# HSP-TL: A Deep Metric Learning Model With Triplet Loss For Hit Song Prediction

Petros Vavaroutsos and Pantelis Vikatos
*Orfium Research*
Athens, Greece
Email: {peter, pantelis}@orfium.com

***Abstract*—The music industry is interested in the future success of a song and its presence in popular rankings such as the Billboard charts. However, a song's popularity might be impacted by variables such as music trends and social influences, which are indifferent to audio signals. In this paper, we present HSP-TL, a deep learning model, to identify likely hit songs. Our work combines temporal information and features derived from audio and lyrics to estimate the success of a recording. We adopt the concept of the triplet loss function to minimize the distance between objects with similar popularity. Also, we use convolutional neural networks on 2-D low-level audio features, contrary to the current approach. We use pre-trained language models for text-based feature extraction. Our method is evaluated on the Hit Song Prediction Dataset, which we enrich with the lyrics of each song. Our results show that the inclusion of lyrics improves song uniqueness and reflects musical trends. The proposed model outperforms the current approach by up to $8\%$.**

***Index Terms*—Hit Song Prediction, Lyrics, Deep Learning, Metric Learning**

## I. Introduction

The characteristics of a song's popularity and the emergence of a rising star are objects of interest to the music industry, which seeks to determine what causes a song's success and tailor new releases according to popular trends [1], [2]. Millions of new songs are released each year, and only a relatively small percentage of them become hits [2].

Many studies focus on predicting the success of songs in the well-known charts [3]–[5] or on the popularity of songs on streaming platforms [6], [7]. Research approaches mainly treat the problem of song popularity prediction either as a classification [2], [6], [8] or regression task [5], [9]. Most existing studies also include factors independent of sound recordings, such as social media presence [9], [10] or user behavior [11]. In terms of indicating popularity, some approaches use the presence of the song on Spotify [6], [7]. Particularly, Martín-Gutiérrez et al. [6] introduce an auto-encoder to compress high-level features consisting of audio, text, and metadata to rank tracks based on their popularity on Spotify. In addition, some studies attempt to model user behavior and extract indicators through users' activity. Demetriou et al. [12] conduct a user study to identify the most influential factors that contribute to users liking or disliking a song. The experiments in [11] demonstrate the correlation between song popularity and changes in user behavior over time and provide a predictive model for the next month's music trends using an LSTM network. However, most recent work [5], [13] use fundamental audio features as predictors to explain hit songs. In particular, the study [5] presents a model for Billboard ranking that uses low & high-level features of songs from the Million Song Dataset [14]. Alternatively, other efforts exploit the use of song lyrics to model the predictability of hits [6], [15]. Effort [16] introduces ranking loss for hit song prediction. Specifically, they employ a commercial dataset with daily play counts to train a CNN network with Euclidean loss and pairwise ranking loss in order to learn from high- and low-level audio features the relative ranking relationships between songs.

Our work uses audio-based features in combination with attributes generated by song lyrics using pre-trained language models to enlarge the predictive factors which affect the success of a recording in terms of Billboard ranking. We propose an extension of the deep-learning architecture of [5], regarding the processing of low-level audio features using convolutional neural networks, which contributes to dimensionality reduction and increases predictive performance. We conduct a comprehensive analysis of the contributions of different features to the prediction performance, revealing the importance of lyrics-related features and the potential of incorporating them with audio features in hit song prediction models. We also adopt the concept of metric learning using triplet loss [17], [18] to enhance the separability of the points of hits and non-hits in the produced embedding space. Compared to the work of [5], our approach achieves an $8.51\%$ improvement in accuracy and a $7.22$ reduction in RMSE. Finally, our study contributes to the extension of Hit Prediction Dataset [5] by adding song lyrics embeddings for hit song prediction benchmarking. For reproducibility, we make our dataset publicly available[1].

The rest of the paper is structured as follows. Section II describes the process of data creation and the feature sets used for the model. Section III contains the model overview, presenting the details of the model layers. Section IV contains a reference to our experimental setup, the implementation of the model, and a discussion of its strengths and limitations. Additionally, we provide an analysis of the performance of the proposed model in comparison to previous approaches. Finally, in section V we provide an overview of the results and contributions and conclude with an outlook on future work.

