International Journal of Asia Digital Art&Design

Received September 3rd, 2020; Accepted February 10th, 2021

Automatic Generation of News Contents from Blog Posts

Hayashi, Masaki
Uppsala University
hayasim77@gmail.com

Bachelder, Steven
Uppsala University
steven.bachelder@gmail.com

Tsuruta, Naoya
Toyo University
tsuruta@iniad.org

Teraoka, Takehiro
Takushoku University
tteraoka@cs.takushoku-u.ac.jp

Kanematsu, Yoshihisa
Tokyo University of Technology
kanematsuyh@stf.teu.ac.jp

Sasaki, Kazuo
Tokyo University of Technology
ksasaki@stf.teu.ac.jp

Kondo, Kunio
Tokyo University of Technology
kondo@stf.teu.ac.jp

Abstract
To promote UGC (User Generated Content) on the Internet, several techniques have been developed to allow users to create CG animations only by writing scripts quickly. TVML (TV program Making Language) is a technology capable of obtaining TV-program-like CG animation by writing text scripts. This paper aims to propose an application that automatically converts blog posts into CG animations with news show format with the aid of TVML. The process is: 1) to fetch HTML of the blog posts and perform web scraping and natural language processing to obtain summarized speech texts, 2) to automatically give a show format received from the analysis of professional TV program to get TVML script, 3) to apply the CG character and artworks, etc. that fit the blog content to obtain the final CG animation. In this paper, we describe the process and explain the application that we developed based on the method, and explain the evaluations of the outcome produced from the blog posts.

Keywords: Internet blog; News contents; Animation

1 Introduction

User-Generated Content (UGC), which is created and distributed by users, has been widely spread. In particular, the amount of UGC provided on the Internet, such as blogs, video sharing sites (YouTube, etc.) and SNS (Social Networking Service) are huge. It can be said that the current media would not continue to work without the UGC.

On the other hand, since UGC is the content produced by amateurs, its quality is often not as good as the one made by professionals. However, this situation varies depending on the type of content. When the content is, for example, a blog composed of texts and pictures, the difference between amateur and professional is becoming gradually small.

However, when it comes to the movie, the difference is still significant. The movie made by professionals such as, for instance, broadcasted television (TV) programs has much better quality compared to that of an amateur. We have to analyze a lot of special skills in movie production.

To enable CG animation creation even for amateur people, the technology called TVML (TV program Making Language) has been developed and used widely [1, 2]. Users can make a TV-program-like CG animation automatically generated by a computer by merely writing a text script. The show format of the TV program is given automatically. The point of TVML is to analyze techniques and know-how used in professional TV production, convert them into data and algorithms, and apply them to CG animation generation. The user can obtain CG animation like a TV program by the professional rendering format by merely describing the minimum "content" by a script.

In this paper, we propose a system that uses natural language processing to convert blog pages to TVML scripts to create CG animations using templates to mimic a real TV show, in this case, a TV news program.

UGC users can write blog entries, convert them to CG animation, and publish them via movie-sharing media. In our automation process, there are the following three essential points:

1) A method to obtain texts of the announcer's speech and superimposed texts used in a TV program from a blog post using natural language processing.

2) A method of automatically giving the effects (camera angle, artwork, video switching, etc.) to the texts obtained by the process 1) mentioned above to convert it into a TVML script. The effects are obtained by analyzing the professional TV programs.

3) Technology for setting up professional-quality CG characters, CG studio sets, the lighting used in CG animation made by TVML.

By combining these three aspects above, with the blog as a source, we are able to obtain CG animation output close to the professional quality. In this paper, we explain the details of the process and describe the developed application with the CG animation production experiment results from the actual targeted blog.

2 Related Work

2.1 Automatic Generation of Animation from Web Sites

Some attempts have been made to acquire text information such as websites, e-mails already existing in large quantities on the network, automatically impart a TV directing to it, and generate CG animation without manual work.

