Sequential skip prediction using deep learning and ensembles

Ferenc Béres
MTA SZTAKI[†]
beres@sztaki.hu

Domokos M. Kelen
MTA SZTAKI[†]
kdomokos@sztaki.hu

András A. Benczúr
MTA SZTAKI[†]
benczur@sztaki.hu

ABSTRACT

In this paper, we present the solution of the team Definitive Turtles for ACM WSDM Cup 2019 Spotify Sequential Skip Prediction Challenge that reached 10th place on the public leaderboard. Our solution is based on gradient boosted trees (GBT) combining several statistical features as well as the output of a neural network classifier. We were able to train GBT by using small size data samples, and assess the importance of the various features and classifiers for skip prediction.

KEYWORDS

Music Recommendation, Skip Prediction

ACM Reference Format:
Ferenc Béres, Domokos M. Kelen, and András A. Benczúr. 2019. Sequential skip prediction using deep learning and ensembles. In *WSDM 2019: ACM International Conference on Web Search and Data Mining, 11-15 February 2019, Melbourne, Australia.* ACM, New York, NY, USA, 4 pages.

1 INTRODUCTION

One task of the 2019 ACM WSDM Cup was sequential track skip prediction. The data [1] provided by Spotify contained several music listening sessions. Challenge participants had to infer which songs will be skipped in the second part of each session based on the track skip events in the first part of the sequence. The task is in part a conventional binary classification problem, and in part a collaborative filtering recommender task. The immense size of the data and the missing user information from the sessions also made this challenge both arduous and novel.

In this paper, we present the solution of the team Definitive Turtles that reached 10th place on the public leaderboard. Our solution is based on gradient boosted trees (GBT) combining several statistical features as well as the output of a recurrent neural network classifier. By using GBT, we were able to assess the importance of the various features and classifiers for skip prediction. A big advantage in our approach is that GBT model weights were trained using only 2.4% of the public dataset, approximately 50 million records out of the total 2072 million.

[†]Institute for Computer Science and Control, Hungarian Academy of Sciences

WSDM 2019, 11-15 February 2019, Melbourne, Australia

1.1 Data

The challenge data consists of 160 million music listening sessions with rich session and track metadata. The length of each session is up to 20 songs. For most of the sessions, a track start and end reason annotation, a premium user flag, and timestamps were also available. In total, users interacted with almost 4 million tracks in these sessions. The dataset includes general metadata (e.g. duration, instrumentalness, release year) as well as an acoustic feature vector for all of these tracks.

Skip events can follow three different scenarios, the track only played **very briefly**, or **briefly**, or **most of the track was played**. Skip information was encoded in three binary fields skip_1-3; note that for 0.001% of the data, the three bits were inconsistent. If none of the bits were set, we assumed that the user listened to the entire track. In the Challenge task, the skip_2 field served as ground truth label, which means the track was only played very briefly or briefly.

The full data set consists of a public and a hidden part with approximately 130 million and 30 million music listening sessions. In the public data the skip information is provided for each track while in the hidden part this label is withheld for evaluation purposes. For these hidden sessions the participant is provided all the user interaction features for the first half of the session, but only the track IDs for the second half.

1.2 Evaluation

The following two evaluation metrics were used in the Challenge:

MAA: Mean Average Accuracy (MAA) was the main evaluation metric. It has two favorable properties. First, it can measure whether our guess for the next immediate track is correct. On the other hand, it is the weighted average of accuracy at each position, assigning higher relevance for the first few interactions. The formula of Average Accuracy (AA) proposed by the organizers is

$$\mathrm{AA} = \frac{\sum_{i=1}^{T} A(i)L(i)}{T}. \quad (1)$$

Here, T is the number of tracks to be predicted for the given session, $A(i)$ is the accuracy at position i of the sequence and $L(i)$ is the Boolean indicator for whether the i^{th} prediction was correct.

FPA: First prediction accuracy (FPA) was the tie breaking secondary metric of the competition, which measures the average accuracy of the predictions for the first unknown skip value in each session.

2 FEATURE ENGINEERING

We defined 508 features for each track in each session that we describe next. We give our final prediction by a Gradient Boosted Tress Classifier [3] using these features along with the prediction of the deep learning classifiers that we will describe in Section 3.

2.1 Track skip information

As each track is identified by a track ID, we can calculate the total number of listenings, as well as the skip ratio for every song and for each skip type. Our intuition is that for a given track, these statistics can depend on the session position, thus we computed them for three different scenarios:

(1) first half of every session (counts rounded down) in the public and hidden part of the data;
(2) second half of every session in the public data;
(3) the union of the former two cases.

We planned to calculate skip statistics for each artist and album as well. Unfortunately, artist and album information was removed from the data a few weeks after the start of the competition.

