Deep Live Video Ad Placement on the 5G Edge

Mohammad Hosseini
University of Illinois at Urbana-Champaign (UIUC)
shossen2@illinois.edu

ABSTRACT

The video broadcasting industry has been growing significantly in the recent years, specially on delivering personalized contents to the end users. While video broadcasting has continued to grow beyond TV, video advertising has become a key marketing tool to deliver targeted messages directly to the audience. However, unfortunately for broadband TV, a key problem is that the TV commercials target the broad audience, therefore lacking user-specific and personalized ad contents.

In this paper, we propose a deep edge-cloud ad-placement system, and briefly describe our methodologies and the architecture of our designed ad placement system for delivering both the Video on Demand (VoD) and live broadcast TV contents over MMT streaming protocol. The aim of our paper is to showcase how to enable targeted, personalized, and user-specific advertising services deployed on the future 5G MEC platforms, which in turn can have high potentials to increase ad revenues for the mobile operator industry.

CCS CONCEPTS

• Networks → In-network processing;

KEYWORDS

Ad detection, deep learning, video streaming, MMT, 5G MEC

ACM Reference format:
Mohammad Hosseini. 2018. Deep Live Video Ad Placement on the 5G Edge. In Proceedings of ACM Multimedia Systems 2018, Amsterdam, Netherlands, June 2018 (MMSys ’18), 5 pages.
DOI: http://dx.doi.org/10.1145/3083187.xxxx

The video streaming industry has witnessed significant growth in the recent years, specially on delivering personalized contents to the end users. While videos have continued to grow beyond TV, video advertising have become a key business tool to deliver targeted messages directly to a wide audience. Statistics show that the US TV commercials produced 70.6 billion dollars in revenues in 2016 alone [10, 13, 14], and more specifically, 88% of short ads (of around 30 seconds), are watched on mobile devices [11]. Unfortunately, for broadband TV, a key problem is that the commercials target the broad audience and lack user-specific and personalized ad contents.

At the same time, use of AI-driven approaches on the cloud especially leveraging deep learning has become a de facto to provide computing-intensive processing services and automated solutions. With the rapid growth of video streaming and over-the-top (OTT) streaming market, leveraging AI can be a key marketing tool to personalize user experience and recommendations, for instance to deliver targeted video commercials.

In this work, we showcase a deep edge-cloud ad-placement prototype, and briefly discuss how we designed our system for both VoD services (offline mode) and live streaming from a broadcast TV. We employ the emerging MPEG Media Transport (MMT) protocol, and use that as our underlying streaming protocol. We implemented our ad detection system using both conventional cross-correlation based approaches as well as deep neural network methodology. The aim of our ad-placement prototype is to envision a targeted advertising service on the 5G cloud edge to replace general ad contents with more personalized and user-specific commercial contents. We implemented our system in response to the emergence of the 5G MEC and MPEG Network-based Media Processing (MPEG NBMP) initiatives, and showcase how it can be leveraged to provide personalized mobile services on the edge, potentially to increase advertising revenues for the mobile operator industry.

The paper is organized as follows. In Section 1, we briefly cover some background information and related work. In Section 2, we explain the design of our deep ad-placement prototype, and discuss the architecture and how we implemented the system components. We finally conclude the paper in Section 3.

1 BACKGROUND

1.1 MMT Protocol

Traditional one-way multimedia streaming services such as TV broadcasting using MPEG-TS or media delivery technologies have had the necessity to adapt with the trending MPEG multimedia technologies. The driving technologies for MMT emergence specifically includes IP convergence, dynamic and flexible multimedia consumption, as well as the Information-Centric Networks (ICN) [16].

MPEG Media Transport (MMT), as specified by ISO/IEC 23008-13 [12], is an emerging streaming protocol which provides a functionality for the delivery of coded media data for multimedia services
over concatenated heterogeneous all-IP networks. All-IP emergence has put a pressure on the TV broadcast services delivered over the air or via cable networks to adopt IP for the future broadcast TV networks. In addition, the increasing demand of more personalized services made end users more willing to consume media contents matching their preferences. Therefore, supporting dynamic and flexible multimedia contents transmission and consumption has become a key design element of MMT protocol. Due to MMT’s compliance with content personalization over an all-IP broadcast TV, we employ MMT as the underlying streaming protocol inside our system.