[1]https://github.com/Orfium/hsp-lyrics-dataset

## II. Song Features & Dataset Overview

A recent study [5] provides the Hit Song Prediction Dataset, which contains the release year and low & high-level audio-based features for $95,067$ songs divided into 2 classes of $5,932$ hits, along with their respective billboard rank, and $89,235$ non-hits. We augment this dataset with features derived from the lyrics of each sound recording. We use the Genius[2] database, which contains a collection of approximately $25$ million songs and lyrics, and we query its API using the song titles and artist names for the $95,067$ cases. Since Genius returns multiple results per query, we validate them using a BERT-based text matching approach [19]. We match the query metadata with the metadata of each result and keep the pair with the highest score. From the $95,067$ cases we retrieved $11,634$ lyrics. The final dataset contains $5,962$ non-hits and $5,672$ hits with three feature groups: audio-based features ($512$ low & $50$ high level features computed by [5]), temporal features (release year) and text-based features ($768$ BERT-embeddings from lyrics).

Low-level (LL) features result from computing descriptors that characterize the overall loudness, the dynamics and spectral shape of the signal, the rhythm (including beat positions and BPM value), and the tonal information (including chroma features, keys, and scales) [20]. High-level (HL) features such as mood, genre, gender of voice etc. are computed by pre-trained models from the Essentia Framework [21] using a $30$ second song's audio segment. We normalize numerical data (e.g. average loudness) to $[0, 1]$ using min-max normalization and we convert categorical data (e.g. the chords key, scale) to one-hot vector representation.

Regarding text-based features, we conclude that our dataset includes $11.51\%$ non-English lyrics by using automatic language detection[3]. Thus, we use the Multilingual BERT, which is a variation of BERT [22] but trained in $104$ different languages. We feed the BERT model with the lyrics to generate the third set of features.

## III. Model Overview

The core idea of our contribution is to characterize recordings using metric learning and a combination of audio features, text features extracted from song lyrics, and temporal information. We base our model architecture on the work of Zangerle et al. [5], which uses low & high-level audio features and release year to estimate hits on Billboard. However, this study does not include information from lyrics, which is an important factor as Demetriou et al. [12] claims. We introduce the lyrics features using the pre-trained language model of BERT [22]. In addition, instead of flattening the low-level 2-D features as in [5], we employ convolutional layers as part of our architecture. Furthermore, we enforce the separation of hit & non-hits in the embedding space through metric learning using triplet loss [17].

[2]https://genius.com/

[3]https://spacy.io

We propose the model HSP-TL, whose architecture is shown in Figure 1. Step $1$ of the model is the process of vectorizing the data features. High-level audio features are represented by a probability value in the range of $[0, 1]$ corresponding to the likelihood of the feature (e.g., $0.9$ rock). The release year of the song is normalized between $[0, 1]$ as mentioned before. Therefore, both are simply concatenated with the feature vector in step 2. Low-level features consist of two-dimensional data of size $(13, 13)$ (e.g. the MFCCs of the audio data), one-dimensional data (e.g. melbands), and single values (e.g. overall loudness in dB).

Study [5] introduces the concept of Deep Part which handles the low-level features. We extend Deep Part, by using the architecture shown in Figure 2. The Deep Part is responsible for the dimensionality reduction of the data to single values. These values are concatenated with the feature vector in step 2. In particular, it contains a Convolutional Neural Network (CNN) that processes two-dimensional data and consists of 3 layers. The first is a 2-D convolutional layer with a kernel size of $(3, 3)$ and ReLU activation, followed by 2-D max-pooling with a $(2, 2)$ kernel and a dropout layer with probability $p = 0.2$. It accepts a single channel of two-dimensional data and outputs 32 channels. The second layer is again a 2-D convolutional layer with kernel size $(3, 3)$ and ReLU activation, followed by 2-D max-pooling with $(2, 2)$ kernel and a dropout layer with probability $p = 0.2$. It accepts $32$ channels and outputs $64$ channels. The third layer is a fully connected layer with ReLU activation, that accepts the output of the last max-pooling layer, flattened, and outputs a single value. Each 1-D feature $i$ of size $N_i$ is assigned a fully connected layer with input dimension $N_i$ and outputs a single value.

