INTERNET-DRAFT

Internet Engineering Task Force (IETF) Hong, Choong Seon
Category: Standards Track Kyung Hee University
Expires: October 10, 2020 Kyi Thar
Kyung Hee University
Ki Tae Kim
Kyung Hee University
Seok Won Kang
Kyung Hee University
October 2020

Edge AI assists Partial Content Caching with Smart Content Prefetching Scheme

draft-edge-ai-cache-00.txt

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on August 09, 2021.

Copyright Notice

Copyright (c) 2009 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents in effect on the date of publication of this document (http://trustee.ietf.org/license-info). Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

Abstract

Watching videos (Contents) from mobile devices has been causing most of the network traffic and is projected to remain to increase exponentially. Thus, numerous types of content and chunk based caching schemes have been proposed to handle the increasing traffic. Those caching schemes cache the whole videos at the edge nodes, but most of the users view only the beginning of the videos. Hence, caching the complete video on the edge node is an ineffective solution to reduce the network traffic as well as to improve the cache utilization. Thus, a chunk-level caching scheme to store popular videos partially and a smart prefetching scheme is needed to provide the missing chunks of the video.

This Internet-Draft will expire on August 09, 2021.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
   1.1. Terminology and Requirements Language 2
2. System Model 3
3. Process of Sending Learning Model to predict the popularity . . 4
4. Process of Content retrieving process from the Edge Node. . . . 5
5. Process of Content retrieving process from the Content Server via Edge Node. 6
6. Process of Content prefetching 7
4. IANA Considerations 8
5. Security Considerations 8
7. References 8
7.1. Normative References 8
7.2. Informative References 8
Authors' Addresses 9

1. Introduction

According to the CISCO, watching videos from mobile devices has been causing most of the network traffic and is projected to remain to increase exponentially [a]. Thus, many researchers are proposing numerous types of caching schemes based on reactive approaches and proactive approaches to handle the growing video traffic. In the reactive caching, the edge node decides to store videos when the requests or videos arrived [b]. In the proactive approach, popular videos are cached based on the prediction results before requested by any users [c][d].

The performance of the proactive approach is changing based on the efficiency of the prediction model. Currently, the deep learning models get huge attention to utilize in content's popularity prediction scheme because of the advances in big data and high computing power. The aforementioned caching schemes consider storing the complete popular videos at the edge nodes (i.e., Base station). The main issue is that most of the users view only the beginning of the videos because they stop watching videos when they do not like the beginning. Hence, caching the whole video is an ineffective solution to reduce network traffic as well as to improve the users' Quality of Experience (QoE).

Therefore, edge Artificial Intelligence (AI) assists partial video caching can be improved the cache performance. Additionally, edage AI based smart prefetching scheme can reduce the latency to access the missing chunks. The goal of this work is to minimize the latency to access the videos from the users' devices.

1.1. Terminology and Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. System Model

Figure.1 shows the overview system components needed to implement the proposed scheme. As shown in Figure.2 the cache storage space is divided into two partitions: i) Content Storage and ii) Prefetching Buffer. The Content Storage partition stores the partial popular videos and the prefetching buffer stores the current prefetching chunks of videos. The Popularity Prediction module predicts video popularity with the help of a deep learning model.The Cache Decision module decides to store the chunks of the video based on the popularity profile and historical data. The Prefetching Decision module performs the missing chunks retrieving process. Note that both Cache Decision and Prefetching modules utilize deep reinforcement learning.

```
   +-----------+             +------------+
   | Collected |<----------->| Popularity |
   |   Data    |<-         ->| Prediction |
   +-----------+  |       |  +------------+
                  |       |
                  -----   | +---------+
+-------------------+  |  ->|         |   +-------+
|      Cache        |  ---->|Cache    |<->|Content|
| +---------------+ |<----->|Decision | ->|Server |
| |   Content     | |       +---------+ | +-------+
| |   Storage     | |                    |    ^
| +---------------+ |      +------------+|    |
| +---------------+ |<---->|Prefetching |<-   |
| |  Prefetching  | |  --->| Decision   |     |
| |    Buffer     | | |    +------------+     |
| +---------------+ | |                       |
+-------------------+ |                       |
    ^                 |  +--------+           |
    |                -> |Request |<----------
    ----------------->   |Handler |
                         +--------+
```

Figure 1: System Model

2.   Process of Sending Learning Model to predict the popularity

Figure 1 shows that the process of sending the learning models from the cloud data center to the edge node, where the initial learning models are constructed at the cloud data center. Then, the edge node utilized the received learning models to predict the popularity of content and chunks.

```
  +----------+            +-------+
  |Cloud     |            |  Edge |
  |Datacenter|            |  Node |
  +----------+            +-------+
       |       Stage-1         |
       | -------------------->  |
       | +--------------------+ |
       | | Send Deeplearning  | |
       | |       Model        | |
       | +--------------------+ |
       |                        |
       |                        |
```

