University of Zurich[UZH]

Interactive Advertising Analytics

Thesis October, 13, 2015

Sofia Orlova
of Moscow, Russia

Student-ID: 11-729-738
sofia.orlova@uzh.ch

Advisor: **Shen Gao**

Prof. Abraham Bernstein, PhD
Institut für Informatik
Universität Zürich
http://www.ifi.uzh.ch/ddis

Acknowledgements

I would like to express my gratitude to Shen Gao for guiding me through this thesis and for his input, time and support. Also I would like to say thank you to Prof. Abraham Bernstein, PhD for giving me the opportunity to write my bachelor thesis at the DDIS group. Also I am very grateful to my parents for the possibility to live and study in Switzerland. Without their support I wouldn't be where I am today. Last but not least, I am grateful to Basil Philipp and my friends and roommates for all the advice and encouragement that they gave me.

Zusammenfassung

Werbung ist überall. Wir nehmen es zwar nicht bewusst auf, jedoch sind wir täglich ca 2500 - 10000 exponiert. Von Google, Youtube und anderen Giganten sind wir es uns gewohnt personalisierte Werbung zu erhalten. Obschon diese nun zum Alltag gehören sind noch längt nicht alle Branchen soweit personalisierte Werbung zu gestalten. Eines der meistkonsummierten Kommunikationsmedien, nämlich das Fernsehen, hat diesen Schritt noch nicht gemacht. Die demographischen Information für die personalisierte Werbung waren schlicht und einfach nicht vorhanden. Online Fernsehen Portale verzeichnen mehr und mehr Benutzer, die sich mir den nötigen Informationen registrieren. Dies ermöglicht die Analyse des Benutzerverhaltens. Hinsichtlich dieses Fortschritts sind Mechanismen nötig, die Werbung in den live streams identifizieren, um diese denn mit demographischen Infomationen der Benutzer vergleichen zu können und personalisierte Werbung zu empfehlen.
Diese Arbeit beschäftig sich mit der Entwicklung eines solchen Systems, welches Werbung in den live streams anhand dessen Farbverläufen erkennt. Der Mechanismus ist so konzipiert, dass er für die Extrahierung der demographischen Benutzerdaten erweiterbar ist. Ebenfalls sind die Algorithmen um weitere Funktionen erweiterbar.

Abstract

Advertisement is everywhere. Whether we are aware of it or not, in average we are daily exposed to 2500 - 10000 ads. We are used to customized ads from google search, youtube and several more big players. But not every industry is that far yet. In fact one of the oldest communication media, namely the television doesn't show you customized ads yet. The demographic information needed for such a customization simply wasn't available. Online televion portals are gaining more and more users, which register themselves with the necessary data, which makes their habits traceable. Regarding this development tools to recognize the ads in live streams are required, in order to use the demographic information of the user and propose customized ads to him. This thesis describes how I build such a tool, that compares the video streams to ads based on their colour distributions. This mechanism can be used to expand its functions for extraction of the demographic data and combine the comparison mechanism by adding further recognition features.

Table of Contents

1

Introduction

It is common knowledge, that customized advertisement is a billion dollar industry and that this trend won't blow over soon. It surprises a lot of people, that online television didn't make the step to demographic classification earlier. Recommender systems, that will propose shows you might like and articles you might buy, based on your previous behaviour are running in a lot of mechanisms you interact with daily. The video processing industry on the other hand has been in the shadows for quite a while now. The need to process and analyse multimedia data didn't just emerge from the online television portals. Since the foundation of instagram, 5 billion pictures have been uploaded (status 2012)[KGC, 2013], facebook gets in average 300 million new pictures daily[KGC, 2013], while an average smart phone is equipped with a camera that has a resolution of eight mega pixels. The need to analyse this data is overwhelming and its potential huge.
The biggest challenge in multimedia information retrieval is the semantic gap. The semantic gap describes the meaning of the actual content and the overall meaning of the document. By looking at a balance sheet its quickly clear, what the overall meaning will be. By looking one frame in a video file the interpretation of the whole video becomes way harder and even if we process all the frames we still need a mechanism that can map a meaning to all this objects in all the frames. The human brain is incredible when it comes to processing and mapping this information. Just by watching five seconds of a scene where two people interact with each other the viewer can tell whether it was a sad or a happy scene, in which epoch the plot is playing, the approximate age of the actors, their gender are just the most obvious factors, there are a lot more.
Based on that the first step in my thesis was to determine on which factors my engine should recognize the ads in video streams. Object recognition is the first thing that comes to mind if we are trying to compare images and videos. This method determines the edges in the frames, but by doing that the next step would be putting it in a semantic schema to really make a conclusion, which is much harder.

The engine I build uses comparison of RGB colour values to determine whether this values are similar in two frames or not. Basically it calculates the percentage of a colour value in a frame, compares it to the values of ads stored in the database and decides whether it is a match or not. The big advantage is that we don't have to deal with moving objects or foreground-background segmentation. The overall colour distribution

Figure 1.1: Frame with colour histogram on it. Frame extracted from an Carlsberg ad

will stay the same. Each second of a video is composited of 25 images, that are called frames. By computing the histograms for color distributions for each frame we can compare them to the precomputed histograms for the ads in the database.
The contributions of this thesis are the following:

- An engine that recognizes matching frames based on their colour distribution
- An engine for uploading ads that distracts the colour values and saves them as histograms
- The whole project is extensible so that a lot of new features can be implemented.
- The algorithms make the decisions based on colour values. For further analysis, for example with audio detection the features can be implemented in the same project, using already embedded libraries, which contain this features.
- Limitations of the reliability of the algorithms are discussed.