We feed the language model with the lyrics of each song. We tokenize the raw text in the appropriate form for the pre-trained BERT model, including truncation and max padding to $512$ in length, and pass it through BERT. We extract the embedding vector of the last BERT layer and perform a pooling procedure by selecting the embedding vector of the first token, i.e., the *CLS* token, resulting in a one-dimensional vector of $768$ length, followed by a dropout layer with probability $p = 0.1$. This vector represents the lyrics features for a single song. In step 2, the concatenation layer combines the $4$ feature sets, that include $50$ high-level and $512$ low-level audio features, $1$ release year feature, and $768$ text-based features, aggregating $1,331$ features. In step 3, we add two consecutive layers that include a dense layer with batch normalization and a ReLU activation function and both output the same size as the concatenation layer. In step $4$, we pass the output of step $3$ to a fully connected layer, without an activation function, which outputs a vector length of $64$, corresponding to the embedding of the song. The model architecture ends with steps $5$ and $6$, which illustrate metric learning using triplet loss, calculated as follows:

$$loss = \max\{d(x_a, x_p) - d(x_a, x_n) + \gamma, 0\} \tag{1}$$

where $\gamma$ is a margin parameter, $x_a$, $x_p$, $x_n$ are the anchor,

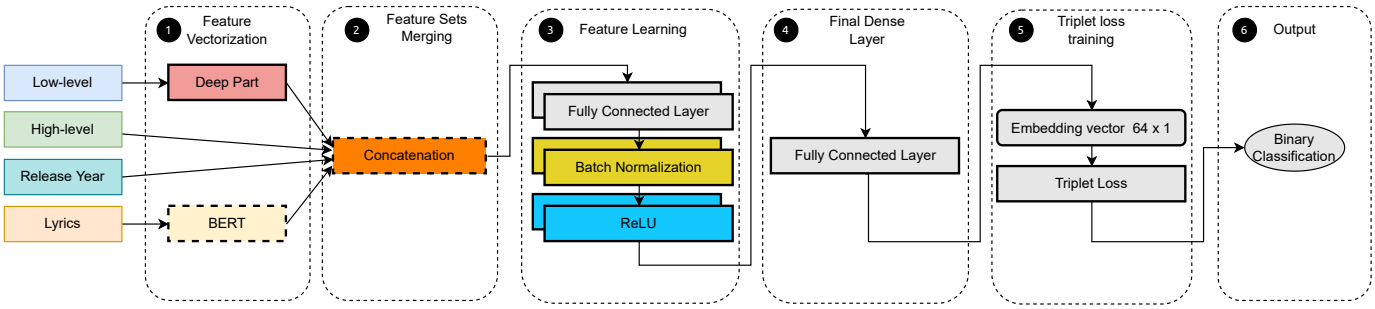


Fig. 1. The general block schema of the deep network architecture of the HSP-TL model outlining the principal functionalities and data components; the dashed line rectangles represent non-trainable layers.

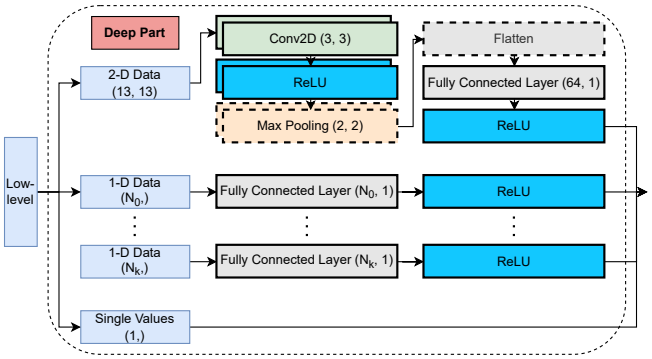


Fig. 2. The deep network architecture employed for the module Deep Part; the dashed line rectangles represent non-trainable layers.

positive and negative embeddings, respectively, and $d$ is a distance function, which in our case is the Euclidean distance.

