Recommendation system based on user actions in the social network

Vitaly Monastyrev
Institute of computer science and technology
Peter the Great St. Petersburg Polytechnic University
Russia, Saint-Petersburg
vit34-95@mail.ru

Pavel Drobintsev
Institute of computer science and technology
Peter the Great St. Petersburg Polytechnic University
Russia, Saint-Petersburg
drobintsev_pd@spbstu.ru

***Abstract* – Currently, a large number of people use various photo hosting services, social networks, online services, and so on. At the same time, users leave a lot of information about themselves on the Internet. These can be photos, comments, geotags, and so on. This information can be used to create a system that can identify different target groups of users. In the future, you can run ad campaigns based on target groups, create recommendation ads, and so on. This article will discuss a system that allows users to identify their interests based on their actions in a social network. The following features were selected for analysis: published photos and text, comments on posts, information about favorite publications, and geotags. To identify target groups, the task was to analyze images in photos and analyze text. Image analysis involves object recognition, and text analysis involves highlighting the main theme of the text and analyzing the tone of the text. The analysis data is combined using a unique identifier with the rest of the information and allows you create a data showcase that can be used to select target groups using a simple SQL-query.**

Keyword — machine learning, recommendation system, natural language processing, image recognition

I. Introduction

Currently, humanity actively uses various Internet services and leaves a lot of different data on the Internet. This can be photos, text information, and so on. Based on this information, you can divide users into groups based on their interests. Many companies have their own recommendation systems that operate on this principle - Yandex [1], Google (YouTube) [2], Netflix [3].

In this article, we will look at a recommedation system that will identify interest groups based on the following data: photos, text, rated publications, and geotags. The final goal is to create a target data table (in SQL format). From the SQL table, you can get a list of users based on the specified interest using an SQL query. To create such a table, you need to recognize objects in images, and recognize the main theme and tone in the text. This will help you understand which topics the user treats positively, which ones negatively, and which ones are neutral.

Thus, the final table will contain information about what the user posts, what they comment on, what and how they evaluate, as well as information about geolocation. Based on this information, which is specific to a particular user, you can easily get different groups of users by interests and geolocation.

II. EXISTING RECOMMENDATION SYSTEMS

As mentioned above, many large companies use different recommendation systems to process their data. It all depends on the specific task and the available data, so companies build the data processing process in a way that is convenient for them and usually such solutions are not open source. These can be systems for recommending movies, music, friends, interesting authors, and so on. Let's look at some of them in more detail.

To generate a smart news feed, the social network Vkontakte marks data with the help of users who have received the status of experts [4]. These users vote for or against publishing on a particular topic. Then the marked-up data is already transmitted to the neural network, which is trained on it and improved. Due to the large amount of marked-up data, the neural network is well trained and can find similar publications that are more likely to attract users' interest. One of the disadvantages is that not every project can attract a large number of users for data markup. In addition, this solution is not an open source solution.

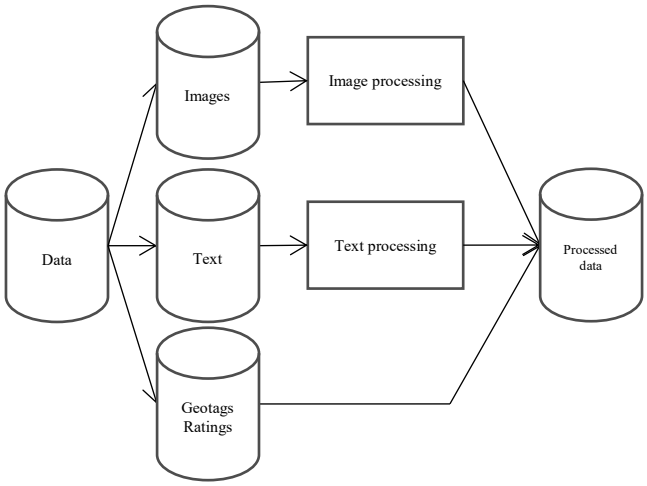
Another example is Yandex music. The recommendation system analyzes the user's actions: likes and dislikes, skipped tracks, repeated playback, and so on. Each action has weights that are later used in the algorithm. In addition, the system analyzes similar profiles. The final list of recommendations is compiled using Matrixnet [5], which processes the list of all possible recommendations and determines which ones should be shown to the user on the Yandex Music home page and in what order to place them. It is worth noting that more than a hundred training models are used when making recommendations for a single user. This consumes a large amount of resources — hundreds of servers collect data about user requests to the search engine, viewed products, etc. this approach can be used by large companies, but it is not suitable for small projects.