2

Literature Review

Before dealing with possibilities of image processing, possible tools, libraries and features a basic understanding of an image was needed. Each image has a certain resolution, that means a quantity of pixels. For better understanding of its structure we did the course course Fundamentals of Digital Image and Video Processing, held by Aggelos K. Katsaggelos from the Northwestern University. The recommended background knowledge is this course is basic programming. The course contains a introduction on how the image is composed and stored followed by basic explanation of how image and video are processed and analysed and what a magnetic spectrum is. Later it dealt with 2D and 3D signals, convolution, several form fourier transformation, motion estimation, video enhancement (e.g., edge detection, noise filtering, histogram equalization, inpainting), recovery of the data, compression and segmentation and clustering. The course contains a lot of information which at first seemed a bit to detailed, but later when dealing with other papers the knowledge became very useful for beein able to quickly decide whether a paper contains the necessary information or not.
The next step was to determine which mechanism to use for add detection. The book Modern Information Retrieval[Baeza-Yates and Ribeiro-Neto, 2011] contains a chapter about multimedia retrieval. The sub chapter about color-based retrieval gave us the idea to build the mechanism based on colour distributions. The other sub chapters about texture, co-occurrence texture measures, salient points and sub chapters from the audio and music retrieval chapter were great guides to have ideas what could be done in the future and what the limitation for multimedia retrieval exist. This book was a great tool for starters and gave us very good ideas about what we wanted to do. For better understanding of the sub chapters we deepened the knowledge by going through several papers. To differentiate between the abilities of a computer and the human eye, the book Sensation and Perception[Goldstein, 2013] gives a very good introduction about how the human brain processes the environment around it. After reading this the problems that occur when dealing with semantic gaps become more comprehensible. The paper Another look at the problem of the semantic gap in image retrieval[Hare et al., 2006] explaines the various gaps and how they can be handled to a certain point. To discover further research on image processing and object detection we studied the paper Global Contrast based Salient Region Detection to deepen the knowledge in order to be able to build the project as extensible as possible.

Having read literature about several tools for image and video processing and analysis we decided to choose tools that are as easy as possible to extend, because there are so many tools that can used to improve the engine.

3

Content recognition mechanism

To build a mechanism that recognizes content test data is needed. The data we use comes from one Swiss OTV provider. To be able to recognize data from live video streams an algorithm is needed to compare content in general. The first step was to build an engine that can compare two videos to each other, based on their colour values. In the next step we rewrote the algorithm, so that it processes live video in a satisfying time and speed. Ultimately we build a graphical unit interface (gui) for better interacting and nicer visualisation. The interface is implemented as a responsive website, that adopts its structure to all possible formats of devices and demonstrates the algorithms behind it. Figure 2 shows the architecture of the project.

3.1 Choosing the right library

At first we needed a library that deals with image processing. After reasearching several libraries like JMagick, ImageJ and imgscalr we found one, namely openImaj, that provides the tools for comparison based on colour values. OpenImaj is a very large library[Hare et al., 2011], that provides functions from basic image manipulation to face and object detection to advanced data clustering and content analysis.

In average a second of video content consists of 24 frames. Each frame is basically a normal picture. The idea of the comparator mechanism is to compare each pictures with an other based on their colour distributions.

3.2 Ads

The ads, that have been uploaded to the database are iterated frame by frame and their colour values are stored in a list. The database class provides a method that returns a map with all the lists, which contain the histograms for the colour values. For each add there is a multidimensionalhistogram, a histogrammodel and a list. The multidimensionalhistogram creates a histogram with the given number of bins per dimension. The histogrammodel creates the multidimensional histogram calculated from image pixels then after passing the image to it, it calculates the model parameters. We save this