## IV. Experiments and Results

### A. Experimental Setup

We evaluate the replicated (in PyTorch) model described in [5] and compare it with two proposed models, which are modified versions of HSP-TL architecture. In the first model, HSP-Linear-reg, we treat the 2-D features as 1-D by flattening them. We then assign a fully connected layer to each flattened feature and remove the CNN entirely from the Deep Part. We also modify the fully connected layer from step 4 to output a single value corresponding to the regression value. Our second model is the HSP-reg, in which we only modify the fully connected layer of step 4 to output a single value corresponding to the Billboard rank. All models use the loss function MSE (step 5) for training. Also, we conduct experiments on the performance of the combinations of the HL, LL, and lyrics feature sets for the HSP-reg model.

Furthermore, we test alternative loss functions and adapt metric learning to address the binary classification problem of hits and non-hits having all the features (HL+LL+lyrics). Binary Cross Entropy Loss (BCE) and Triplet Loss (TL) form two different models, HSP-bce and HSP-TL, respectively. The HSP-bce model uses the same architecture of HSP-TL in steps 1 - 3, as shown in Figure 1. We modify the fully connected layer of step 4 to output a single value corresponding to each class (hit, non-hit) and use the BCE as a loss function in the training phase. To make the training of the HSP-TL model more robust, we use hard sampling. Hard sampling refers to selecting triplets that are hard to classify correctly, i.e., triplets where the distance (Euclidean) between the anchor and the negative is very close to the distance between the anchor and the positive. By selecting hard triplets, the model is forced to learn more discriminative embeddings that can better separate data points from different classes [23]. We evaluate the classification capabilities of the HSP-TL network by extracting the embeddings of the training set and the testing set and using them as the database and query set, respectively. For each query, we compute the Euclidean distance from all database embeddings and keep the $k$ with the lowest distance. For these $k$ embeddings, we choose their dominant class as our predicted class. After experimentation, we conclude that $k = 11$ is the best value.

We use 10-fold cross-validation to evaluate the quality of the predictive models. For the baseline model [5] and {HSP-Linear, HSP}-reg models, the prediction reflects the rank in Billboard, ranging from 1 to 100. Same as [5], we assign the value of 150 to the songs that never made the Billboard list, corresponding to non-hits. We evaluate the performance of our model using the task-related metrics of Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) and we use the threshold of 100 to report the accuracy scores of each model and see how they perform in discriminating between hits and non-hits.

All models are implemented in Python using the PyTorch framework[4] and the Transformers library[5]. The maximum sequence length of the BERT tokenizer is set to 512, with truncation and maximum padding activated. The learning rate is set to $10^{-3}$. For triplet loss, we choose a margin of $\gamma = 0.1$ and for hard triplets selection, we use cosine distance, while for computing our database and queries during evaluation we use Euclidean distance. We choose a batch size of 64 and 100 epochs for training, with early stopping at 10 epochs. We store the respective checkpoint with the lowest loss value in

[4]PyTorch: https://pytorch.org/
[5]Transformers: https://huggingface.co/transformers/

| Model | Evaluation Metrics | | |
|---|---|---|---|
| | Accuracy | RMSE | MAE |
| Zangerle et al. [5] | 71.71% | 58.35 | 50.03 |
| HSP-Linear-reg | 77.04% | 53.21 | 43.06 |
| **HSP-reg** | **79.14%** | **51.13** | **38.98** |

TABLE I
RESULTS FOR HIGHEST RANK PREDICTION ON THREE DIFFERENT ARCHITECTURES; THE BEST RESULTS ARE PRINTED IN BOLD FONT.

| Feature Class | Evaluation Metrics | | |
|---|---|---|---|
| | Accuracy | RMSE | MAE |
| HL | 68.80% | 60.27 | 51.21 |
| LL | 62.49% | 66.45 | 57.65 |
| HL+LL [5] | 72.19% | 58.57 | 50.83 |
| Lyrics | 72.98% | 57.21 | 45.17 |
| HL+Lyrics | 77.92% | 53.07 | 39.91 |
| LL+Lyrics | 75.81% | 55.81 | 43.03 |
| **HL+LL+Lyrics** | **79.14%** | **51.13** | **38.98** |

TABLE II
RESULTS FOR EACH FEATURE CLASS, USING HSP-REG ARCHITECTURE; THE BEST RESULTS ARE PRINTED IN BOLD FONT.

the validation set. As an optimizer, we employ the adaptive learning rate optimization algorithm, Adam.