It is worth noting that the systems described above and other similar systems are sharpened for a specific set of data that a particular service works with. Also, the entire data processing process (data cleaning, preprocessing, model learning) is not open source. This article will discuss the process of working with the most popular data types, as well as building an algorithm for data processing and training models in such a way that this algorithm can be reused on other data types and in other projects.

III. Approach to building a recommendation system

The data set analyzed in this article was collected in one of the photo hosting services. This data set contains 127 images, 307 comments, 496 rating entries (likes and dislikes), and 47 geotags. The recommendation system will consist of several data processing modules. The algorithm of the system is shown in figure 1:

Fig. 1. The architecture of the recomedation system.



Raw data is sent to the system input. This data is divided into three categories:

- Images;
- Text;
- Geotags and ratings.

To identify user interests, images and text will be processed by machine learning modules. Image processing involves a module that will recognize objects in the image. Text processing includes two submodules: recognition of the main subject of the text and recognition of the tone of the text (positive or negative).

All processed data will be combined by a unique identifier (id). As a result, this will create a target tables that will contain the following information:

- What the user posts;
- What the user writes about and in what key;
- What the user evaluates positively;
- What the user evaluates negatively;
- Geotags attached to the user's records.

This data will help you identify user groups based on their interests. You can use interest groups to recommend new publications, recommend various products, and so on.

It is worth noting that MySQL [6] relational database was chosen for storing information. Moreover, images are not stored directly in the database, but are stored in the file system. The database stores only links to images. Machine learning modules are written in Python, as this language offers a wide range of tools for data processing.

IV. Machine Learning Modules

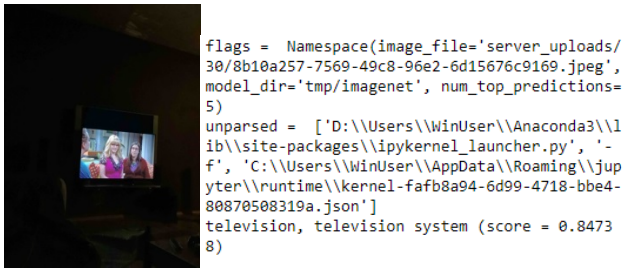
Let's take a closer look at how machine learning modules work for image and text processing.

A. *Module for recognizing objects in an image.*

The pre-trained Inception-v3 [7] model was used for recognizing objects in images. This is one of the most popular models for recognizing objects in images [8]. This model achieves an accuracy of more than 78.1% on the Imagenet dataset. The model has been trained in 1000 [9] classes. The use of the pre-trained model is due to the fact that the model has good performance, has open source code, is easily integrated into existing solutions, and works fast enough (about 1-2 seconds for 1 image on Intel core i7).

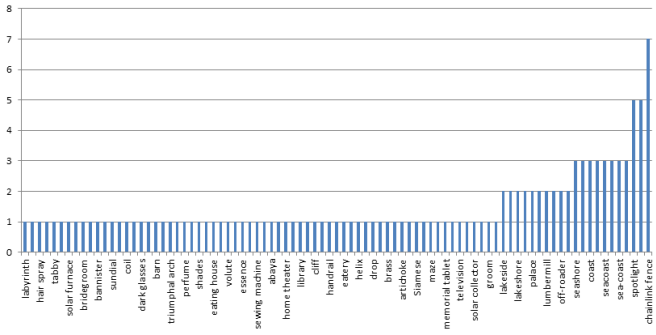
When analyzing images, this model outputs the top prediction classes with the highest score value. Within the recommendation system, only the value with the highest score was recorded. An example of how the model works is shown in figure 2:

Fig. 2. Example of how the image recognition model works.