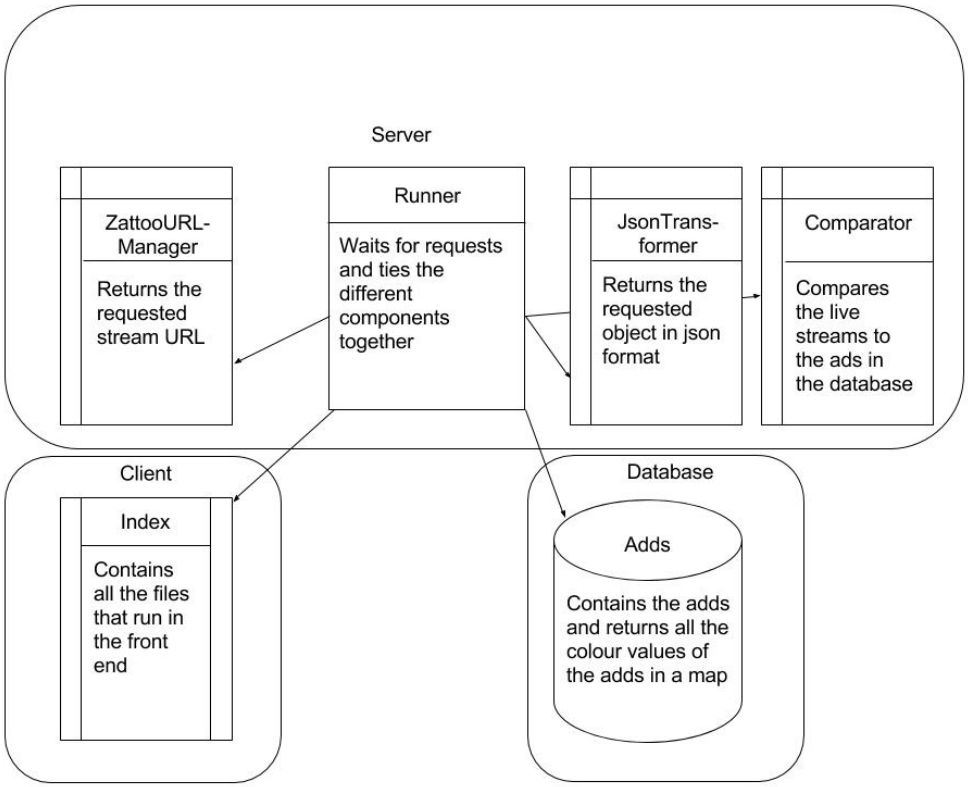

Figure 3.1: Client server architecture diagram

parameters as a histogram in the list. There are 24 histograms stored in the list for each second of the video content. After iterating through the video, the list and the name of the add are stored the map with adds.

3.3 Live stream content

The connection to the server of a popular Swiss OTV provider is established by making a post call to the API. This is done by creating a simple initial state object, adding the cookies from the persistent storage to it, getting the http client instance, adding several needed parameters and finally receiving the the list with the available channels. Each channel has a unique identifier and by adding this identifier as a parameter to the post call the stream url for the chosen channel can be accessed. The url manager class has a method that returns the stream url and the name of the channel in a map.

3.4 Graphical interface

For better demonstration of the comparator engine we decided to build a graphical interface. The visualisation of the project is based on a website. The website is based on HTML5 and CSS 3. For responsiveness we choose the Bootstrap css framework. To play the video live stream on the website we integrated the video.js plugin. Several other script are required to handle the m3u8 format of the stream.
The communication between the server and the client is established with asynchronous ajax calls. During the live stream the same live stream that runs in the front end is processed in the backand. The comparator engine takes a frame of the live stream, extracts the colour distributions from it, scales them so that they fit the colour histogram and compares to the frames of the ads from the database. As soon as a match is detected a local variable changes its status to add detected and saves the information in a map. The ajax call makes all two seconds a request to the server and gets the map with the stored values. Later we'll explain why we choose the time interval of two seconds. If the received request contains the confirmation of an add being detected, a script in the frontend changes the content of the website. We added the handlebars framework, a semantic web template system, to composite the html site. That way the handlebars template site is composed of standard html elements and elements from java files, that are rendered to html, so that together a html page is presented to the browsers. To run the website the java spark web application framework is implemented. It runs on a Jetty webserver. This framework requires the java version 8 or higher. All the settings of the other libraries and frameworks in the project have to be set to java 8 in order to work together. Even the newest version of the Maven plugin runs java 6 as default. To change that, the pom.xml file, namely the configuration file of the Maven project has to be manually set to java 8.
The axaj calls between the client and the server have to contain the information in JavaScript Object Notation format (JSON). Before making a call from the backend the required object has to be transformed in a JSON object by the JsonTransformer class. The benifit of it is that it is a language independent format. We choose to implement ajax calls for requesting whether an add is detected or not, because the asynchronous ajax call has the great benefit of only loading and then updating the requested part of the website. That way the user can keep watching the live video stream without the interruption of the whole page being reloaded all two seconds. In that case the user would need to press play every time the page was freshly loaded. In our case it wouldn't be fatal, since the graphical interface only supports the visualitaion of the mechanism we're actually presenting. But considering the future work on this project this implementation with ajax calls is of great importance.

3.5 Comparator engine

The comparator engine gets the stream url from the url manager. When the watch mode is on, which it always is when the engine is running, the algorithm creates a Xuggle

video object for the current frame of the live stream. The Xuggle video format allows java to process almost every video format and run operation on it. Then the current frame in the video format is extracted and stored as an image object. A multidimensionalhistogram amd histogrammodel are created to compute the colour histogram. The multidimensionalhistogram creates a histogram with the given number of bins per dimension. The histogrammodel creates the multidimensional histogram calculated from image pixels then after passing the image to it, it calculates the model parameters.
The engine has a map with all the adds from the adds database. After processing a frame of the live stream, the frame is compared to the first frames from the adds. If a match is detected this information is stored in a map with details about the add. There is a time thread of 42 millisecond before the next frame is processed. In one second there are 24 frames in agerage. 42 millisends is the average time scope of a frame being shown. A powerful cluster can do this computations for every frame and achieve a very accurate result.
We also tested the algorithm on a notebook with a four core 2.00GHz Intel i7 processor with an even worse graphics support of the Intel Inybridge Mobile, which counts as rather low and not suitable for image processing or gaming. This notebook manages to process a frame in two seconds. That basically means that it processes every fiftiest frame. It seems highly ineffective to even compare the fiftiest frame to the first frame in the add. Yet somehow the right add was detected.
If you go through several adds you might realise, that even if the scenary changes it won't do that until you've seen it for at least two to five seconds. In figure 3.1 and 3.2 you find extracted frames from an Wix ad. The frames are two seconds apart. On the frames we plotted the colour values of the colours red, green, blue, black, cyan, dark gray, light gray, gray, magenta, orange, pink, white and yellow. We compared the differences between each colors in procent. Then we did this for the other frame and compared the difference between each colour of the picture. The overall difference for the two frames is 0.07%. The resolution of the video doesn't matter since the colour distribution diagrams contain the information in procent.
Based on that information we set the threshold for matching frames by 2%. If the overall error is lower than 2% the algorithm sets the detected add variable to true.