### B. Results and Discussion

The results in Table I show that the addition of text-based features and our model modifications for LL features improves previous work [5]. In particular, the HSP-reg model achieves the highest value of 0.79 accuracy and the lowest RMSE and MAE values of 51.13 and 38.98, respectively.

We conclude that enriching the feature vector with multiple types of sound recording features is beneficial for prediction performance as Table II presents. It is worth noting that the HL audio features, which reduce the recordings to a specific mix of categories, are highlighted as more important factors than the LL features. However, training exclusively with song lyrics outperforms the exclusive use of audio-based features. This suggests that the features generated from the song lyrics have a positive effect on prediction and demonstrates the effectiveness of the transformer architecture. We should note that the combination of song lyrics and HL audio features is only slightly less accurate than the combination of all features, indicating that this combination is sufficient to characterize the success of a song.

We also note that the RMSE and MAE gradually decrease and reach the maximum difference of 15% and 18% respectively, when we add the full set of features. However, the observed error measures in the regression are still high, suggesting that the model does not have the predictive power to estimate the exact Billboard rank. We consider that the choice of the target value of non-hits to 150 affects the error bounds of the regression model.

The use of different loss functions not only improves performance but also shows that metric learning effects beneficially to the estimation of hit songs, as Table III describes. Both alternative loss functions (BCE and MSE) improve the accuracy of the model, with the triplet loss having the highest value of 0.80. More specifically, the prediction of the non-hits class is more precise than the prediction of the hits class for all models and ranges from 0.81 to 0.87. On the other hand, the recall values for hits are higher than those for non-hits and vary between 0.82 and 0.89. The results using triplet loss show that the points of hits and non-hits in the produced embedding space effects efficiently the separability of the two classes.

We have to acknowledge that this research work is limited to the audio and text features of a song, which are indifferent to external variables such as music trends, the historical presence of certain artists in the music industry, or marketing promotion by record labels. However, we believe that this is a fair way to evaluate new releases without external knowledge. We should note that hits represent the outliers in the population of songs and therefore any dataset for this task suffers from imbalance. In addition, the dataset is skewed towards U.S. and U.K. tracks that dominate the Billboard charts. The dataset also includes songs released through 2011 and therefore does not reflect today's trends and culture. In addition, the study [5] does not report the exact data split used to train its models, which explains for the inconsistency of the results.

| Model | Evaluation Metrics | | |
|---|---|---|---|
| | Accuracy | Precision | Recall |
| HSP-reg | 79.14% | 73% / 87% | 89% / 69% |
| HSP-bce | 79.05% | 74% / 85% | 87% / 72% |
| **HSP-TL** | **80.22%** | 79% / 81% | 82% / 78% |

TABLE III
BINARY CLASSIFICATION; RED REPRESENTS THE CLASS OF NON-HITS, BLUE REPRESENTS THE CLASS OF HITS.

## V. CONCLUSIONS & FUTURE WORK

In this paper, we propose a novel deep learning model, namely HSP-TL, which combines song lyrics with descriptive audio features and temporal information to predict hit songs. Our results show that the proposed model performs best compared to state-of-the-art deep networks, on the Hit Song Prediction Dataset. We show that the inclusion of lyrical context enhances song uniqueness and reflects musical tendencies. We intend to improve the feature extraction by fine-tuning the language model specifically for song lyrics and using updated techniques for higher-quality audio features. We aim to update the existing dataset with newly released hits to test the model's efficiency on current music trends. Furthermore, we plan to explore the impact of various elements, including the social media presence and/or the profile of the artist, on the overall success of the song.

## REFERENCES


[1] F. Pachet and P. Roy, "Hit song science is not yet a science." in *International Society for Music Information Retrieval Conference (ISMIR)*, 2008, pp. 355–360.

[2] D. Herremans, D. Martens, and K. Sörensen, "Dance hit song prediction," *Journal of New Music Research*, vol. 43, no. 3, pp. 291–302, 2014.