In total, 127 photos from the original data set were processed using this model. Of the 1000 classes available in the model, 87 images were recognized. The average score value for all data is about 0.49. Information about recognized objects is shown in the figure:

Fig. 3. The recognized objects.



The most popular "chainlink fence" images shown on the chart are a classifier error, such images have a very small score. The most common objects are the sea, the coast, cars, and architectural objects. In the data set under consideration, the results of the classifier were analyzed. Correctly predicted values had a score greater than 0.5, so these images were considered correctly recognized and taken into account in the future (there are also incorrectly recognized images, but only about 10% of them).

All data was written to a MySQL table with the following fields:

- id;
- photo_id;
- photo_desc;
- score.

Where id is a unique identifier, photo_id is a foreign key from the photo table, photo_desc is the name of the recognized object, and score is the value of score.

B. *The analysis module of text subject*

Working with text is a more complex topic than image recognition, so there are no ready-made models here. This is because each language has its own grammar and it is difficult to adapt one model for all languages at once. In our case, the entire text was in Russian. However, there are various algorithms that can be adapted to your data and trained. To highlight the main topic of the text, a model

based on the Latent Dirichlet Allocation (LDA) [10] algorithm was used. The main idea of this algorithm is that each document is considered as a set of topics in a certain proportion. Each topic is a set of the most common word and each document consists of a specific set of words [14].

The origin data set cannot be passed directly to the model. First, you need to additionally process the text:

- Eliminate unnecessary characters (punctuation marks);
- Remove stop words (conjunctions, particles, etc.);
- Form stable phrases;
- Make lemmatization.

The simple_preprocess() method of the Gensim library [11] was used to remove punctuation and tokenize the text. To delete stop words, a set of stop words from the nltk [12] package was used. Bigrams and trigrams were formed as stable phrases using the Gensim library. The ru2 model from the spacy package was used for lemmatization.

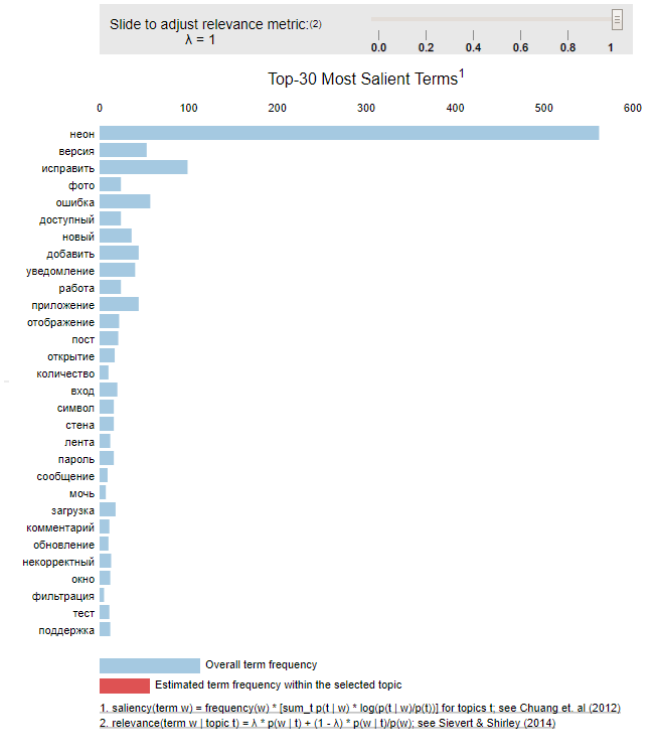
The main input data for the LDA model is the dictionary and corpus. Gensim creates a unique identifier for each word in the document, and the corpus shows the frequency of occurrence of this word.

One of the hyperparameters is the number of topics in the text. Since we had a fairly small data set, we set 20 topics. Other alpha and eta (was set to 'auto') parameters affect the sparsity of topics, chunksize (was set to 100) - the number of documents in each training chunk, and passes (was set to 10) - the total number of training passes.

