SiamRec : Music Playlist Recommendation Using Self-Supervised Network And Its Audio Representation

SungRae Hong

Korea Advanced Institute of
Science and Technology(KAIST)
sun.hong@kaist.ac.kr

Abstract

It is a very important task to recommend items that suit the tastes of users on the music streaming platform. Unlike platforms in other domains, users continuously consume items while staying on the music platform. Until now, music recommendations have been made using the Collaborative Filtering (CF) algorithm. However, in order to properly connect new songs with users added every day, any methodolgies to solve the cold-start problem were needed. In this study, we propose a playlist recommendation using audio presentation, which has not been frequently used. A network that has trained with large amounts of audio with self-supervised learning recommends new items to the appropriate playlist based on the presentation of existing playlist songs. To train the network, we propose three novel augmentation which is applicable to audio data. Our experimental results are presented qualitatively or quantitatively. SiamRec's source code is available at `github.com/HongSungRae/SiamRec`.

1 Introduction

Collaborative Filtering(CF) proved to be a powerful algorithm when it won the *Netflix Prize*[Bennett *et al.*, 2007]. Since then, it has been a big stream in the recommendation system for numerous follow-up studies. CF is an algorithm with two assumptions. First, it is assumed that any user similar to other user cluster will follow their tendency. It can be interpreted that there is a strong correlation between users' behavior. Therefore, if you find a group of users similar to a certain user, you can recommend an item. Second, it is assumed that items preferred by some users will have similar items and high preferences. Therefore, if any user has a preferred item, it is possible to recommend it based on that item.However, CF inevitably have one problem. It is a *Cold-Start Problem*[Lam *et al.*, 2008].

The cold-start program is caused by the assumption of CF and the characteristics of its algorithm. CF assumed a correlation between users, but if a user who has no correlation appears, no recommendation can be made. In particular, when a new user appears, there is a problem that it cannot be recommended as a CF until the user's behavior is known. There is also a problem with the emergence of new items. It is difficult to recommend items to suitable users because they have never been consumed by any user.

Apart from the cold-start program, there is also a problem that occurs as the number of items and users increases. For example, in Amazon.com, there are more than 100 million members and more than 400 million items which increase thousands of mounts everyday. A matrix consisting of 100 million rows and 400 million columns is difficult to implement and manage efficiently. If only one new item be added, 100 million ingredients must be added. As the number of items increases, it is more likely that users don't know existence of items not they don't prefer it.

In particular, this phenomenon must be resolved on music streaming platforms. Unlike platforms such as shopping platforms, users in music platforms continuously consume items while they are staying. Users can escape the platform at any time if they don't think they're getting good recommendations. Therefore, there have been attempts to recommend music using meta information such as artist name, track name and genre. However, these recommendations were often trivia or predictable for users. For example, if there is a playlist about a specific artist, it is recommended when his/her new song released. In the case of a specific genre, classical music that is popular in the genre manias was often recommended.

To solve this problem, we propose to train audio data to neural network using self-supervised learning. The trained network analyzes the representation of the playlist and recommends the most similar song among the new songs. Self-supervised learning requires various augmentation to prevent collapsing problems. We propose three novel augmentations applicable to audio data.

We present the novelty of this study as follows.

- We apply self-supervised learning to recommendation system research and verify its effectiveness.
- We propose three augmentation applicable to audio data.

2 Related Work

2.1 Recommendation System On Playlist

There are studies that have tried to solve the cold-start problem to recommend a music playlist. Chen et al., defined the problem by dividing the case into cold user, cold item, and cold playlist[Chen *et al.*, 2019]. This study solved the problem through multi-task learning. By introducing bipartite loss, songs on the playlist created by the user be closed and not closed which the songs not in the playlist.

van den Oord et al., recommended music based on contents[Van Den Oord *et al.*, 2013]. Using weighted matrix factorization, the user's implicit preference was reflected in factorization. There are too many rows and columns in the matrix, so the matrix was optimized using alternating least squares (ALS) optimization. Finally, 29 seconds of audio clips were used together to enrich the music meta data.

