Optimizing Television Content for Video on Demand, Leveraging Image Retrieval to Detect Opening/Closing Credits, Recaps, Bumpers and Previews in TV Shows´ Video Files

Master Thesis

Niels ten Boom
s4767314

Contents

1 Introduction

Viewing rates for TV are dropping gradually while the user counts of streaming services such as Netflix and Videoland are growing year over year. This shift from TV to Video on Demand introduces new problems for broadcasters (in this case RTL), as they now have to support a hybrid form of traditional broadcasting and Video On Demand (VOD). RTL does this by putting their content on Videoland after it has aired on TV.

Video content is optimized for just one form, and currently that is TV. This means that the video content contains recaps, opening credits, bumpers (to ease a viewer into commercials) and closing credits. A viewer watching this content back-to-back may find it preferable to be able to skip these recurring segments to improve their viewing experience. Providing this functionality to users, requires metadata on where the skippable segments occur in the videos. For a large percentage of their content, such metadata has not been retained. Videoland currently hosts over 1000 different titles consisting of multiple seasons and episodes and this selection is prone to change. An automated solution that can detect these segments in videos would be very useful to solve that problem and transform the video content into a format suited for VOD. In this thesis we will explore the best methods for such an automated solution.

There is little margin for error with this solution. The accuracy of the end result should be sufficiently high enough for it to be trusted to label all of the content automatically. If that is not the case then it needs double checking by a human. Because if a part of the video gets mislabeled, a user may skip over actual content. It is critical to RTL that this does not happen.

This research presents my findings on detecting recaps, opening credits, bumpers, closing credits and previews unsupervised given the video files from a TV-show. We are going to attempt to tackle this problem by using image retrieval techniques. The motivation behind this is that the general characteristic of the segments is that they reoccur. To be able to detect reoccurring parts we are going to compare similarities of frames, this should be done efficiently and accurately, both methods are important in image retrieval research. Our main research question then reads:

> ***Is it possible to leverage image retrieval to accurately detect recaps, opening credits, bumpers, closing credits and previews in video files from a TV-show?***

With the subquestions:

> ***What accuracy does each image retrieval approach achieve?***
>
> ***How efficient is each method?***

This thesis aims to answer aforementioned research questions. The rest of this section will be used to expand on the different type of video classes which this research is focused on. Section 2 explains all of the related work. In section **??** the data is presented. Section 3 expands on the methodology and then the produced results are

presented in section 4. Lastly, the results are discussed in section 5.

1.1 Recurring Segment Classes

In this section we will give a background on the different types of segments to be detected. What all the segment classes have in common is that they reappear (partially) in previous or future episodes. We will use this attribute for the detection.

1.1.1 Closing Credits

The closing credits at the end of a video contain a lot of scrolling text most of the time and there is little variation between the frames. In these texts everybody related to the production of the video is mentioned. In general the frames all have a black background with white text. But backgrounds can also vary as can be seen in figure 1, therefore a solution that simply detects black and white would not suffice.

Figure 1: Examples of varying types of closing credits

1.1.2 Opening credits

The opening credits of a TV-show are generally the same for all of the episodes in the same season. It opens the show with a theme song, presenting the most important actors at the start. If the sequence were known then it would be a rather simple problem to solve, by matching this sequence with all of the videos to locate the opening credits. The difficult part here is that they start somewhere at the start of a video, never right at the start. There is also no prior knowledge on how the opening credits of a show look like and they often change every season.

1.1.3 Recaps

A television show may contain recaps before the opening credits. In the recaps content of previous episodes is repeated to refresh the viewer's memory. This is useful for linear TV when an episode is aired every week. A binge watcher ideally wants to skip the recaps together with the opening credits because they have seen the content recently. Not all material preceding the opening credits is a recap though, sometimes it is original content. Classifying whether the video part before the opening credits consists of recaps is important for a fully automated solution.