To visualize the result, an interactive diagram was built using the pyLDAvis [13] package, which is shown in figure 3:

Fig. 4. Visualization of the LDA model operation.

Fig. 5. Visualization of the LDA model operation.



It is convenient to use the coherence metric to evaluate the model performance. As a result, the coherence metric of the model was 0.6561.

Additionally, an LDA model was built for each user individually, and the list of topics was set to 1. The coherence metric was also calculated for each model. The average value of this metric for all models was 0.6585. Data with the selected topics was written to a MySQL table.

C. The analysis module of text sentiment

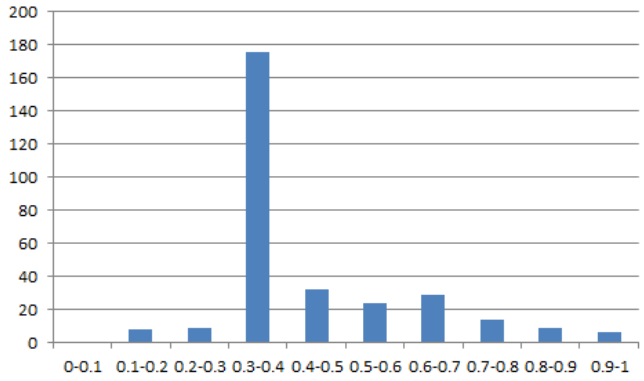
A convolutional neural network was used to analyze the tone of the text [15, 16, 17, 18]. The Word2Vec library was used to create the feature space. The training was conducted on a corpus of words based on Russian-language messages from Twitter, which contains 114991 positive and 111923 negative tweets, as well as 17639674 unmarked tweets [19]. Before training, all data was pre-processed (reduced to lowercase, replacing links to the token, etc.). The Word2Vec model was trained using the Gensim library. The Keras library [20] was used to build the neural network. This model, trained on tweets, was applied to text messages from the data set in question. The model metrics are shown in figure 4:

Fig. 6. Metric models the tone of the text.

	precision	recall	f1-score	support
0	0.76179	0.80437	0.78250	22236
1	0.79569	0.75180	0.77312	22534
accuracy			0.77791	44770
macro avg	0.77874	0.77808	0.77781	44770
weighted avg	0.77885	0.77791	0.77778	44770

This model was used to process the original data set that contained comments. As a result, the following results were obtained:

Fig. 7. The results of the model determine the tone of the text.



In this case, the abscissus axis shows the percentage predicted by the model, and the ordinate axis shows the number of similar comments. As you can see, most of the comments in the provided data set had a mostly neutral accent (values between 0.3 and 0.7 were taken as neutral, this data was viewed manually).

The trained model was used on the source data. All results were written to a MySQL table.

V. Results

The results of all three models were recorded in MySQL. All data is combined with a single id. This way we can now distinguish user groups based on their interests. As a result the database contains the following tables:

- Post. This table stores the id, photo and / or text, rating, geolocation (if available), and author of the publication;
- Comment. This table stores the id, publication id, text, rating, and comment author;
- Rating. This tables stores id, user id, photo id and rating (negative, positive or neutral);
- Object in the photo. This table stores the id, information about objects in the image (this information was obtained using the model), and the image id;
- Main theme of the text. This table stores the id, the main subject of the text, the type of post (post or comment), and the id of the post or comment.
- Tone of the text. This table stores the id, the tone of the text, the type of post (post or comment), and the id of the post or comment.

Let's look at an example of making recommendations using these tables. Let's say that we create an individual recommendation system to recommend interesting authors. To do this, we need to select what the user posts and what they rate positively (posts and comments). Then we need to find authors who publish similar images and recommend such authors to the user. For example, if these are sea coasts, we can use the following SQL query: “select distinct(photo.userid) from photo_desc, photo where (photo_desc.photo_desc like '%coast%' or photo_desc.photo_desc like '%sea%') and photo_desc.photo_id=photo.id and score > '0.5'”. 3 such users were found in the data set under consideration:

Fig. 8. Result of the SQL query.

	userid
▶	45
	66
	99

Another example of using this system is searching for a target audience. Let's say we want to find an audience for advertising a guitar store. Then we look for users who publish photos of guitars, write about them, or positively evaluate publications with guitars. You can also use information about geotags. In other words, we filter additionally those users who live near the guitar store.