2.2 Representation Learning

Self-supervised learning or representation learning is a learning method in which a model is trained to learn the data representations based on a large amount of unlabeled data and transferred to other main tasks such as classification. These approaches typically set the pretext task for learning the data representations in a heuristic way and use the model trained with the pretext for a downstream task. For instance, there are relative image patch prediction, image rotation angle prediction, and image color jittering.

Self-supervised learning is usually divided into generative and contrastive learning. Generative presentation[Liu *et al.*, 2021] learning trains an encoder that assumes the data distribution and converts an input into an explicit vector, and then a decoder that reconstructs the input from the explicit vector. This method mainly uses an auto-regressive model or an auto-encoding model.

Contrastive representation learning[Liu *et al.*, 2021] also converts an input into an explicit vector through an encoder. However, contrastive methods do not have a generation process but measure similarity. These approaches learn representations by narrowing the distance of similarity between different versions of the same data or 'positive samples', and by lowering the similarity from representations of different data or 'negative samples'. Contrastive learning requires a large amount of data for comparison between positive and negative samples, but has shown the highest performance in classification tasks recently.

2.3 Representation Learning of Audio Data

Self-supervised learning for audio data such as music has been mainly based on contrastive methods. Janne et al. classified music genres with musical representations learned from SimCLR using negative samples and contrastive losses. Jiyoung et al. also trained Siamese DCNN using artist labels and applied transfer-learning to music classification and retrieval tasks. In addition, not only artists but also meta information originally annotated in music such as album and track information has been used to train networks for representation learning.

2.4 Siamese Networks

Most of the self-supervised learning approaches are based on the Siamese networks. The Siamese network is a general architecture for comparing the two data inputs. This network typically consists of an encoder that combines the CNN backbone and a projector and is trained by increasing the similarity between the output vectors from the two augmented views of input data. Siamese networks generally have a collapsing problem. Collapsing refers to a phenomenon in which all networks' outputs converge to a constant, so loss converges without properly learning representations. Xinlei et al.[Chen and He, 2021] suggested Simsiam to show that these Siamese networks can solve the collapsing problem and perform well enough with only the Siamese structure itself and stop-gradient without contextual learning or momentum encoder.

This study focus on representation learning using music data augmentation based on this Simsiam architecture. We present an effective feature learning methodology using only the Siamese network structure with stop-gradient and data augmentation without applying a contrastive approach that requires a large dataset.

3 Method

In this chapter, we present the structure of the model and the training methodology. The dataset and crawling and augmentation methodology will be proposed. Finally, the reward score, a rank-based metric for performance evaluation, is presented.

3.1 Architecture

Our architecture named SiamRec is based on the SimSiam. x_1 and x_2, two augmented views of data x, are entered into the model. x can be audio or image data. The overall structure of the model is shown in Figure 1. The two views x_1 and x_2 pass through an encoder consisting of a backbone and a projector. CNN-based models such as ResNet, or Transformer's encoder structure can be used for the backbone. The projector is an MLP consisting of 3 layers. Encoder shares weight between the x_1 and x_2. Predictor is also an MLP structure, which predicts output using the vector transformed from x_1 through the encoder to match the x_2. For a projected vector of x_2 that has not passed through the predictor, a stop-gradient operation is applied. The stop-gradient which does not propagate gradient is significant to prevent the collapsing problem of the Siamese network. The goal of SiamRec is to minimize the negative cosine similarity between the two outputs from x_1 and x_2. We define the loss to train the model as follows. h is the predictor and sg is the stop-gradient operation.