1.1.4 Bumpers

The bumper is a part of the video that eases a viewer into the upcoming commercial (see figure 2). Most of the times it has a voice over and text on screen saying: 'Next' (or the dutch translation of 'next') and some footage of what is to be expected after the commercial break is shown.

Figure 2: Example 'Next' bumper of the dutch television show Expeditie Robinson.

1.1.5 Previews

A preview is a segment at the end of an episode where an advance showing of fragments of the next episode(s) are played. Previews can typically be found in content that was created specially for broadcast TV. It gives the viewer a taste of what to expect in the next episode that will air a week later. However, this part is not of interest to a binge watcher.

1.2 Definitions and Techniques

This section will be used to elaborate on some topics mentioned in this thesis.

We define a season with multiple episodes of a TV-show as T.

$$T = \{E_1, E_2, \ldots, E_x\}$$

An episode within a season is defined as a set of frames.

$$E = \{f_1, f_2, \ldots, f_x\}$$

1.2.1 Image Retrieval

Image retrieval is a subset of the research field Information Retrieval (IR). Explained briefly, it investigates the problem where one has a query **q** expressing an information need and wants to find the best possible match for this **q** in a set of documents **D**.

In the case of image retrieval the query consists of an image for which the best matching image should be found in a document set of images. There are two approaches for doing this.

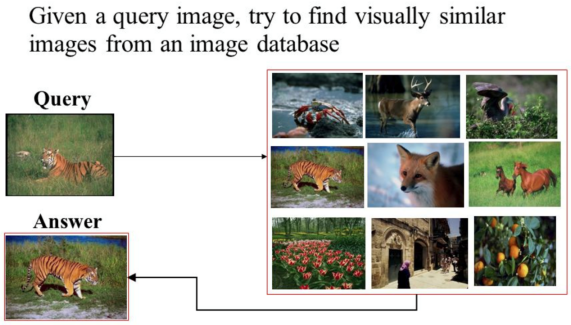

Figure 3: Visual example of image retrieval.

Text-based image retrieval refers to retrieval of images based on textual metadata associated with these images. It matches images based on for instance their titles, keywords and more. The downside of this method is that if there is no metadata available then it needs to be added manually. With the growing sizes of image databases this can become quite inefficient, hence the more focus on content-based image retrieval in past years [1].

Content-based image retrieval is retrieval based on the content of the image rather than the metadata as is the case with concept-based image retrieval. Computer vision is used to evaluate the image similarity, this can be done by looking at colors, shapes, textures and more. The advantage of content based image retrieval is that it does not rely on the metadata of images and thus does not require manual labeling.

1.2.2 Feature vectors

A feature vector is a vector containing a numeric representation of multiple characteristics of an object. In this case the objects will be images, the frames of a video. The explanation of feature vectors for image data is not very intuitive. A more simple and intuitive example would be to construct a feature vector to represent a person. The features could be age and length in centimeters, so that the resulting feature vector will look like $\mathbf{x} = [age, length]^T$. This vector can now be represented in 2D-space, and compared to other persons in 2D space by calculating the distance between other points in that space. The distance is computed by calculating the euclidean distance between each vector.

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1 - q_1)^2 + \ldots + (p_n - q_n)^2}$$

In figure 4 we give an example of three different representations. According to these features the 25 and 35 year old are the most alike because the distance between them is the smallest.

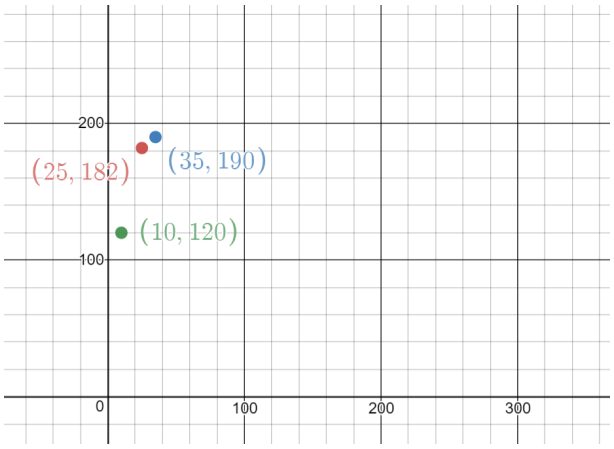

Figure 4: Example of distance in 2D between feature vectors that represent persons. According to the chosen features, the red and blue person are more alike. Image feature vectors will consist of many more dimensions but this cannot be visualized.