VI. Conclusion

As a result, we implemented a recommendation system that allows us to identify target groups of users. The process of data processing by several machine learning models was considered. The Concept-v3 model was used for image processing, an LDA-based model was used to highlight the subject of the text, and a neural network-based model was used to determine the tone of the text. The model results were used for building SQL queries. The results of all models in the test data set were checked manually. For the object recognition model, the extreme score value was set to 0.5. For the text tone recognition model, the values 0-0. 3 were set for negative text, 0.3-0.7 for neutral text, and 0.7-1 for positive text.

This system can be used on small projects, since models are trained on marked - up data from open sources. In addition, the logic of setting up search targets is quite clear; it can be performed by any analyst who knows the SQL language. This architecture is suitable for almost any purpose, whether it is recommending services, searching for interesting publications, etc.

In future plans:

- Building the process of fully automating the launch of model training. To do this, you plan to use the Linux scheduler, or Jenkins/TeamCity;
- Implementation of a recommendation system in a real project. At this point, the data was received as a separate set of values and processed on a separate computer. For the full operation of the service, it is planned to transfer the entire data processing process to an industrial server;
- Analysis of model metrics. After implementing this system in the service, it is planned to analyze the accuracy of the models. This can be tracked by user clicks on the proposed content. It will also allow you to conduct A / B tests when some users see suggestions of recommendations from one model, and others from another. These tests will help you identify the best-performing models.

References

[1] Recommendation Technology ‘Disco’ URL: https://yandex.com/company/technologies/disco/.

[2] Covington, P., Adams, J., & Sargin, E. (2016). Deep Neural Networks for YouTube Recommendations. Proceedings of the 10th ACM Conference on Recommender Systems - RecSys '16. the 10th ACM Conference.
[3] Gomez-Uribe, C. A., & Hunt, N. (2015). The Netflix Recommender System. ACM Transactions on Management Information Systems, 6(4), 1–19.
[4] VK Experts URL: https://vk.com/press/theme-feeds.
[5] Matrixnet URL: https://yandex.ru/company/technologies/matrixnet/.
[6] Joel Murach: Murach's MySQL – 2012. – p. 612. ISBN 1890774685.
[7] TensorFlow models GitHub URL: https://github.com/tensorflow/models.
[8] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016, June). Rethinking the Inception Architecture for Computer Vision. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).
[9] 1000 synsets for Task 2 (same as in ILSVRC2012) URL: http://image-net.org/challenges/LSVRC/2014/browse-synsets.
[10] Bíró, I., & Szabó, J. (2009). Latent Dirichlet Allocation for Automatic Document Categorization. In Machine Learning and Knowledge Discovery in Databases (pp. 430–441).
[11] Gensim project page URL: https://pypi.org/project/gensim/.
[12] NLTK project page URL: https://www.nltk.org/.
[13] pyLDAvis project page URL: https://www.nltk.org/.
[14] Thematic modeling using Gensim (Python) URL: https://webdevblog.ru/tematicheskoe-modelirovanie-s-pomoshhju-gensim-python/.
[15] Jin, R., Lu, L., Lee, J., & Usman, A. (2019). Multi-representational convolutional neural networks for text classification. Computational Intelligence, 35(3), 599–609.
[16] Text tonality analysis using convolutional neural networks URL: https://habr.com/ru/company/mailru/blog/417767/.
[17] Cliche M. BB twtr at SemEval-2017 Task 4: Twitter Sentiment Analysis with CNNs and LSTMs //Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). — 2017. — C. 573-580.
[18] Zhang Y., Wallace B. A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification //arXiv preprint arXiv:1510.03820. — 2015.
[19] Rubtsova, Y. V. (2015). Построение корпуса текстов для настройки тонового классификатора. Международный журнал "Программные продукты и системы," 27, 72–78.
[20] Keras project page URL: https://keras.io/.