1.2 Cloud Media Processing and MPEG NBMP
Cloud computing aims to provide a distributed platform to perform various resource-intensive processing tasks in the network. Recently, many AI-driven cloud applications such as recommendation systems [17], autonomous cars [15], and live video analytics [18] have been studied on feasibility of deploying deep learning-based functionalities on the cloud edge for security, privacy, as well as latency improvements. There are also several industry practices such as [5], [7], [2], and [3] to name a few, to offer cloud media processing solutions, including live media encryption, adaptive streaming, or basic AI analytic such as object or scene change detection.

On the MPEG side, a new ad-hoc group has been initiated for Network-based Media Processing, or in short, MPEG NBMP [8]. Large-scale resource-intensive media processing tasks deemed to exhaust the client resources are offloaded to the cloud to benefit from the resource-extensive cloud platform provided by NBMP, also to respect the client’s limited resources. Our proof-of-concept demonstrated here is implemented in response to the MPEG NBMP functionality and requirements.

2 SYSTEM DESIGN AND STRUCTURE
The aim of our work is to showcase a system packaged as a container mounted on a edge-cloud server, as enabled by 5G MEC, aimed to benefit the operator’s services with lower service latency and enhanced user awareness. In our prototype, an automated ad-placement system is developed where ad video portions of a linear video stream are detected through AI engines, and are replaced with a secondary ad content supposedly a more personalized and user-specific ad. Figure 1 illustrates the concept. A video content, possibly containing temporal commercial ads, is given as an input to our cloud media processing platform. Our developed system can support VoD scenarios as well as live scenarios where broadcast TV streams are continuously ingested to our cloud system. AI engines residing on the cloud edge recognize the ad contents inside the input video contents. Upon processing, the detected ad contents are then replaced with secondary ad videos, based on the user’s location, view history, or other preferences. The output video streams are then aggregated together, and the new video is streamed to the client via MMT protocol.

Figure 2 illustrates an abstract architecture of our AI-driven cloud ad-replacement system, with the dashed area identifying components inside the virtualization core, packaged as a LXC container deployed on a hexa-core Intel Xeon Core i7-6900K running 64-bit Ubuntu 17.04 LTS. In the following, we enumerate the major components of our system.

2.1 Broadcast TV Receiver
To enable live media processing, we used August DTA300W, an ATSC digital TV antenna with 20dB gain amplifier integrated with our server machine that generates audio and video signals that are picked up from over-the-air broadcast TV. We used the open-source Tvheadend 4.2 TV streaming server [9], and configured the ATSC receiver to feed live media contents from a tuned TV channel, in our case, Laff[6] and Bounce[4] TV channels. We implemented an x264-based transcoder to ingest the MPEG-TS streams, temporally segments the live streams into videos of shorter duration while encoding and packaging, and records the audio and video files in a shared storage for processing.

Figure 3 demonstrates our server machine together with the ATSC digital TV receiver setup.

2.2 RESTful Server
Given the distributed nature of our prototype, a RESTful server residing on the edge cloud is designed to remotely communicate with the user, receive the client’s media processing requests, and initiate the specified media processing pipeline. A client initiates the ad-placement media processing service through communicating with the RESTful server and sending a request through the RESTful API that we have developed. We used Node.js 6.11 with Express 4.15.3 web application framework to develop our REST server and API.

For convenient deployment, we implemented some utility functionality on the server to notify the client with updates on the media processing tasks, such as the user’s geographical information, server’s specs, and the total processing time. The RESTful server then prepares the media processing pipeline, deploys the media processing container, and starts the ad detection AI engine. The source stream, either the source video uploaded by the client in the VoD mode or the broadcast TV streams in the live mode, are referenced through a unique URL. After that, the AI engine processes...
Figure 2: Abstract structure of our ad-replacement edge cloud system.

Figure 3: Our server and the ATSC digital TV receiver setup.

Figure 4: The workflow of our deep neural network AI engine.

Figure 5: Our cross-correlation AI engine as applied on an example frame from Laff TV.

2.3 AI Engine

The main purpose of the AI engine is ad detection and content categorization in a given video segment. To achieve that, we proposed a novel deep neural network that extracts certain features using the audio-visual information of a video content, and performs ad detection and content recognition simultaneously. We developed our deep neural network using TensorFlow 1.3 with Python 3.5 packaged in Anaconda 4.2.