We choose ten random adds from known brands and tested the overall error change for each frame. Table 3.1 shows the average amount of frame that have a colour distribution difference of less than 2% from one another. The detailed data and calculations can be found in the Appendix. As we expected the ads can be separated into several segments of very similar colour distribution. This segments usually have an overall error of less then 2% from one frame to an other. In average there are 117 frames that contain colour distributions that vary by less than 2%. If we take out the outlier ad Nike, we have an average of 48 frames. 48 frames are exactly two seconds of video. That makes it possible to run the algorithm on machines that are less powerful and need up to two seconds for computation.

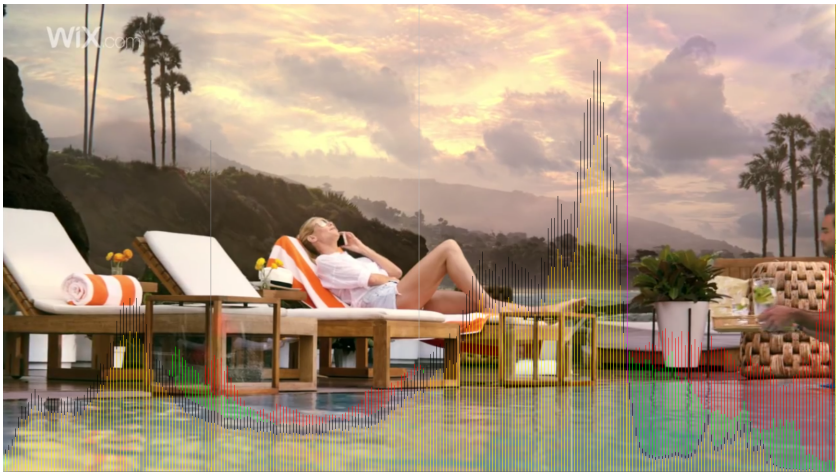

Figure 3.2: Extracted frame from the Wix ad, frame Nr. 554

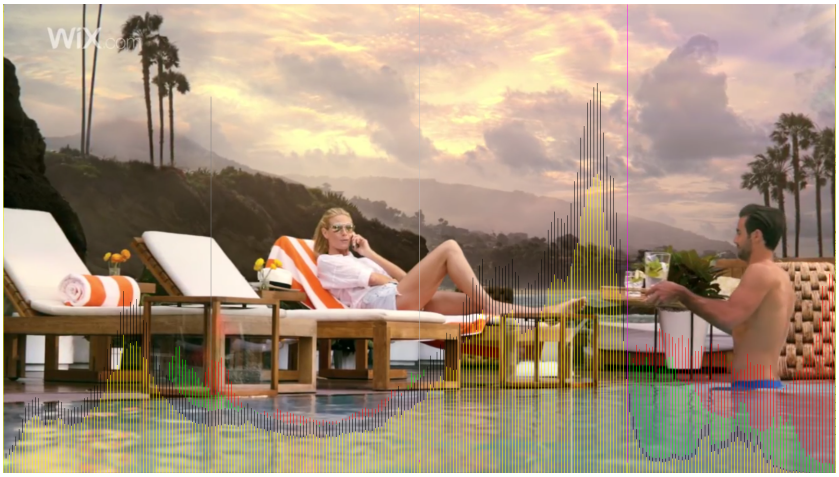

Figure 3.3: Extracted frame from the Wix ad, frame Nr. 587

Table 3.1: Average number of frames that have a less than 2% difference in colour distribution from one another

Ad	Rolex	Wix	Nike	ikea	Dove	Carlsberg	CarCrash	Netflix	Gucci	Burger
Avg. dif.	42	50	732	35	43	49	48	60	47	60

4

Limitations

Content detection based on colour distribution values has the benefit of setting a percentage threshold which regulates the accurateness. The problem remains that even though some pictures can have the exact same colour distributions their content can be completely different. More information can be gained from the colours by storing the distance of the colours into the histogram with the stored colours. For p_1 being the first pixel and p_2 the second we can compute the distance $d(p_1 - p_1)$ by predefining which colours to choose.
An even bigger challenge is the *colour constancy*.[Baeza-Yates and Ribeiro-Neto, 2011] The human brain is constantly repeating the perceptual process.[Goldstein, 2013] It starts with a motion from the environment and ends with the conscious experience of the perceived signal. If the midday sun is shining upon a banana palm the human eye sees a bundle of yellow bananas and perceives them as yellow. Same thing under conditions where the sun isn't shining directly upon the palm. Even in the twilight the human brain can process the image of the bent fruit and assign it to the category yellow banana. That is not possible with the approach of colour detection with colour distribution histograms. The computer clearly categorizes the pixels purely on their colour composition. And what happens if the image contains a banana that is older than a day. The fruit is more blackish-brown and the picture it will be on will correspondingly contain blackish-brown pixels.
Even though we found a way to run our algorithm on slower devices, because its able to detect a match set by a threshold, the light conditions of the frames still have to match more or less. To correctly detect the banana image it has to be in the same ripeness and be taken in similar lighting conditions.
One possible solution to improve the detection mechanism is to add object detection. There is model-based detection, that has predefined models to which certain objects should match, the image invariance methods, which basically do the same like model-based detection, just for several areas of the object and example-based learning algorithms that learn the models based on test sets.[Mohan et al., 2001]. But all these methods have the semantic gap in common.[Hare et al., 2006]. If you picture a fox and a squirrel the first thing that comes to mind are the bushy tails. If we show you two tails on images you can easily tell whether its a fox tail or a squirrel tail. To detect this features the object detection mechanism will fail without the whole image. There

are algorithmic approaches for detection with trained machine learning algorithms that might learn the S shape of a squirrels tail, but then again as soon as you zoom in and loose the shape the detection fails.[Mohan et al., 2001].
Our algorithm only focuses on colour values. That has certain limitations. Later we discuss the possibilities for future work to improve the accurateness.