$$S(x_1, x_2) = -\frac{x_1}{\|x_1\|_2} \cdot \frac{x_2}{\|x_2\|_2} \tag{1}$$

$$Loss = \frac{S(sg(x_1), h(x_2)) + S(h(x_1), sg(x_2))}{2} \tag{2}$$

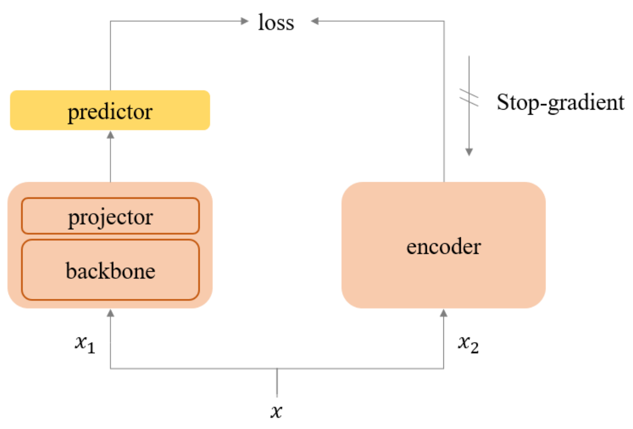

Figure 1: SiamRec architecture

Audio Effect	Description
compressor	Compresses the volume of the sound.
gain	Amplifies the volume of the sound.
chorus	Shakes the sound to spread it and make it abundant.
filter	Filters out the sounds more than cutoff.
phaser	Makes a buzzing sound like a jet is passing by.
distortion	Distorts and crushing the sound.
reverb	Makes the sound like the sound of a concert hall.

Table 1: Audio Effect

3.2 Data Augmentation

We used data augmentation in three ways. First, a method of giving various modifications to audio data was applied. A total of 7 audio modification effects were randomly applied: *compressor, gain, chorus, filter, phaser, distortion, and reverse*. A description of each audio effect is shown in Table 1.

Next, audio random-mix was applied for the audio augmentation. We cut the sequence of audio into small pieces, then mixed the order randomly and connected the pieces.

Finally, the spectrogram image of each audio was extracted and the imge augmentation method was used. We applied a total of six image augmentation methods to the spectrogram image: *horizontal flip, random crop, hue sature, random brightness contrast, rgb shift, and channel shuffle*. The horizontal flip was applied randomly, and the rest of the augmentation techniques were always applied to the image, but the degree of effect for each technique was random.

3.3 Dataset

Here, the dataset and acquisition method used for training and inference are described.

Pre-Training Data

FMA_small was used for network pre-training. FMA_small is a subset of FMA_medium. The data contains about 7,700 audio of 30 seconds, including 202 genre classes equally. Audio is data in the form of mp3. In python, the mp3 load is still very slow, so all mp3 files were converted to json-type files and used.

Playlist

Spotify-Million-Playlist was used in the experiment. This data is collected by spotify and contains 1 million playlists. Each playlist contains not only song title but also meta information such as artist name, number of tracks, collaboration, and number of follower. Among the 1 million playlists, 100 playlists with the largest number of follower were extracted. This is because it is assumed that it is created through collaboration of many people and that most users' tastes can be reflected only when there are many followers.

Youtube Crawling

The neural network receives songs in the playlist and outputs the presentation of the playlist. Therefore, songs on the playlist should be downloaded and used as datasets. A total of 2,000 songs were downloaded from YouTube, including 20 added at the end of the 100 playlists. After writing a code that automatically extracts URLs, it was automatically downloaded with a library called YouTube-DL. In some cases, advertisement videos are exposed at the top, so when a language except English in the playlist appeared, all of them were considered advertisements. It was confirmed that there was no legal problem with crawling the sound source for research purposes.

3.4 Inference

In the experiment, ResNet50, ResNet101, ResNet152 and Transformer were used as backbone network. Pedalboard, random mix and spectrogram image augmentation were used as augmentation. The network was self-supervised-learned using only FMA data. Forward the song extracted from the playlist to the train network. Forward the ground truth song in the playlist and the song that is not. Compare the two results. The song with the highest similarity is recommended on the playlist.

3.5 Metric

Metric is a measure based on rank. Mask the same number of songs on the playlists. Equation 3 shows *Reward Score*. Masked songs are called *candidates*. The songs left on the playlist are forwarded to the pre-trained network to find the songs on the original playlist among the candidates. Notations are in the table 2. Note that we used $n(P_i) = 20$ songs per playlist.

The process of calculating $f(s_k, P_i)$ is as follows. First, among the songs in P_i, unmasked songs are forwarded to trained network and averaged to obtain a playlist vector v_{P_i}. Second, all M_P are forwarded to the network to obtain masked songs vectors. And then sort the masked songs vectors in order closest to v_{P_i}. Third, the query vector is obtained by forwarding s_k to the network. Finally, find the index that v_{s_k} appears among the sorted vectors.