Extending this analogy to images. The similarity of images can be computed by computing the distance between their feature vectors. A low distance indicates a higher similarity. Many research in image retrieval has been done on the construction of these feature vectors for the problem described in section 1. Part of this research is choosing the best method for the construction of the feature vectors. We will expand on the methods that are going to be explored.

Color histograms

A color histogram is a representation of the distribution of colors in an image. Color histograms are a flexible and low dimensional way of representing images. A color histogram can be computed by counting the number of pixels in a certain color range, the size of this range is variable, called the bin size. The higher the bin size, the lower the dimensions of the resulting histogram. For example, if one chooses a bin size that is half of the intensity range. The resulting vector would have 6 dimensions, two bins for each color channel. See figure 5 for an example with such a bin size.

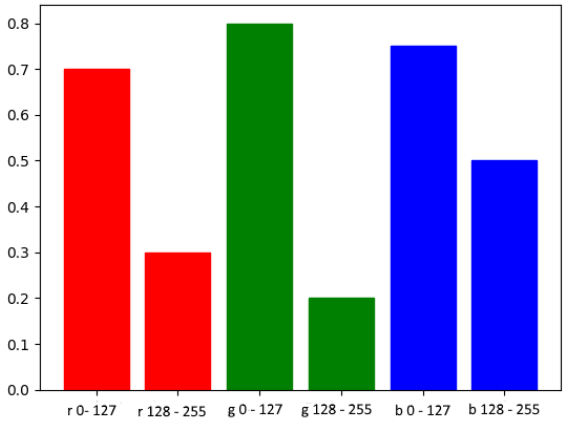

Figure 5: Color histogram with a large binsize (binsize=128) for illustration.

Color histograms are going to be used to compute image similarity to match similar frames later in this thesis. b

SIFT
valt weg hoogstwaarschijnlijk

Convolutional Neural Network Features
Convolutional neural networks are very good at image classification and segmentation tasks. However it was found that using the resulting channels after the convolution layers as feature vectors also perform well at image retrieval tasks. This will be expanded upon in section 2.

1.2.3 Shot Change Detection

Shot detection is a technique that can be used on videos to determine shot boundaries, see figure 6 for an example of such a shot change.

Figure 6: Example of a shot change within 4 frames of a video.

Shot detection is going to be used and thus will we expand on it. A lot of research has been done related to shot detection [2] and many techniques have been proposed.

This thesis used a method that looks at the shift in mean color in HSV space [3], see figure 7 for a visual depiction of HSV color space. HSV (Hue, Saturation, Value) color space is a more intuitive color mixing model compared to RGB (Red, Green, Blue) color space. Because one can change a value in either of the three values in HSV and expect what the new color is going to look like. This is almost impossible for RGB because for this same color change to happen, you need to change all three red, green and blue values to result in the same color space.

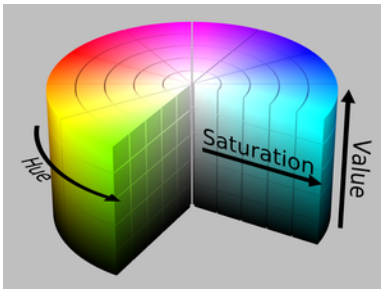

Figure 7: The HSV color space.

The shot boundaries are classified by calculating the difference in hue, saturation and value for each frame and then from this the mean is calculated. If the mean is higher than a threshold H, then a shot boundary is marked, refer to algorithm 1 for a pseudo-code representation.