2.2 Session heterogeneity

In our intuition, users skip tracks that are dissimilar to their usual consumption pattern more likely. To detect outliers in a session, we calculated the Mahalanobis distance [6] between the given song vector and the mean representation of the session tracks.

Another key feature that can describe session heterogeneity is the information about track repetitions. Thus we considered i.) **repeat count**, the number of times a given track was repeated in the current session, and ii.) **unique track ratio**, which is the number of unique tracks in the session divided by its length.

2.3 Skip-pattern features

For skip prediction, the *last action* proved to be a strong baseline, which reached 19th place in the competition. Last action predicts the last known skip value of the user for the rest of the session. In our skip-pattern model, we take this idea further: for every possible sequence of skip values at the first part of the sessions, we calculated the fraction of skip in the second part of the public sessions.

In our **binary model**, we only consider a track either skipped or not skipped. For the 1024 possible skip patterns, the least frequent one appeared in 3278 sessions of the public data. The binary model itself reached MAA 0.546 and AUC 0.695.

In our **ternary model**, we distinguished three kinds of skip values: no skip, most of the track played but then skipped, and played only briefly (or very briefly). In this case, there are 59049 possible patterns and the least frequent one occurs only 9 times. Since very low frequency skip ratios are noisy, we included not just the ratio but also the sample size in our feature set. The ternary model itself reached MAA 0.552 and AUC 0.709.

2.4 Aggregate statistics

We calculated the mean, standard deviation, minimum, and maximum values for most of the raw and the implemented features with respect to the sessions. It is important to note that for some session related variables (e.g. user behavior, context type), these statistics are restricted to the first half of the sequence, as the feature value is not available for the second part.

3 DEEP LEARNING METHODS

3.1 Denoising autoencoder

Autoencoders are models that aim to learn a function that reproduces its input as closely as possible, while limiting the size of the function's internal representation of this input. Denoising autoencoders [7] additionally handle the task of removing noise from the input: instead of returning the input itself, they are expected to return the input without noise.

Our model embeds tracks in a vector space [5]. We consider the sessions as sets of items, where each item in the set has either negative or positive sign. We expect the model to correctly predict the sign of all of the items in the session as a skip prediction without revealing all the skip values. Formally, we have two matrices, $U \in \mathbb{R}^{N\times k}$ and $W \in \mathbb{R}^{N\times k}$, where N is the number of all track IDs, and k is the dimension of the embedding vector. These matrices are the encoding and decoding layers of the network, functioning as embedding matrices. Given an input set $\{(i_1, s_1), (i_2, s_2), \ldots, (i_n, s_n)\}$, where n is the number of visible (i.e. not hidden) items in the set, i_m is the track ID and $s_m \in \{-1, 1\}$ is the sign of item m in sequence, we can calculate the prediction for s_m as

$$\tilde{s}_m = \left\langle W_m, \sum_{i=1}^{n} U_i s_i \right\rangle. \tag{2}$$

We try to minimize a loss metric $L(\tilde{s}_m, s_m)$ over all of the items in the session (visible *and* hidden) using stochastic gradient descent. After experimenting with multiple standard loss metrics, we observed that, when measured on a holdout set, the squared loss over the raw dot products generalized best:

$$L(\tilde{s}_m, s_m) = (\tilde{s}_m - s_m)^2. \tag{3}$$

Other metrics we tried include the squared norm over the $\tanh(\tilde{s}_m)$ values and the binary cross entropy loss as well.

We experimented with two training approaches: randomly hiding half of the set from the input, and hiding the second half, the latter similar to how the actual prediction task is arranged. We observed that the first approach worked slightly better. We also applied a dropout value of 0.3 on the individual U_i vectors, which improved the model.

This model achieved an MAA score of 0.505 and an AUC score of 0.650 in our measurements. On their own, the values are weaker than even some baselines, however the prediction combines well with other information sources. As we will describe in Section 4.2, the prediction of the autoencoder forms an important feature in our final GBT model.

3.2 Sequential denoising autoencoder

Note that the autoencoder model described in Section 3.1 considers the sessions as sets of signed track IDs, and disregards all information about their sequential ordering. Since this could carry useful information, we also experimented with sequential autoencoders [2]. The task remains the same as in Section 3.1, except for the fact that instead of considering the input sessions as sets, we treat them as lists of (i_m, s_m) pairs in their original order. Accordingly, instead of adding them up, the $U_i s_i$ values are fed to an LSTM network [4]. As prediction, we use the dot product of the network output and