Notation	Description
P	Playlists
P_i	Playlist i
$n(\cdot)$	The number of $\cdot$
M_P	Masked songs list from playlists P
M_{P_i}	Masked songs list from playlist P_i
s_i	A single song i
$v.$	Vector. Output of the network

Table 2: Notations

$$\frac{1}{n(M_{P_1})}\sum_{P_i \in P}\sum_{s_k \in M_{P_i}}\frac{1}{\lceil (f(s_k, P_i)/n(M_{P_i})) + 1\rceil} \quad (3)$$

4 Experiment

In the experiment, four backbone networks were used: *ResNet50, ResNet101, ResNet152*, and *Transformer*. And three augmentation were used: *Audio Effect*, *Random Mix*, and *Spectrogram Image Augmentation*.

Adam optimizer was used with weight decay 0.00001. Except for some experiments, pre-training was trained 100epochs. All operations were computed on two GPUs.

4.1 Pre-Training : Siam Architecture

As mentioned in Section 3.2, each Siamusic model was pre-trained by applying three augmentation methods: *audio effect, audio randommix, and image augmentation*. In addition, each Siamusic model was compared by applying four backbone networks: ResNet50, ResNet101, ResNet152, and Transformer. In the case of the model to which image augmentation was applied, only ResNet50 and ResNet152 were compared. All experiments for pre-training were conducted on two datasets, FMA_small and MTA, while experiments for fine-tuning were conducted on FMA_small, MTA and GTZAN.

Figures 2, 3, and 4 show the results of pre-training with a combination of each backbone and augmentation using FMA dataset. As shown in Figure 2, when the Transformer was used as the backbone compared to the ResNet, the training loss did not decrease stably. In the case of using random-mix augmentation as shown in Figure 3, collapsing problems occurred in all cases.The training loss also decreased unstably and tended to collapse in the case of image augmentation.

4.2 Quantitative Result

Table 2 shows the quantitative performance of SiamRec. R@m/n means that network get m songs on the playlist as input and n songs were masked. 'R' is reward score. Overall, it did not make a perfect recommendation. However, the overall performance was high in a model with pedalboard augmentation on the ResNet101 backbone. In addition, the transformer with pedalboard augmentation showed the best performance at R@18/2.

4.3 Qualitative Result

The qualitative evaluation of the pretrained Siamusic model was conducted using a total of 50 songs from 10 artists. Five