Algorithm 1: Shot boundary detection

```
shotBoundaries = List();
previousMeanHSV = getMeanHSV(Episode[0]);
for frame in Episode do
    meanHSV = getMeanHSV(frame);
    if abs(meanHSV - previousMeanHSV) > H then
        shotBoundaries.add(frame);
    end
    previousMeanHSV = meanHSV;
end
```

2 Related Work

No prior work exists that explores a solution for the problem previously described (automatic labeling of segments). A lot of related work exists that focuses on commercial or repeated sequence detection in large broadcast streams. These detection methods can be divided into three groups: fingerprint, feature-based and unsupervised methods.

Fingerprint methods set up a database of fingerprints of known commercials or repeated sequences to detect these in a broadcast stream. Lienhart et al. [4] propose a method based on features to roughly localize advertisements in a stream and then construct a fingerprint database based on color coherence vectors. Gauch et al. [5] propose a method based on the color moments in a stream. Covell et al. [6] implement a fingerprinting method based on audio with visual verification after a proposed match. The disadvantage of these fingerprinting methods is the setting up and maintenance of such a fingerprint database.

The feature-based methods detect commercials based on extracted video features. Wang et al. [7] fuse the results of audio scene changes and textual content similarity between shots to segment programs including commercials.

With the unsupervised methods the authors typically do a dimensionality reduction operation and then try to find repeated sequences or commercials with clustering methods. Herley et al. [8] convert the stream with a Discrete Cosine Transform (DCT) for dimensionality reduction and then propose an extensive probability framework to detect repeated sequences. Benezeth et al. [9] use the Electronic Programme Guide (EPG) in addition to the dimensionality reduction to detect program boundaries.

Abduraman et al. [10] propose a system to detect repeated sequences in streams by performing a DCT operation on all of the frames in the stream and then use a micro clustering technique to detect repeated sequences. They were also able to link the trailers to their respective programs that occur at a later point in the stream.

All of the previously mentioned works in this section never cover the specific topic of segmenting the classes mentioned in section 1.1. The methods yielded high accuracies in the range of 90% - 95% precision. This work aims for higher accuracies and thus will not be replicating most of the previously mentioned methodologies, however from the aforementioned papers we conclude that a large dimensionality reduction step will be necessary to efficiently process a large dataset of videos.

Dimensionality reduction and efficient matching of large amounts of visual data is typically used within content based image retrieval, Zheng et al. give a very detailed summary of all the significant contributions from past years [11]. Their summary covers three periods in image retrieval: Early methods, SIFT-based methods and CNN-based methods. Smeulders et al. present all the contributions of the early methods [12], these methods focused on looking at the color, texture and local geometry of images for retrieval. Not much later the Bag-of-words (BoW) model was proposed as a new method for image retrieval [13]. The advantage of this is that inverted indexes can be used for immediate retrieval of similar images. The BoW model paired with SIFT-descriptors [14] as the feature vectors was used in image retrieval research for years [15, 16, 17, 18, 19]. Since 2012 when the convolutional neural network was introduced [20] research switched to CNN-based methods for image retrieval because they achieved better performance on several image retrieval tasks [21, 22, 23].

3 Methodology

We want to explore whether the problem described in section 1 is solvable and if so, which method scores the best in terms of accuracy and efficiency. The characteristic of all the segment classes is that they reoccur either in previous or future episodes. To match frames from one episode with other episodes, feature vectors will be constructed from the frames and the distances between these vectors will be computed and saved. If a part of the video consecutively has near zero distant similar feature vectors from previous or future episodes, then that part is labeled as a recurring segment. It is therefore important that the resulting feature vectors have a clear distinction between the distances of actual recurring content and content that looks alike.

3.1 Data

The data set contains 70 video files originating from 16 differing seasons of tv-shows. Of most seasons only the first three episodes will be looked at. It can be argued that this amount of data is on the low side, however we expect that this amount will be sufficient to decide whether one of the methods is feasible to deploy on a larger scale. Originally these files were in .mxf full broadcast format, this meant each file being 25GB on average. All the files were converted to 1920p .mp4 files with FFmpeg [24], resulting in each file being 1 GB on average.