5

Future Work

5.1 Future usage for the algorithm

As mentioned before in the limitations chapter there is plenty more work to do. Ad detection in live stream content is just the first step into the market of personalized advertisement. You can see in figure 5.1 how googles revenues from online advertisement grew yearly. Google alone generated 59.06 billion U.S dollars from personalized advertiment in 2014. Similar possibilities are thinkable for OTV providers. There are no statistical measures yet for determining the exact user behaviour. For using the online services users have to register themselfes with certain information.
The comparator engine can be extended by a mechanism that gets the user information for the current live stream, processes it statistically and proposes personalized ads to the user. At first we'd need to know who is watching the current stream and when this user turns it off. Based on that a machine learning algorithm could cluster the ads that were watched and rejected. If the cluster would fit to certain criteria, a recommender system may be used to propose ads with the same tags for the user. Amazon uses three features for their recommendations, namely traditional collaborative filtering, cluster models and search-based methods.[Linden et al., 2003]. Similar methods can be used for online television. Search-based methods can't be applied in the same way, but online television could apply own watch-based methods by not only processing the ads the user are watching, but also the other content. Someone who was watching Californication, 90210 and Melrose Place might want to visit the place this series were made and an ad about holidays in California might push the decision. This can be done with advanced data clustering, a feature that is provided by the openImaj library and can easily be integrated in the project. This is just one example of the possibilities there are for online television and personalized advertisement.

5.2 Extension of the detection mechanism with other features

Our detection mechanism works with colour distribution histograms. Its limits lie in the fact that two completely different pictures can have the same colour distributions.

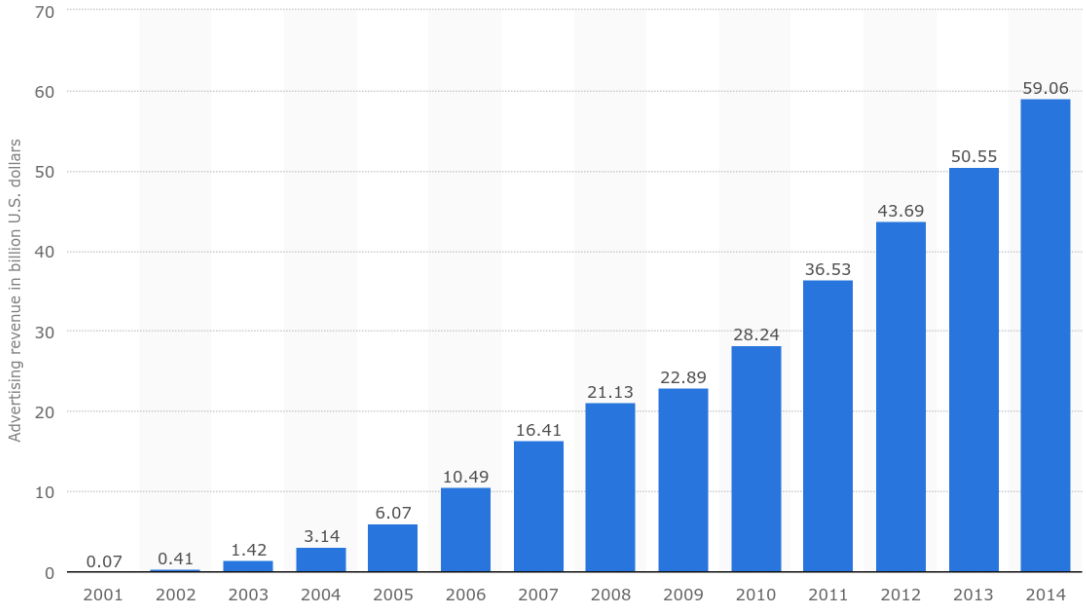

Figure 5.1: Advertising revenue of Google from 2001 to 2014 (in billion U.S. dollars)[Statista, 2015]

To make the algorithm more efficient several features can be added to the engine.The openImaj library provides several tools which can be added to receive a better result. One of the provided tools is salient region detection. This feature proposes a regional contrast based on a saliency extraction algorithm. As with the colours we could store this information for the adds and compare them to the live stream. Other approaches proposed by Ricardo Baeza-Yates and Berthier Ribeiro-Neto in Modern Information retrieval[Baeza-Yates and Ribeiro-Neto, 2011] are texture detection and Co-occurance texture measures. Texture detection works by describing the repeating patters of image intensity. This approach may be very interesting for object detection. To find a match while looking for advertisement it certainly may help, but its one of the more advanced methods.
Audio detection is something that should certainly be considered when looking for a match. A second of audio content can be broken into frames just like video content. The frames are called samples. In one second there are 44,100 samples also known as 44.1KHz. Based on the signal a waveform can be created and represented visually. This model can be used to compare the audio content. Again a threshold for similarity can be set. The openImaj library provides a lot of tools for audio processing. All this methods combined would certainly improve the accurateness of the engine.
We decided to work with the openImaj library, because its a great tool and can be used further to improve the mechanism. The engine is build to be expanded easily and build a lot of new features upon it.