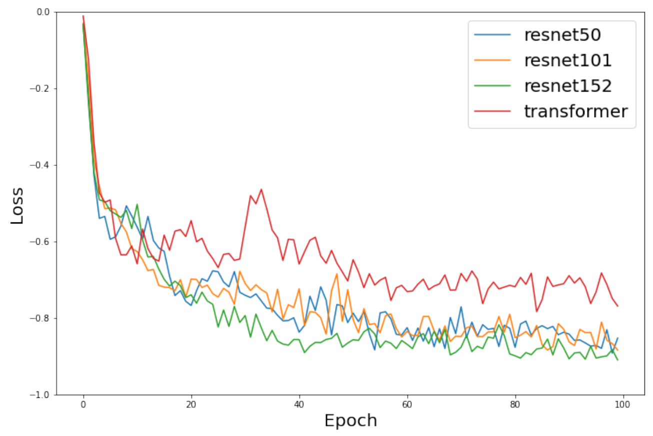

Figure 2: FMA using Audio Effect Augmentation

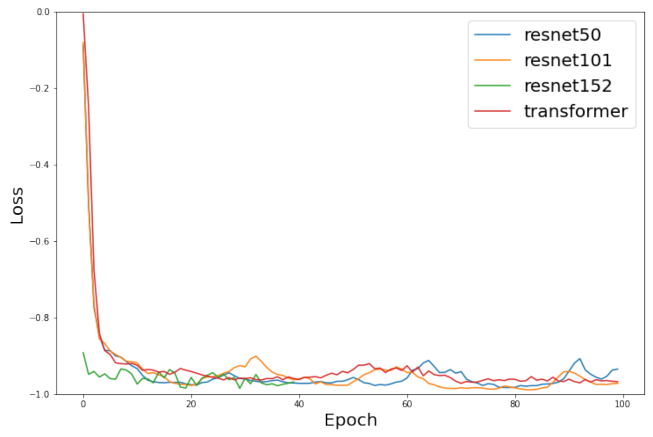

Figure 3: FMA using Random-Mix Augmentation

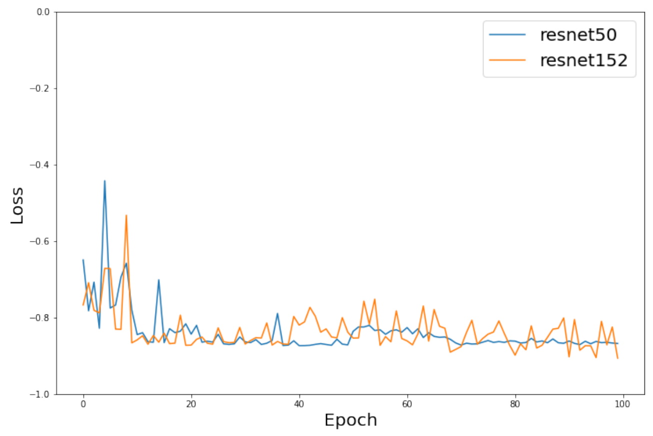

Figure 4: FMA using Image Augmentation

Backbone	Augmentation	R@10/10	R@13/7	R@15/5	R@18/2
ResNet50	Pedalboard	4.93±0.29	4.88±0.33	**5.30±0.42**	**5.32±0.88**
	Random Mix	4.90±0.37	5.06±0.29	5.08±0.46	5.02±0.86
	Image Aug	4.92±0.22	4.98±0.34	5.01±0.12	5.02±0.11
ResNet101	Pedalboard	**5.22±0.34**	**5.19±0.43**	5.06±0.51	**5.37±0.44**
	Random Mix	5.02±0.30	5.06±0.25	5.02±0.48	4.83±0.46
ResNet152	Pedalboard	5.04±0.30	4.78±0.52	4.95±0.38	4.91±0.90
	Random Mix	5.13±0.32	5.07±0.41	5.10±0.40	5.05±0.71
	Image Aug	5.07±0.40	5.06±0.39	5.12±0.29	5.09±0.68
Transformer	Pedalboard	5.13±0.33	5.01±0.34	4.77±0.34	**5.40±0.97**
	Random Mix	4.98±0.46	4.84±0.33	4.81±0.37	4.65±0.93

Table 3: Performance On Playlist

songs were selected for each artist to form a dataset. After that, the songs were plotted in a graph using tSNE method.

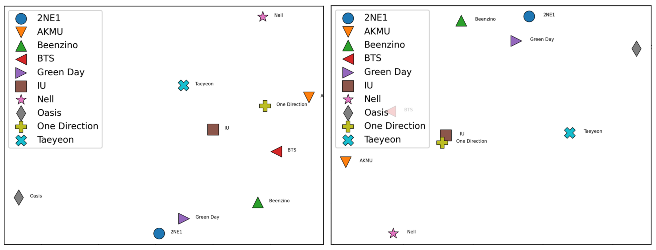

Figure 5: tSNE of Pedalboard Augmented Artists

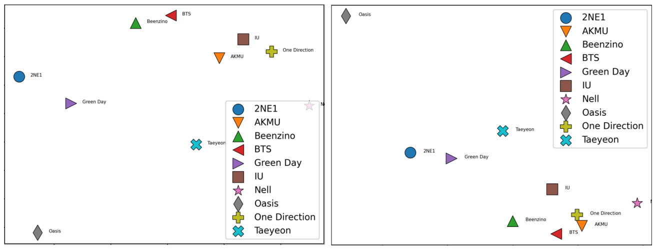

Figure 6: tSNE of Random Mix Augmented Artists

We visualized how the pre-trained networks map 10 POP artists and their 50 songs at the latent space using tSNE. It showed a common phenomenon. Figures 5 and 6 are the results from different augmentation of ResNet50 and ResNet101, respectively. Two networks commonly mapped Oasis far from other artists. Taeyeon and Nell were also mapped far away from other artists. It can be seen that IU, BTS, and One Direction are mapped to similar distance.