For each file the start and end timestamps for the recaps, opening credits, closing credits and previews were annotated in the HH:MM:SS format in a CSV file, so that it could be loaded effectively.

3.2 Feature Vector Construction

Video usually plays at 25 frames per second, frames very close to each other should only have very slight variations. That is why frames every 5 frames or just keyframes will be taken into account to drastically reduce the computing complexity. We take all the video files of a season $E_x \in T$ and convert each file to a set of feature vectors $S_x = \{v_1, v_2, \ldots, v_l\}$ with the function f.

$$f(E_x) = S_x = \{v_1, v_2, \ldots, v_l\}$$

The rest of the subsections will expand on different implementations of the function f, how the different types of feature vectors are constructed.

3.2.1 Color Histograms

For the construction of the color histograms the pixels intensities in each channel (Red, Green, Blue (RGB)) are counted and binned according to a specified bin size. All these bins will be concatenated to result in a feature vector, the size of the feature vector is a result of the chosen bin size.

3.2.2 SIFT

TODO

3.2.3 CNN Features

TODO

3.3 Matching

All the resulting lists of feature vectors will be added to the index of an empty Faiss instance. Faiss is a library for efficient similarity search of dense vectors [25, 26] and thus perfectly suited for our task. This library will efficiently handle building the inverted index and nearest neighbor matching.

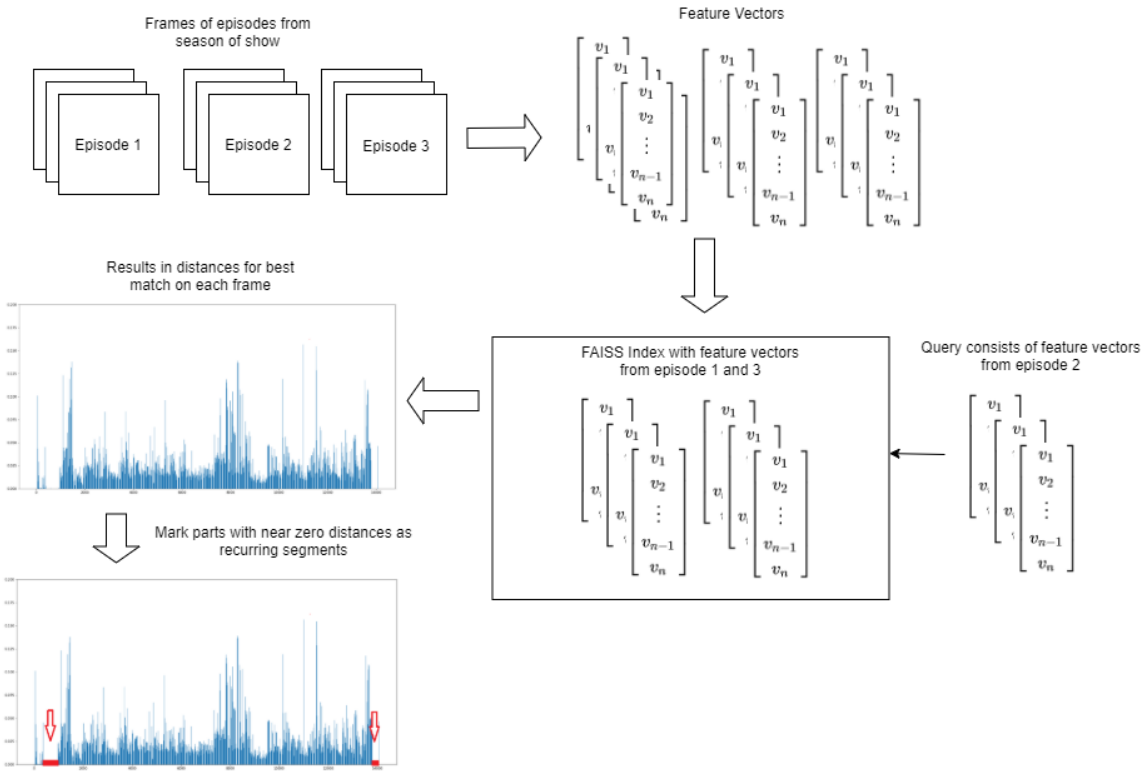

Figure 8: Diagram of the described methodology. The segment detection process for episode 2 is portrayed.