6

Conclusions

The online televion market has a lot of potential for personalized ads. Now that the demographic information of the registered user is given, analysis tools can be build for that matter. We build an add detection mechanism by comparing the live stream content to ads in the database. The comparison is based on colour distributions of the images, that are computed and compared with the precomputed ones for the ads in the database. The algorithm performs well on powerful clusters and even manages to do its task on slower devices, since there are usually several seconds of similar content on the screen, what allows the colour distribution comparison between frames that aren't completely the same. The accuracy of the algorithm may be falsified by images with same colour distributions, but completely different content. To avoid this, several other features like object recognition and audio detection can be applied in future. The project is build so that it can be easily extended and build on.

References

[Baeza-Yates and Ribeiro-Neto, 2011] Baeza-Yates, R. and Ribeiro-Neto, B. (2011). *Modern Information Retrieval*. Addison-Wesley Publishing Company, USA, 2nd edition.

[Goldstein, 2013] Goldstein, E. (2013). *Sensation and perception*. Cengage Learning.

[Hare et al., 2006] Hare, J. S., Lewis, P. H., Enser, P. G., and Sandom, C. J. (2006). Mind the gap: Another look at the problem of the semantic gap in image retrieval. In *Electronic Imaging 2006*, pages 607309–607309. International Society for Optics and Photonics.

[Hare et al., 2011] Hare, J. S., Samangooei, S., and Dupplaw, D. P. (2011). Openimaj and imageterrier: Java libraries and tools for scalable multimedia analysis and indexing of images. In *Proceedings of the 19th ACM international conference on Multimedia*, MM '11, pages 691–694, New York, NY, USA. ACM.

[KGC, 2013] KGC, B. G. . C. (2013). Das internet in zahlen.

[Linden et al., 2003] Linden, G., Smith, B., and York, J. (2003). Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80.

[Mohan et al., 2001] Mohan, A., Papageorgiou, C., and Poggio, T. (2001). Example-based object detection in images by components. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 23(4):349–361.

[Statista, 2015] Statista, I. (2015). Advertising revenue of google from 2001 to 2014 (in billion u.s. dollars).

List of Figures

List of Tables

Appendix

1 Ad Statistics

Rolex Error	Frame Nr.	Difference
0.5760857994	457	432
0.4667979561	579	122
1.4380136257	604	25
0.2954066425	605	1
0.2491004897	606	1
0.26724505	607	1
0.639770066	619	12
0.3885831382	620	1
1.2349904194	634	14
1.4135724931	657	23
1.6823770492	713	56
1.5529753034	741	28
1.0703534171	822	81
0.8156536087	877	55
0.9295454545	937	60
0.2457685757	1006	69
0.7034649776	1048	42
1.089429423	1154	106
1.400234192	1233	79
1.2748296785	1351	118
1.3074249521	1526	175
0.5493772621	1614	88
0.814370875	1691	77
1.0360496061	1761	70
1.3559665744	1818	57
0.9917553758	1855	37
1.7501224186	1909	54
0.2211571216	1920	11
0.258223334	1929	9
0.2188737492	1930	1
0.3199169683	1934	4
0.2678677879	1935	1
0.2032680434	1936	1
0.2080796253	1940	4
1.4914679583	1941	1
0.2268362785	1944	3
0.2856610603	1949	5
0.4117521822	1950	1
0.3056844795	1956	6
0.245060677	1957	1
1.6057004471	1958	1
1.3046306153	2019	61
0.3265169257	2033	14
0.3111560571	2034	1
1.0681818182	2073	39
0.3439056845	2114	41
0.2252501597	2115	1
1.073334043	2139	24

0.3479561422	2156	17
0.2525601448	2189	33
0.3017883756	2191	2
0.2309027039	2206	15
0.8979561422	2209	3
0.5220140515	2276	67
1.6575686608	2286	10
0.2340057484	2321	35
0.2449595486	2322	1
0.2016446668	2323	1
1.863338301	2340	17
0.9008782201	2394	54
0.2009633809	2397	3
1.0131520119	2549	152
1.9107568661	2674	125

Average difference	42.047619048

Wix Error	Frame Nr.	Difference
0.4364800347	133	66
0.6905837674	197	64
0.6949544271	238	41
0.5751258681	242	4
0.577515191	244	2
0.5774153646	266	22
0.5764973958	268	2
0.5975716146	285	17
1.0083572049	311	26
1.0997699653	365	54
0.8921809896	379	14
0.8853125	381	2
0.5061328125	404	23
0.235766059	460	56
0.6809960938	553	93
0.2835264757	624	71
1.3846419271	766	142
0.5032335069	803	37
0.3668164063	905	102
0.6348546007	1008	103
0.8621310764	1068	60
0.5176063368	1096	28
0.7133181424	1160	64
0.3508637153	1197	37
1.5594552951	1284	87
1.7147309028	1378	94