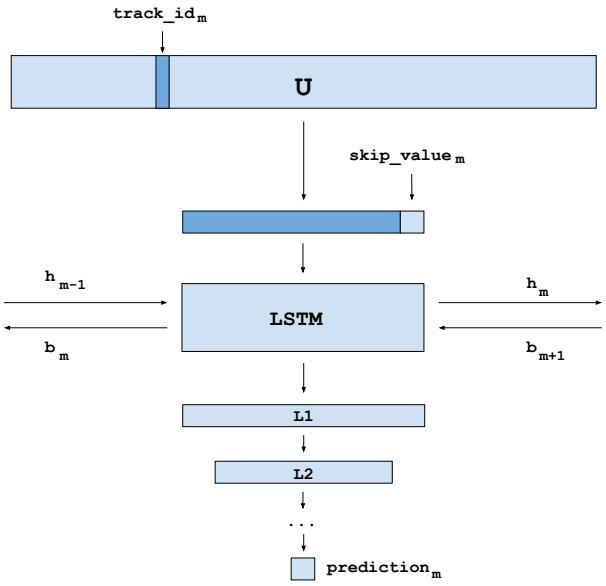

Figure 1: Architecture of the recurrent classifier network. The model features an embedding matrix U with 128 dimensions, a bidirectional LSTM network with 2 layers, and 5 fully connected layers with ReLU activation. The variables b_m and h_m symbolize the forward and backward directional hidden layer vectors of the LSTM network.

W_m,

$$\tilde{s}_m = \left\langle W_m,\ \mathrm{LSTM}^{(i)}(U_i s_i) \right\rangle. \tag{4}$$

This approach was able to improve upon the results of the set-based autoencoder approach slightly, however still does not produce very good results on its own. It achieved an MAA score of 0.551 and an AUC score of 0.716 in our measurements. Unfortunately, due to limited time we were not able to include this model in our final ensemble.

3.3 Recurrent classifier network

The approaches described in Sections 3.1–3.2 both lack explicit information about the skip values, which is only indirectly encoded in the direction of the learned embeddings. Furthermore, they can only receive information about track IDs for which the corresponding skip values are known, even though the track ID could carry useful contextual information even when the skip value is unknown.

To handle these issues, we use a recurrent classification model. We keep the embedding matrix U from the autoencoder approach, however instead of taking the product of the s_i and U_i values, we concatenate them and feed these vectors to an LSTM network. The decoder part of the network is replaced by stacked fully connected layers with ReLU activation, with decreasing number of neurons, see Figure 1. This setup allows us to directly feed the network with the skip values $(-1, +1)$, and also with explicit *unknown* values, represented by 0.

In the training phase, we hide the second half of the sessions, simulating the prediction task. The model was tested in both the single directional and the bidirectional case, with the latter performing better: the single direction model had an AUC of 0.750, while the bidirectional model had an AUC of 0.774. This proves the assumption that the unlabeled track IDs carry useful information about the skip values.

Our recurrent model achieved an MAA score of 0.602 in our measurements, which is stronger in itself than our final submission. Unfortunately, this model was not ready until the competition deadline, so we could not include it in our final submission.

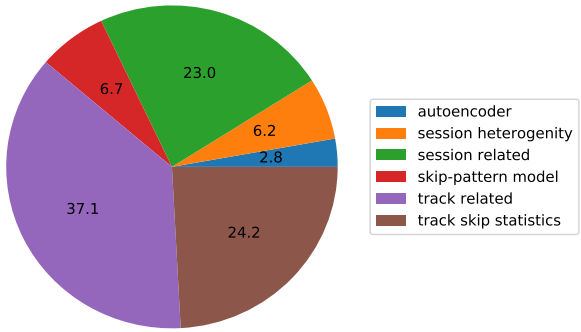

Figure 2: Category distribution of the features used in the final model. The majority of the variables were generated from raw track features (purple). Percentages are shown up to one decimal point.

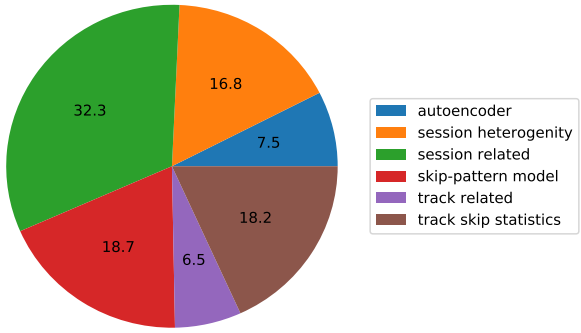

Figure 3: Distribution of the accumulated category importance in the final model. Session related (SR) and skip-pattern model (SP) features account for more than 50% of the total feature importance. Percentages are shown up to one decimal point.