[3] A. H. Raza and K. Nanath, "Predicting a hit song with machine learning: Is there an apriori secret formula?" in *International Conference on Data Science, Artificial Intelligence, and Business Analytics (DATABIA)*. IEEE, 2020, pp. 111–116.

[4] M. Interiano, K. Kazemi, L. Wang, J. Yang, Z. Yu, and N. L. Komarova, "Musical trends and predictability of success in contemporary songs in and out of the top charts," *Royal Society open science*, vol. 5, no. 5, p. 171274, 2018.

[5] E. Zangerle, R. Huber, M. Vötter, and Y.-H. Yang, "Hit Song Prediction: Leveraging Low- and High-Level Audio Features," in *Proceedings of the 20th International Society for Music Information Retrieval Conference (ISMIR)*, 2019.

[6] D. Martín-Gutiérrez, G. H. Peñaloza, A. Belmonte-Hernández, and F. Á. García, "A multimodal end-to-end deep learning architecture for music popularity prediction," *IEEE Access*, vol. 8, pp. 39 361–39 374, 2020.

[7] C. V. S. Araujo, "A model for predicting music popularity on spotify," *Recall*, vol. 50, pp. 173–90, 2020.

[8] R. Dhanaraj and B. Logan, "Automatic prediction of hit songs." in *Proceedings of the 6th International Society for Music Information Retrieval Conference (ISMIR)*, 2005, pp. 488–491.

[9] E. Zangerle, M. Pichl, B. Hupfauf, and G. Specht, "Can microblogs predict music charts? an analysis of the relationship between #nowplaying tweets and music charts." in *Proceedings of the 17th International Society for Music Information Retrieval Conference (ISMIR)*, 2016, pp. 365–371.

[10] E. Tsiara and C. Tjortjis, "Using twitter to predict chart position for songs," in *International Conference on Artificial Intelligence Applications and Innovations (AIAI)*, 2020, pp. 62–72.

[11] K. Li, M. Li, Y. Li, and M. Lin, "Lstm-rpa: A simple but effective long sequence prediction algorithm for music popularity prediction," *arXiv preprint arXiv:2110.15790*, 2021.

[12] A. M. Demetriou, A. Jansson, A. Kumar, and R. M. Bittner, "Vocals in music matter: the relevance of vocals in the minds of listeners," in *Proceedings of the 19th International Society for Music Information Retrieval Conference (ISMIR)*, 2018.

[13] L.-C. Yang, S.-Y. Chou, J.-Y. Liu, Y.-H. Yang, and Y.-A. Chen, "Revisiting the problem of audio-based hit song prediction using convolutional neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 621–625.

[14] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[15] A. Singhi and D. G. Brown, "Can song lyrics predict hits," in *Proceedings of the 11th International Symposium on Computer Music Multidisciplinary Research*, 2015, pp. 457–471.

[16] L.-C. Yu, Y.-H. Yang, Y.-N. Hung, and Y.-A. Chen, "Hit song prediction for pop music by siamese cnn with ranking loss," *arXiv preprint arXiv:1710.10814*, 2017.

[17] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International workshop on similarity-based pattern recognition*. Springer, 2015, pp. 84–92.

[18] J.-C. Wang, J. B. L. Smith, W.-T. Lu, and X. Song, "Supervised Metric Learning For Music Structure Features," in *Proceedings of the 22nd International Society for Music Information Retrieval Conference (ISMIR)*, Nov. 2021, pp. 730–737.

[19] N. Katakis and P. Vikatos, "Entity linking of sound recordings and compositions with pre-trained language models." in *WEBIST*, 2021, pp. 474–481.

[20] A. Porter, D. Bogdanov, R. Kaye, R. Tsukanov, and X. Serra, "Acousticbrainz: A community platform for gathering music information obtained from audio," in *16th International Society for Music Information Retrieval Conference*, 2015.

[21] D. Bogdanov, N. Wack, E. Gómez, S. Gulati, P. Herrera, O. Mayor, G. Roma, J. Salamon, J. Zapata, and X. Serra, "Essentia: an audio analysis library for music information retrieval," in *Proceedings of the 14th International Society for Music Information Retrieval Conference (ISMIR)*, Nov. 2013.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[23] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.