Average difference	50.4230769231

Nike Error	Frame Nr.	Difference
1.3591319444	24	24
1.3538975694	1464	1440
Average difference		732

ikesStones Error	Frame Nr.	Difference
1.3488259549	18	1
0.3363715278	19	1
1.0300585938	21	2
0.4988085938	22	1
0.2596614583	23	1
0.2008376736	411	388
0.82	660	249
0.218999566	758	98
0.2042274306	790	32
0.2843402778	795	5
0.2810286458	797	2
0.238109809	822	25
0.2108463542	1165	343
0.203500434	1166	1
0.2029774306	1175	9
0.2341644965	1200	25
0.4089952257	1650	450
0.2704947917	1709	59
0.3261545139	1769	60
0.2751367188	1770	1
1.9217447917	1819	49
0.3460850694	1850	31
0.2242274306	1984	134
0.2356597222	2060	76
0.3674826389	2202	142
0.2019900174	2209	7
0.2047829861	2210	1
0.2409027778	2263	53
0.2723198785	2267	4
0.2531597222	2285	18
0.2068272569	2319	34
0.3139127604	2321	2
0.2090690104	2343	22
0.2273177083	2387	44
0.2934852431	2388	1
0.2474978299	2409	21
0.2942100694	2410	1
0.2073025174	2411	1
0.3125477431	2590	179
0.2235416667	2605	15
0.2475108507	2615	10
0.2432204861	2616	1
1.6788888889	2740	124
0.2171354167	2866	126
0.2108940972	2877	11
0.2702907986	2968	91
0.2112673611	2969	1
0.2136436632	2973	4

0.202734375	2981	8
0.3393055556	2987	6
0.2090407986	3008	21
0.2090299479	3016	8
0.2416341146	3033	17
0.5362434896	3035	2
0.2597677951	3050	15
0.4084244792	3051	1
0.4042643229	3052	1
0.265922309	3053	1
0.2259982639	3060	7
0.2195768229	3061	1
0.2095225694	3131	70
0.2796766493	3192	61
0.3038736979	3196	4
0.2617730035	3197	1
0.2254383681	3246	49
0.2205815972	3247	1
0.2030577257	3250	3
0.2143098958	3265	15
0.2882834201	3267	2
0.2329644097	3272	5
0.2839171007	3273	1
0.2605230035	3274	1
0.2184613715	3275	1
0.2012543403	3277	2
0.4253580729	3309	32
0.3420290799	3316	7
0.376187066	3348	32
0.7901236979	3349	1
0.2396788194	3618	269
0.2453059896	3624	6
0.2020616319	3648	24
1.7139518229	3717	69
0.37359375	3719	2
0.2892664931	3722	3
0.3570963542	3724	2
0.3343489583	3726	2
0.3247092014	3727	1
0.2489561632	3731	4
0.2510221354	3748	17
0.3945551215	3750	2
0.2081141493	3756	6
0.2241471354	3758	2
0.2208116319	3760	2
0.2111306424	3762	2
0.2599175347	3776	14
0.2927300347	3778	2
0.2642686632	3779	1

0.2794986979	3781	2
0.2345616319	3785	4
0.2831119792	3786	1
0.3574978299	3789	3
0.3935590278	3790	1
0.2718945313	3795	5
0.2884288194	3796	1
0.5449522569	4019	223
0.2142730035	4115	96
0.3447309028	4119	4
0.290687934	4126	7
0.3178342014	4129	3
0.2458355035	4135	6
0.3223220486	4136	1
0.2052777778	4142	6
0.2933702257	4146	4
0.2229144965	4192	46
0.2091276042	4198	6
0.2508224826	4211	13
0.2506749132	4216	5
0.2002690972	4218	2
0.2074717882	4224	6
0.2829253472	4225	1
0.2432834201	4226	1
0.3378602431	4227	1
0.2763519965	4252	25
1.7195138889	4321	69
0.9751302083	4389	68
0.2580338542	4392	3
0.2921831597	4472	80
0.280766059	4474	2
0.295375434	4482	8
0.2734939236	4490	8
0.2290885417	4498	8
0.2687174479	4499	1
0.2105859375	4505	6
0.2305013021	4507	2
0.2627907986	4509	2
0.2234244792	4520	11
1.0303103299	4687	167
1.8855729167	4915	228
1.9252083333	4919	4

Average difference	35.2661870504

Dove Error	Frame Nr.	Difference
0.2416015625	80	28
0.7581510417	120	40
0.2846223958	182	62
0.5848828125	222	40
0.8763151042	248	26
0.4301822917	305	57
0.7027864583	340	35
0.2270442708	369	29
0.6424479167	378	9
0.5016536458	450	72
0.291484375	533	83
0.5385416667	549	16
0.5783072917	594	45
0.2072786458	708	114
1.2876432292	743	35
0.7107161458	752	9
0.35359375	779	27
0.2408333333	829	50

Average difference	43.1666666667

Carlsberg Error	Frame Nr.	Difference
0.2208311632	56	31
0.278515625	86	30
0.4274913194	113	27
0.422280816	146	33
0.5195290799	172	26
0.2247287326	192	20
0.2131857639	193	1
0.2408875868	194	1
0.2007942708	225	31
0.6498763021	262	37
0.3238715278	296	34
0.3501128472	327	31
0.3032638889	457	130
0.2012977431	607	150
0.2552148438	731	124
0.3397222222	847	116
0.2030034722	952	105
0.3756293403	1130	178
0.5122417535	1192	62
0.2284657118	1218	26
0.2384592014	1247	29
0.328500434	1375	128
0.3296484375	1412	37
0.2990407986	1459	47
0.342578125	1517	58
0.4852625868	1554	37
0.4639908854	1574	20
0.2906488715	1756	182
0.2532139757	1856	100
0.2635850694	1890	34
0.3556054688	1931	41
0.3623806424	1953	22
0.5983919271	2023	70
0.6601085069	2338	315
0.2322960069	2370	32
0.2080772569	2382	12
0.3682226563	2384	2
0.2456901042	2385	1
0.5324262153	2387	2
0.2012304688	2388	1
0.2889453125	2406	18
0.6229752604	2408	2
0.5281966146	2427	19
0.2658875868	2439	12
0.5112152778	2445	6
0.2091471354	2453	8
0.2209027778	2455	2
0.6836284722	2469	14