4 Results

5 Discussion

I did not do an extensive study on all of the TV shows, so other edge cases are possible in a larger amount of data.

6 References

[1] F. Rajam and S. Valli, "A survey on content based image retrieval," *Life Science Journal*, vol. 10, no. 2, pp. 2475–2487, 2013.

[2] R. W. Lienhart, "Comparison of automatic shot boundary detection algorithms," in *Storage and Retrieval for Image and Video Databases VII*, vol. 3656, pp. 290–302, International Society for Optics and Photonics, 1998.

[3] H. Shao, Y. Qu, and W. Cui, "Shot boundary detection algorithm based on hsv histogram and hog feature," in *5th International Conference on Advanced Engineering Materials and Technology*, pp. 951–957, 2015.

[4] R. Lienhart, C. Kuhmünch, and W. Effelsberg, "On the detection and recognition of television commercials," 1997.

[5] J. M. Gauch and A. Shivadas, "Finding and identifying unknown commercials using repeated video sequence detection," *Computer Vision and Image Understanding*, vol. 103, no. 1, pp. 80–88, 2006.

[6] M. Covell, S. Baluja, and M. Fink, "Advertisement detection and replacement using acoustic and visual repetition," in *Multimedia Signal Processing, 2006 IEEE 8th workshop on*, pp. 461–466, IEEE, 2006.

[7] J. Wang, L. Duan, Q. Liu, H. Lu, and J. S. Jin, "A multimodal scheme for program segmentation and representation in broadcast video streams," *IEEE Transactions on Multimedia*, vol. 10, no. 3, pp. 393–408, 2008.

[8] C. Herley, "Argos: Automatically extracting repeating objects from multimedia streams," *IEEE Transactions on multimedia*, vol. 8, no. 1, pp. 115–129, 2006.

[9] Y. Benezeth and S.-A. Berrani, "Unsupervised credit detection in tv broadcast streams," in *Multimedia (ISM), 2010 IEEE International Symposium on*, pp. 175–182, IEEE, 2010.

[10] A. E. Abduraman, S.-A. Berrani, and B. Merialdo, "An unsupervised approach for recurrent tv program structuring," in *Proceedings of the 9th European Conference on Interactive TV and Video*, pp. 123–126, ACM, 2011.

[11] L. Zheng, Y. Yang, and Q. Tian, "Sift meets cnn: A decade survey of instance retrieval," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 5, pp. 1224–1244, 2018.

[12] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 12, pp. 1349–1380, 2000.

[13] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *null*, p. 1470, IEEE, 2003.

[14] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[15] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, pp. 2161–2168, Ieee, 2006.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *2007 IEEE conference on computer vision and pattern recognition*, pp. 1–8, IEEE, 2007.

[17] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *European conference on computer vision*, pp. 304–317, Springer, 2008.

[18] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR 2010-23rd IEEE Conference on Computer Vision & Pattern Recognition*, pp. 3304–3311, IEEE Computer Society, 2010.

[19] H. Jegou, F. Perronnin, M. Douze, J. Sánchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 9, pp. 1704–1716, 2012.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[21] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *European conference on computer vision*, pp. 584–599, Springer, 2014.

[22] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 53–61, 2015.

[23] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of cnn activations," *arXiv preprint arXiv:1511.05879*, 2015.

[24] "Ffmpeg." `http://ffmpeg.org`.

[25] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *arXiv preprint arXiv:1702.08734*, 2017.

[26] "Faiss - facebook research." `https://github.com/facebookresearch/faiss`.