Their 50 songs, 5 songs each, were plotted using tSNE. Figures 7 and 8 are the tSNE mapping results of ResNet50 trained by random mix and pedalboard augmentation, respectively. Both results can show common characteristics. Beenzino's raps were mapped to a similar space. Taeyeon's songs were also mapped to a similar space.

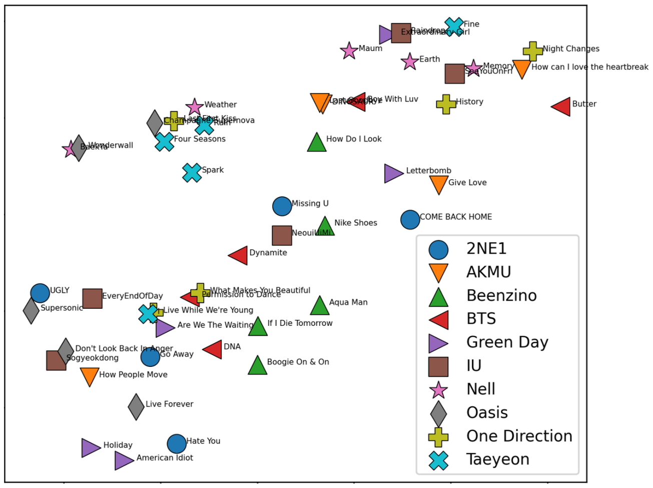

Figure 7: tSNE of Pedalboard Augmented Songs

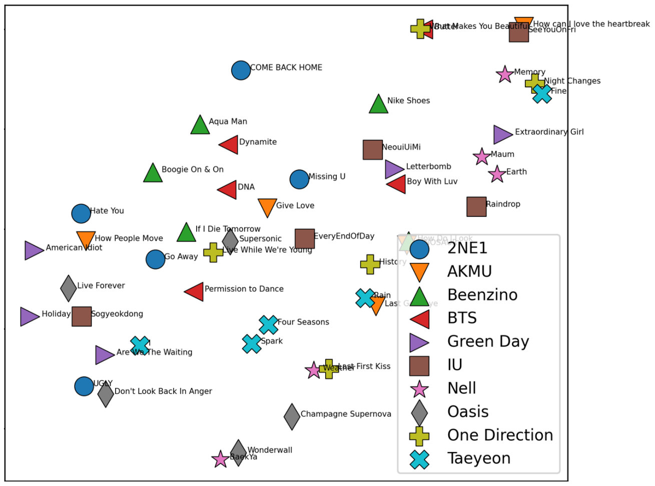

Figure 8: tSNE of Random Mix Augmented Songs

5 Conclusion

In this study, representation learning was proposed to solve the cold-start problem and trivial recommendation in the music domain. To prevent network collapsing, three augmentation methods that can be used for audio proposed. By presenting the experimental results quantitatively and qualitatively, it was shown that learning audio presentation directly can help recommend new songs.

There are also limitations to research. Since the network trained only the presentation of audio, it ignored the rich information of other meta data. If audio presentation and meta data were used together, it would have become a more powerful model. In addition, the trained representation not yet reliable. It seems to be due to the small amount of learning data, and in the case of image self-supervised learning, more than 100 millions of image data are used for training.

This study applied self-supervised learning and audio presentation to recommended tasks. We hope that related research can be continued in the future.

References

[Bennett *et al.*, 2007] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. New York, NY, USA., 2007.

[Chen and He, 2021] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15750–15758, 2021.

[Chen *et al.*, 2019] Dawei Chen, Cheng Soon Ong, and Aditya Krishna Menon. Cold-start playlist recommendation with multitask learning. *arXiv preprint arXiv:1901.06125*, 2019.

[Lam *et al.*, 2008] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. Addressing cold-start problem in recommendation systems. In *Proceedings of the 2nd international conference on Ubiquitous information management and communication*, pages 208–211, 2008.

[Liu *et al.*, 2021] Xiao Liu, Fanjin Zhang, Zhenyu Hou, Li Mian, Zhaoyu Wang, Jing Zhang, and Jie Tang. Self-supervised learning: Generative or contrastive. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[Van Den Oord *et al.*, 2013] Aäron Van Den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Neural Information Processing Systems Conference (NIPS 2013)*, volume 26. Neural Information Processing Systems Foundation (NIPS), 2013.