0.2060763889	2492	23
0.2019270833	2499	7
0.6632052951	2501	2
0.3263736979	2530	29
0.2064887153	2539	9
0.3861697049	2557	18
0.8516927083	2582	25
0.427046441	2617	35
0.4009006076	2669	52
0.444233941	2689	20
0.2593315972	2713	24
0.2241080729	2762	49
0.4856705729	2818	56
0.4353884549	3049	231
0.3150065104	3128	79
0.2713476563	3296	168
0.2756467014	3353	57
0.2788237847	3383	30
0.231108941	3417	34
0.2103342014	3608	191
0.26734375	3680	72
0.2284418403	3714	34
0.6102235243	3756	42
0.9378038194	3787	31
0.4841059028	3816	29
0.2392599826	3950	134
0.2560807292	3973	23
0.2872829861	3993	20
1.0440082465	4014	21
0.7494661458	4033	19
0.6439930556	4057	24
0.4344292535	4088	31
0.3733723958	4115	27
0.4041124132	4133	18
0.2214735243	4161	28
0.5021506076	4181	20
0.5185503472	4199	18
0.5663563368	4236	37
0.3726584201	4333	97

Average difference	49.5172413793

CarCrash Error	Frame Nr.	Difference
0.8745225694	123	48
0.2299500868	152	29
0.4816276042	190	38
0.5817686632	217	27
0.4804644097	236	19
0.3728385417	258	22
0.2524153646	271	13
0.5397092014	320	49
1.4343012153	367	47
0.2159982639	434	67
0.5444466146	496	62
0.408969184	586	90
0.7108289931	672	86
0.6196115451	732	60
0.6317057292	854	122
0.6173914931	990	136
1.0732096354	1017	27
0.8098198785	1061	44
0.3691276042	1160	99
0.4762521701	1204	44
0.4586697049	1252	48
0.431703559	1318	66
0.5719032118	1361	43
0.4770269097	1442	81
0.4101085069	1502	60
0.2759027778	1544	42
0.7555902778	1592	48
0.8838476563	1633	41
0.5826649306	1672	39
0.2066883681	1709	37
0.4351497396	1729	20
0.2357703993	1732	3
0.2012000868	1743	11
0.3519270833	1744	1
0.2540321181	1746	2

Average difference	47.7428571429

Sheet1

Netflix Error	Frame Nr.	Difference
0.6838932292	186	74
0.9210590278	243	57
0.338843316	359	116
0.8812391493	463	104
1.2626671007	537	74
0.3487304688	594	57
0.3701801215	624	30
0.3821853299	644	20
0.7512413194	727	83
1.0937999132	758	31
0.6557595486	819	61
0.5100130208	843	24
0.4337521701	863	20
0.4056488715	912	49
0.245562066	1030	118
0.3961393229	1081	51
0.3325390625	1133	52
0.5399739583	1168	35
0.5408311632	1264	96
1.3588085938	1358	94
1.4039605035	1401	43
1.3027126736	1440	39
0.5793164063	1459	19
0.5887304688	1605	146
0.3675889757	1606	1

Average difference	59.76

Gucci Error	Frame Nr.	Difference
0.7682230679	493	147
1.2577332162	534	41
0.7071818891	585	51
0.8607289227	635	50
1.1276395394	686	51
0.9287226776	734	48
0.2149053474	831	97
0.2908665105	880	49
1.181376854	1071	191
1.1569525761	1130	59
1.1905737705	1203	73
1.1482142857	1246	43
1.1570843091	1295	49
0.9781664715	1331	36
1.0014734582	1380	49
1.1857045277	1427	47
1.5699599922	1475	48
1.3336407104	1514	39
1.3779713115	1558	44
0.8613680718	1602	44
0.3486436378	1642	40
0.612329235	1665	23
0.3532396565	1842	177
0.7457113583	1843	1
1.7914763856	1844	1
0.3175351288	1846	2
0.2094847775	1847	1
0.2070843091	1848	1
0.2567671741	1849	1
0.4215163934	1850	1
0.7111241218	1851	1
1.8430864559	1852	1

Average difference	47.0625

Sheet1

Burger Error	Frame Nr.	Difference
0.803921441	113	76
0.7309461806	143	30
0.8584548611	205	62
0.6103190104	238	33
0.4443077257	330	92
0.6256336806	360	30
0.649155816	399	39
0.3658962674	463	64
0.6040972222	494	31
0.4214344618	534	40
0.6758832465	636	102
0.4931857639	671	35
0.6311154514	743	72
0.3595269097	780	37
0.4426822917	809	29
0.9252929688	835	26
0.8318185764	858	23
1.880078125	1032	174
1.5471875	1037	5
1.6094509549	1237	200
Average difference		60

Overall Average of difference	Overall Averag
116.6986148	48.33179425

ge of difference without outliers