4 RESULTS

Our final model is the combination of five GBT models trained on different weeks of the data. Each model has 80 trees with maximum depth 5. The rationale behind our approach was to reduce the effect of overfitting for single models by using the majority decision of 5

independent classifiers. A crucial part of our solution is the process of data sampling and feature selection by importance. In this work the importance of a given feature corresponds to the number of times it was selected by a GBT model. In order to optimize our models, we separated the public data into a training and a validation set. From each log file, we selected random 80% of the sessions for training and the remaining 20% for validation.

4.1 Training by feature reduction and sampling

The data spans approximately 9 weeks from 2018-07-15 to 2018-09-18. Due to the immense size of the data, we selected weeks and small samples for training, as we will describe next.

In the first feature selection step, we considered 5 weeks of data, July 15–22, July 30–August 5, August 13–19, August 27–September 2 and September 10–16. We sampled 200000 random sessions for each week. In each week, we trained a GBT classifier to select the 100 most important features out of the total 508.

The final models were trained independently for the five weeks. We increased the weekly sample size up to 10 million records to train GBT classifiers by using only the selected 100 features. Our skip prediction is the majority decision of the five models trained over different weeks.

4.2 Feature importance analysis

We discuss the importance of the features in our model by grouping them into six categories:

(1) **autoencoder (AE):** generated from the skip probabilities predicted by the autoencoder model (from Section 3.1).
(2) **skip-pattern model (SP):** generated from the skip probabilities and confidence values of the binary and ternary pattern based models (see Section 2.3).
(3) **session heterogeneity (SH):** see Section 2.2.
(4) **session related (SR):** extracted from raw session features.
(5) **track skip statistics (TS):** see Section 2.1.
(6) **track related (TR):** extracted from raw track features.

As discussed in Section 4.1, we selected the most important 100 features independently in each of the five weeks that we considered. The total number of variables that appear in any of our models is 178. The majority of them are raw track related (TR), track skip statistics (TS) or session related (SR) features, see Figure 2.

In Figure 3, we show the cumulative importance for each category. Session related (SR) variables account for most of the cumulative importance. However skip-pattern model (SP), session heterogeneity (SH) and autoencoder (AE) based features have a huge impact in predictive power as their proportional cumulative importance is usually almost three times the ratio they account for out of the selected 178 features (less than 7%). On the other hand, the cumulative importance of raw track related (TR) variables is only 6.5% compared to the fact that they account for more than third of the selected feature set. Finally, in Table 1 we present the 20 most important features with the sum of importance from the 5 GBT classifiers.

Table 1: The 20 most important features in our final model. Count is the total number of times the variable was selected by any of the five GBT models. Categories are autoencoder (AE), session heterogeneity (SH), session related (SR), skip-pattern model (SP), track related (TR), track skip statistics (TS).

Name	Category	Count
ternary skip ratio	SP	880
repeat count	SH	618
session position divided by session length	SR	556
autoencoder skip prediction	AE	447
unique track ratio	SH	392
repeat count stddev in first half of session	SH	382
track skip_3 ratio in second half of public sessions	TS	353
ratio of 'fwdbtn' in first half of session	SR	345
track skip_2 ratio in second half of public sessions	TS	333
min of ternary skip ratio in first half of session	SP	306
track not skipped ratio in second half of public sessions	TS	278
ratio of 'backbtn' in first half of session	SR	261
max of ternary skip ratio in first half of session	SP	251
context_type='user_collection' in first half of session	SR	226
session length	SR	195
track skip_1 ratio in second half of public sessions	TS	191
behavior='endplay' in first half of session	SR	172
binary skip ratio	SP	168
Mahalanobis distance from session mean	SH	167
behavior='clickrow' in first half of session	SR	166

5 CONCLUSIONS

We described the solution of team Definitive Turtles for the Spotify Sequential Skip Prediction Challenge, which reached 10th place on the final leaderboard. We implemented several independent prediction models and combined their output with engineered features using Gradient Boosted Tree classifiers. We achieved an MAA of 0.583 and FPA of 0.772, improving over the best baseline method reaching an MAA of 0.537. In after-work, we were also able to construct a recurrent deep network classifier that reached an MAA of 0.602. The source code for producing our final submission for the challenge is available online at https://github.com/ferencberes/wsdm-spotify-challenge-2019.

REFERENCES

[1] Brian Brost, Rishabh Mehrotra, and Tristan Jehan. 2019. The Music Streaming Sessions Dataset. In *Proceedings of the 2019 Web Conference*. ACM.
[2] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*. 2980–2988.
[3] Yoav Freund and Robert E Schapire. 1997. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *J. Comput. Syst. Sci.* 55, 1 (Aug. 1997), 119–139. https://doi.org/10.1006/jcss.1997.1504
[4] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.
[5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
[6] Kay I Penny. 1996. Appropriate critical values when testing for a single multivariate outlier by using the Mahalanobis distance. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 45, 1 (1996), 73–81.
[7] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. ACM, 1096–1103.