Figure 4 briefly illustrates the deep-learning workflow in use by our AI engine. First, a shot detection and segmentation module splits a given video segment into multiple shots, each containing a consistent scene. The process is done through iteratively performing a frame difference analysis on consecutive frames to derive the probability distribution of a scene cut. Next, we develop a deep neural network to extract a shot’s audio-visual features, providing a basis to distinguish an ad content from a general non-ad video content. To achieve that, we perform a Mel-Frequency Cepstral Coefficients (MFCC) feature extraction method followed by dimension reduction and normalization. The list of features are then evaluated to determine the subset of features that must be used to classify the content. Once the optimum feature subset is selected, they are fused with post-processing to use for classification, and a target ad from a suitable category is identified for replacing the original ad.
content. We use 80% of a test dataset as training, and the rest for testing.

When extending our prototype to live contents, we witnessed that all the broadcast TV networks place the TV logo on the movie contents, while the logo is taken off when commercial contents are shown. To enable ad content detection based on that cue, we developed a simple AI module based on cross-correlation approach, which selectively analyzes any linear video streams, and finds a specific pattern in a long-duration signal, in this case, a single frame within a video segment. When a TV network logo is present in a video frame, our AI algorithm generates a vigorous cross-correlation peak at its approximate center compared to a case where no logo is present. We then apply a defined threshold to detect this peak, leading to detection of the logo presence and therefore, a correct classification. We used OpenCV 3.3 to implemented our logo detection AI engine.

Figure 5 shows the visual view of how our cross-correlation AI engine works on an example frame from LaF broadcast TV. Our cross-correlation algorithm is lightweight, high precision, doesn't require re-training, and runs significantly faster compared to our deep-learning-based approach on processing of live broadcast TV contents. Table 1 provides a quick comparison of the two AI engines on the broadcast content processing.

```
| Engine           | Execution Time (ms) | Accuracy |
|------------------|---------------------|----------|
| Deep-Learning    | 7477 ms             | 80.2     |
| Cross Correlation| 83 ms               | ≈ 100    |
```

2.4 Ad Meta-Data
The output of the AI engines is a text meta-data, which mainly includes time intervals of ad contents within the source video stream and a target ad content ID. In specific, the generated ad meta-data includes the following fields:
- ad content’s start timestamp
- ad content’s end timestamp
- ad content’s start frame number
- ad content’s end frame number
- is ad flag indicating if the content specified is ad (optional-always set to 1 for ads).
- target ad ID specifying the URI of the target ad that is going to replace the original ad content.

2.5 Ad Placer
After the ad meta-data is generated, it is then passed to an ad-placer module which substitutes the original ad with a new ad content given the target ad ID, and performs a concatenation on the whole sub-streams. We used Java 1.8 wrapped around FFMpeg 3.3.6 (Hilbert) as the core engine of our video splitting and stitching processes. The ad-placer module is interfaced with a shared storage to retrieve the source video segments, an ad repository for the target ads, as well as a shared storage to place the generated output videos. Upon completion of the ad-placement process, the generated video stream is then passed to the MMT streaming server, where the media contents are packetized and prepared for transmission. The generated MMT packets are then transferred to an MMT delivery server, which pushes the MMT packets to an MMT-compliant client for playout.

2.6 MMT Streaming Server
We implemented our streaming server on top of the MMT reference software package as a part of the MPEG-H standard in ISO/IEC 23008-4 [1]. The MMT streaming server consists of two components: The MMT packetizer, and MMT transmission server. The MMT packetizer takes a configuration file as an input with the location of input assets defined, reads fragmented x264 MP4-formatted audio and video files as its inputs, and encapsulates them into MMT Media Processing Units (MPU). The packetizer then generates the MMT flow as a multiplex of numbered packets, and stores them into a pre-configured shared location prepared for transmission. The role of MMT transmission server is then to take the MMT packets and transmit them to a registered MMT client using WebSocket protocol.

2.7 MMT Client
The MMT client receives an MMT flow, buffers them, de-multiplexes them, and then passes the packets to a reconstruction module to extract the payload and reconstruct the MPU. The MPUs are then sent to a decoding engine for playback. We implemented the MMT player using web-based technologies, making it suitable for a heterogeneous exposure via a JavaScript interface in HTML 5 compliant browsers. The interface provides a full-duplex communication channel, which also enables interactive functionality to initiate cloud media processing tasks and retrieve additional information such as the processing server’s specs, the total processing time, and other information. We also implemented a statistics module on the client that plots some of the players information, specifically the total playback time elapsed and the total time of buffered media.
Figure 6 illustrates an example playback session running on a Galaxy S8 smartphone, with the MMT-encapsulated media streamed over a 5G MEC testbed that we have developed. A short video of our prototype for VoD mode is also provided.