Nadamoto et al. [3] made a mechanism to define unique XML tags to the HTML of the Website and automatically convert the content of the site into a TV program produced by CG. Also, an attempt was made to specialize in news programs [4], by limiting the target news websites, and automatically convert them to news CG programs.

As a relatively successful example, an attempt is made to automatically generate a CG talk-show program from an Internet forum [5]. It fetches HTML of the Internet forum, extracts posts from it, makes six CG characters talk by synthesized voices, gives automatic camera switching, and ultimately enables users to "watch" the Internet forum by automatically created CG animation.

The above are the results of research on animation automation; however, even in professional broadcasting stations, there are many examples of content produced by using CG announcers. By observing the quality of the output CG animations from those several attempts mentioned above, we could say that it is important to give an appropriate "show format," which is fitted to the "content" very well. If the product is wrong or the product does not merely match the content, the output animation cannot convince viewers. This combination of the show-format and the content is the one that should be handled by professional people with much experience in the video production field. In other words, even if the system itself is automated and the Web page is easily converted into CG animation, however, professional skill in TV program making is required for the design of the automation system.

2.2 Mimicking Real TV Program by CG

In our previous study, we performed CG imitation of TV programs and an experiment for the evaluation in response to the problems mentioned in the previous section [6]. First, we analyzed in detail TV programs made by professionals and aired on TV, and we mimicked them to create CG animations using TVML. Then, the original TV program and the imitated CG program were presented to the subjects of the experiment using the survey method, and to evaluate the degree of completeness and satisfaction.

In the previous study, we conducted experiments on various program genres, but in this section, we introduce only the production of the news program used in this research. The target TV program is "NHK News 845" (NHK: Japan Broadcasting Corporation). Table 1 shows an excerpt of the analysis results. Based on the results of this analysis, we created a faithful imitation program using TVML. After that, 15 participants were shown the

Table 1 Analysis of TV program "NHK News 845" (excerpt).

Footage	Analysis
Opening	A short CG animation with the News logo and music (jingle). Duration time is 7 seconds.
Announcer and the studio	A male announcer wearing a suit in a studio set which is designed as bright and neutral type.
Camera angle	Fixed shot. Size of the head on the screen is approx. 43% of vertical height, the neck is at the middle line.
Superimposing name of the announcer	Bottom of the screen. Duration time = 5 seconds. Font size = 10% of vertical screen size.
Superimposing title of the article	Bottom of the screen, font size = 10% with some decorations, scrolling in from the left.
Speed of speech	Approx. 6.7 character/sec (average)
Interval pause between utterances	Approx. 2 seconds (fairly precise)

Figure 1 CG animation made with TVML by mimicking the actual TV news program.

real news program and the CG animation (see Figure 1) to evaluate. As a result, by faithfully imitating a program produced by a professional with TVML, it gained an average of 3.3 points on 5-level Likert scale ("1" is bad, "5" is good) in terms of completeness of the CG animation and the satisfaction level gained an average of 3.1 points. This result indicates that our methodology of using TVML animation by imitating a professional TV program has some amount of effectiveness. In this paper, the result of this previous study is used to guarantee the quality of automated CG animation creation. See literature for more detail [6, 7].

3 Proposed System

In this paper, we propose a system that automatically generates CG animation from Web content. The system first extracts necessary information from Web content using Web scraping and natural language processing. And our proposed method is able to convert it to a TVML script using a TVML template and apply appropriate CG models to generate CG animation as close to professional quality as possible. The process is described below.

