters.

Discussion

This paper gives a comprehensive review on current approaches for modelling user portraits. We introduce and compare several machine learning techniques used in this field, including LSTM, CNN, SVM and random forest. We illustrated the strengths and insufficient parts of each methods and emphasizes the conditions to apply each method. For example, CNN is often used for feature extraction, especially with contextual content from users. It is a strong technique to discover latent features of objects by using the word embedding method for mapping contextual information to a lower dimensional space with keeping major patterns. LSTM, a particular recurrent neural network, is capable to capture temporal dependencies and sequential information of words. It is a powerful dynamic system that is also very effective in capturing evolutionary latent features of users through time. LSTM can account for all time steps in the past in calculation. SVM is a two-class classifier, which can also be extended to multiclass classifier by joining multiple two-class classifier. It can be used to classify users and make recommendations based on gender, age, education level and past behaviours. Random forest is also a powerful and common technique to classify users. In general, it works by splitting the randomly selected feature subspace into a number of regions attached by a simple constant function, which is based on many binary splits. By taking into consideration of each individual feature of user portraits, it can generate a value to determine whether to recommend a certain item to a certain user.

Despite much progress has been made in this domain, some challenges remain unsolved. Knowledge transfer of the recommendation system trained in advance to a new field is concerned. To be more specific, the trained recommendation system may do the recommendation task accurately in one domain while it may not perform well in other domains For example, a recommendation system trained for the YouTube may not perform perfectly in restaurant recommendation for even the same user. Second, user portraits creation requires a great amount of collection of data from users. If there is insufficient data gathering from a new user, the recommendation system will be unable to provide personalized recommendation services for users.

In future work, we may solve those problems by applying cross-domain recommendation. Cross-domain learning enlarges the range of user portraits through data gathering across different platforms, this enables user portraits generated contains more valuable information of users. For example, if the portrait of a user is created through several platforms (e.g. healthcare, library and social media), this portrait may be suitable for users across different domains. It may be used for video platforms that can recommend videos depends on the portrait generated through other platforms to the same user. In this way, for a new user in a video platform, it can still do recommendation accurately by borrowing information across different platforms.

Conflict of Interests: the author has claimed that no conflict of interests exists.

References

1. Alaoui, S., Idrissi, Y. E. B. E., & Ajhoun, R. (2015). Building rich user profile based on intentional perspective. *Procedia Computer Science, 73*, 342–349.
2. Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks, 5(2)*, 157–166.
3. Chen, T., Yin, X., Peng, L., Rong, J., Yang, J., & Cong, G. (2021). Monitoring and recognizing enterprise public opinion from high-risk users based on user portrait and random forest algorithm. *Axioms, 10(2)*, 106.
4. Chen, Y., He, J., Wei, W., Zhu, N., & Yu, C. (2021). A multi-model approach for user portrait. *Future Internet, 13(6)*, 147.
5. Chen, Y.-C., Shang, R.-A., & Kao, C.-Y. (2009). The effects of information overload on con- sumers' subjective state towards buying decision in the internet shopping environment. *Electronic Commerce Research and Applications, 8(1)*, 48–58.
6. Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T., Gargi, U., . . . others (2010). The youtube video recommendation system. *In Proceedings of the fourth acm conference on recommender systems* (pp. 293–296).
7. Edmunds, A., & Morris, A. (2000). The problem of information overload in business organisations: a review of the literature. *International journal of information management, 20(1)*, 17–28.
8. Fedushko, S., Shakhovska, N., & Syerov, Y. (2018). Verifying the medical specialty from user profile of online community for health-related advices. *arXiv preprint arXiv:1812.06354* .
9. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press. (http:// www.deeplearningbook.org)
10. Gu, H., Wang, J., Wang, Z., Zhuang, B., Bian, W., & Su, F. (2018). Cross-platform modeling of users' behavior on social media. *In 2018 ieee international conference on data mining workshops (icdmw)* (pp. 183–190).
11. Gu, H., Wang, J., Wang, Z., Zhuang, B., & Su, F. (2018). Modeling of user portrait through social media. *In 2018 ieee international conference on multimedia and expo (icme)* (pp. 1–6).
12. Ho, T. K. (1995). Random decision forests. *In Proceedings of 3rd international conference on document analysis and recognition* (Vol. 1, pp. 278–282).

13. Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen netzen. *Diploma, Technische Universität München*, 91(1).
14. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural computation, 9(8)*, 1735–1780.
15. Huang, H., Zhang, Q., Wu, J., & Huang, X. (2017). Predicting which topics you will join in the future on social media. *In Proceedings of the 40th international acm sigir conference on research and development in information retrieval* (pp. 733–742).
16. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-scale video classification with convolutional neural networks. *In Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1725–1732).
17. LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., & Jackel, L. (1989). Handwritten digit recognition with a back-propagation network. *Advances in neural information processing systems, 2.*
18. Lee, H., Ahn, Y., Lee, H., Ha, S., & Lee, S.-g. (2016). Quote recommendation in dialogue using deep neural network. *In Proceedings of the 39th international acm sigir conference on research and development in information retrieval* (pp. 957–960).
19. Pradhan, T., Kumar, P., & Pal, S. (2021). Claver: An integrated framework of convolutional layer, bidirectional lstm with attention mechanism based scholarly venue recommen- dation. *Information Sciences, 559*, 212–235.
20. Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature, 323*(6088), 533–536.
21. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. *In Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 1–9).
22. Wang, W., Ye, C., Yang, P., & Miao, Z. (2020). Research on movie recommendation model based on lstm and cnn. *In 2020 5th international conference on computational 29 intelligence and applications (iccia)* (pp. 28–32).
23. Wu, T., Yang, F., Zhang, D., Zhu, A., & Wan, F. (2020). Research on recommendation system based on user portrait. *In 2020 ieee international conference on artificial in- telligence and information systems (icaiis)* (pp. 462–465).
24. Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation, 31(7)*, 1235–1270.
25. Zhang, H.-R., & Min, F. (2016). Three-way recommender systems based on random forests. *Knowledge-Based Systems, 91*, 275–286.
26. Zhang, P., Liu, B., Lu, T., Ding, X., Gu, H., & Gu, N. (2022). Jointly predicting future content in multiple social media sites based on multi-task learning. *ACM Transactions on Information Systems (TOIS), 40*(4), 1–28.
27. Zhou, X., Li, Y., & Liang, W. (2020). Cnn-rnn based intelligent recommendation for online medical pre-diagnosis support. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, 18*(3), 912–921.