3 CONCLUSION

In this paper, we demonstrate and explain the design of an AI-driven ad-placement prototype placed on the edge cloud, and discuss how we designed our system for both VoD and live broadcast TV contents. We implemented our ad detection system using both cross-correlation as well as deep neural network-based approaches, and used the emerging MPEG Media Transport (MMT) as our streaming protocol. Our AI-based ad-placement prototype is aimed to help content providers and mobile operators to deliver targeted advertisement services with more personalized and user-specific ads on the cloud edge closer to the end users, possibly through the features enabled by upcoming 5G MEC and MPEG NBMP platforms.

4 ACKNOWLEDGMENTS

Research reported in this paper was conducted in collaboration with Samsung. We would also like to thank Prakash Kolan and Shervin Minaee for their assistance throughout different aspects of this study.

REFERENCES

[1] 2012. ISO/IEC CD 23008-1 MPEG Media Transport. https://mpeg.chiariglione.org/standards/mpeg-h/mpeg-media-transport. (2012).
[2] 2017. Alibaba Cloud Multimedia. https://alibabacloud.com/solutions/multimedia. (2017).
[3] 2017. Bitmovin Video Infrastructure. https://bitmovin.com/. (2017).
[4] 2017. Bounce TV network. http://www.bouncetv.com. (2017).
[5] 2017. Google Cloud Platform. https://cloud.google.com. (2017).
[6] 2017. Laff Media TV network. http://www.laff.com. (2017).
[7] 2017. Microsoft Azure. https://azure.microsoft.com/en-us/services/media-services. (2017).
[8] 2017. MPEG/NBMP. https://mpeg.chiariglione.org/standards/exploration/network-based-media-processing. (2017).
[9] 2017. Tvheadend TV streaming server for Linux. https://tvheadend.org. (2017).
[10] June 2017. TV advertising revenue in the US 2016-2021. https://www.statista.com/statistics/259974/tv-advertising-revenue-in-the-us. (June 2017).
[11] May 2017. Social Business: Why Do You Need to Use Social Media Videos? https://socialnewsdaily.com/66157/why-do-you-need-to-include-social-media-videos. (May 2017).
[12] October 2017. WD of ISO/IEC 23008-13:2017, Part 13: MPEG Media Transport Implementation Guidelines. https://www.iso.org/standard/70438.html. (October 2017).
[13] Mohammad Hosseini, Yu Jiang, Richard R. Berlin, Lui Sha, and Houbing Song. 2017. Toward Physiology-Aware DASH: Bandwidth-Compliant Prioritized Clinical Multimedia Communication in Ambulances. IEEE Transactions on Multimedia 19, 10 (2017), 2307–2321. DOI: http://dx.doi.org/10.1109/TMM.2017.2733298
[14] Mohammad Hosseini, Joseph Peters, and Shervin Shirjmohammadi. 2014. Energy-Efficient 3D Texture Streaming for Mobile Games. In Proceedings of Workshop on Mobile Video Delivery (MoViD’14). Association for Computing Machinery, New York, NY, USA, 1–6. DOI: http://dx.doi.org/10.1145/2579465.2579471
[15] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg. 2015. A Survey of Research on Cloud Robotics and Automation. IEEE Transactions on Automation Science and Engineering 12, 2 (April 2015), 398–409.
[16] Y. Lim, K. Park, J. Y. Lee, S. Aoki, and G. Fernando. 2013. MMT: An Emerging MPEG Standard for Multimedia Delivery over the Internet. IEEE MultiMedia 20, 1 (Jan 2013), 80–85.
[17] Brendan McManah, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agueira y Arcas. 2017. Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research), Aarti Singh and Jerry Zhu (Eds.), Vol. 54. PMLR, Fort Lauderdale, FL, USA, 1273–1282.
[18] Haoyu Zhang, Ganesh Anantharayanan, Peter Bodik, Matthias Philippose, Paramvir Bahl, and Michael J. Freedman. 2017. Live Video Analytics at Scale with Approximation and Delay-Tolerance. In 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17). USENIX Association, Boston, MA, 377–392.