 Figure 2: Sending Learning model from Cloud Datacenter to Edge

3.   Process of Content retrieving process from the Edge Node

Figure 3 shows that the content retrieving process form the edge node with the case where the requested chunk of the content is located at the edge node. When retrieving contents from the user reach a certain chunk level threshold, the prefetching decision module pre-download the chunks before requested by users.

```
  +----------+                     +-------+                   +-------+
  |   User   |                     |  Edge |                   |Content|
  |          |                     |  Node |                   |Server |
  +----------+                     +-------+                   +-------+
       |       Stage-1                 |                           |
       | ------------------------->    |                           |
       | +--------------------+        |                           |
       | |  Request chunk 1 of|        |                           |
       | |   content A        |        |                           |
       | +--------------------+        |                           |
       |      Stage-2                  |                           |
       | <-------------------------    |                           |
       | +---------------------------+ |                           |
       | | Return Content            | |                           |
       | | If the requested chunk of | |                           |
       | | content A is in chace     | |                           |
       | +---------------------------+ |                           |
```

Figure 3: Content retrieving process from the Edge Node

4.Process of Content retrieving process from the Content Server via Edge Node

Figure 4 shows that the process of the content retrieving process from the Content server via edge node with the case where the edge node does not have the requested chunk of the content. The edge node makes a content popularity prediction based on the deep learning model and constructs the popularity profile of the videos. Then, the edge node makes a cache decision based on the collected videos accessed data and predicted popularity profile.

```
  +----------+                    +-------+                 +-------+
  |  User    |                    |  Edge |                 |Content|
  |          |                    |  Node |                 |Server |
  +----------+                    +-------+                 +-------+
       |       Stage-1                |           Stage-2           |
       | ------------------------>    | ------------------------>   |
       | +--------------------+       |+--------------------------+ |
       | |  Request chunk     |       || Forward the request      | |
       | |  of content A      |       || because the requested    | |
       | +--------------------+       || Content is not in cache. | |
       |      Stage-4                 |+--------------------------+ |
       | <------------------------    |         Stage-3             |
       | +--------------------------+ | <------------------------   |
       | | Cache (if popular) and   | | +------------------------+  |
       | | return requsted chunk of | | | Return requested chunk |  |
       | | content A                | | | of content A           |  |
       | +--------------------------+ | +------------------------+  |
```

Figure 4: Content retrieving process from the Content Server via
          Edge Node

5.   Process of Content prefetching

Figure 5 shows the process of content prefetching where the edge node autonomously retrieve the next chunks of the currently requested content chunk.

```
  +----------+                      +-------+                    +-------+
  |  User    |                      |  Edge |                    |Content|
  |          |                      |  Node |                    |Server |
  +----------+                      +-------+                    +-------+
       |       Stage-1                  |           Stage-2             |
       | -------------------------->    | -------------------------->   |
       | +--------------------+         |+-----------------------------+|
       | |  Request chunk 1   |         || Forward the request chunk 1 ||
       | |   of content B     |         || and chunk 1+n,cos requested ||
       | +--------------------+         || Content is not in cache.    ||
       |      Stage-4                   |+-----------------------------+|
       | <-------------------------     |      Stage-3                  |
       | +---------------------------+  | <-------------------------    |
       | | Cache and return          |  | +---------------------------+ |
       | | requsted chunk 1 and 1+n  |  | | Return requested chunk 1  | |
       | | of content B              |  | | and chunk 1+n of content B| |
       | +---------------------------+  | +---------------------------+ |
```

Figure 5: Content prefetching process

4. IANA Considerations

There are no IANA considerations related to this document.

5. Security Considerations

This note touches communication security as in M2M communications and COAP protocol.

6. References

6.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[a] CISCO VNI. Accessed: Feb. 7, 2019.

[b] Saeed Ullah, Kyi Thar and Choong Seon Hong, "Management of Scalable Video Streaming in Information Centric Networking," Multimedia Tools and Applications, Multimed Tools Appl (2016), 28pages, October 2016.

[c] Anselme Ndikumana and Choong Seon Hong, "Self-Driving Car Meets Multi-access Edge Computing for Deep Learning-Based Caching," The International Conference on Information Networking (ICOIN 2019), Jan. 9-11, 2019, Kuala Lumpur, Malaysia.

[d] K. Thar, T. Z. Oo, Y. K. Tun, D. H. Kim, K. T. Kim and C. S. Hong, "A Deep Learning Model Generation Framework for Virtualized Multi-AccessEdge Cache Management," in IEEE Access, vol. 7, pp. 62734-62749, 2019. doi: 10.1109/ACCESS.2019.2916080

6.2. Informative References

Authors' Addresses

Choong Seon Hong
Computer Science and Engineering Department, Kyung Hee University
Yongin, South Korea
Phone: +82 (0)31 201 2532
Email: cshong@khu.ac.kr

Kyi Thar
Computer Science and Engineering Department, Kyung Hee University
Yongin, South Korea
Phone: +82 (0)31 201 2987
Email: kyithar@khu.ac.kr

Ki Tae Kim
Computer Science and Engineering Department, Kyung Hee University
Yongin, South Korea
Phone: +82 (0)31 201 2987
Email: glideslope@khu.ac.kr

Seok Won Kang
Computer Science and Engineering Department, Kyung Hee University
Yongin, South Korea
Phone: +82 (0)31 201 2987
Email: dudtntdud@khu.ac.kr