3.1 Overall System Configuration

Figure 2 shows the system configuration. HTML text data fetched from the website is processed by Web scraping to extract necessary text data. The text data then is processed by natural language

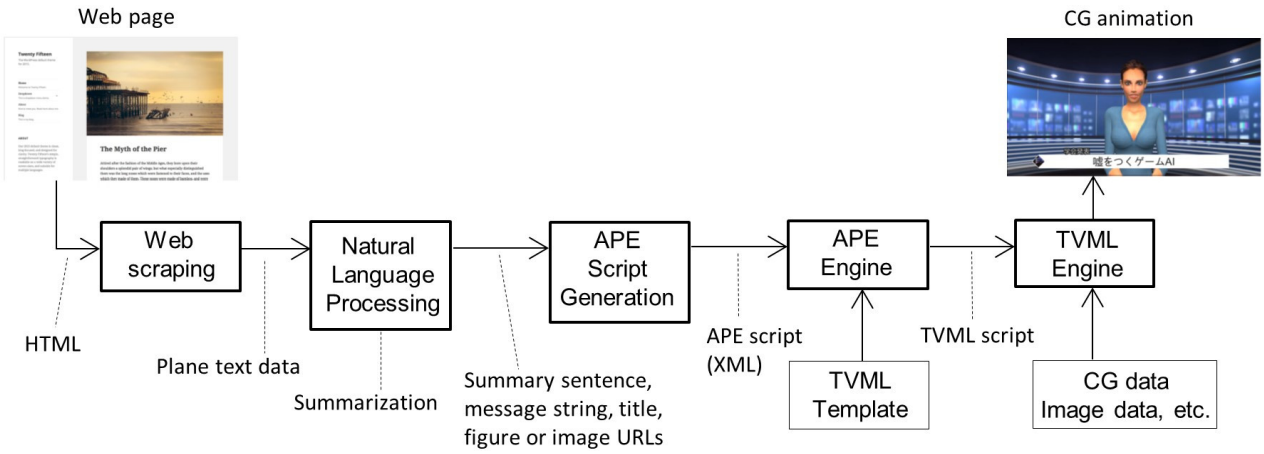

Figure 2 System configuration.

processing to produce a summarized text body of the article, one-line summary, a title, and image URLs used in the article. Those then are composed by the APE Script Generation module to get an intermediate file called "APE script" written in XML format (APE: Automatic Production Engine) [8].

Next, the TVML conversion module called "APE engine" converts the "APE script" into a "TVML script" by referring to the "TVML template". The "TVML template" is specifically made manually from a broadcasted TV program selected as a program suitable for the target Web content by analyzing it and imitating it. Finally, the system applies appropriate artwork to this and obtains the final CG animation [9].

In this study, we conducted an experiment using a Japanese blog of an organization for its publicity as a target web content. It is important first to select the target web content and design a process suitable for the content. We describe the details of the system construction process with the blog we chose from the next section.

3.2 Summarization of Text Data from the Web

The amount of text in a blog post will vary from post to post. In this paper, we provide the blog post in the TV news program format. In a news program, the amount of content of a segment for one topic usually nearly constant. Therefore, in our case, the text of the blog that is too long should be shortened.

For this purpose, we use a summarization algorithm in natural language processing to shorten the article content of the blog limited to 200 words. Also, in the news program, a single line of text is often superimposed on the screen, and the text is taken by summarizing the content being talked by an announcer. In order to obtain this superimposing text, it searches for the most important sentence in the blog article and outputs it. To obtain the texts from the web, we built a simple crawler that collected the blog entries and scraped their titles and texts. Then it was summarized automatically.

To summarize the text body, we employed the Basic Summarization Model [10]. As shown below, this is a word-based summarization model which calculates the optimum $score(s)$ in Eq. (1) for selecting salient sentences.

$$score(s) = \frac{1}{|S|}\sum \log freq(w_i) \tag{1}$$

Let $|S|$ denotes the total number of words in a sentence S and it is obtained by morphological analysis of S. Let $freq(w_i)$ denote the frequency of a word in a document where S and w_i are one of the sentences in the document and one of the words in S, respectively.

By using this model, our system calculates each score of the sentence in a document. After ranking the sentences by their scores, it extracts the sentences in descending order of their scores until a given length limit (e.g., 200 words) is reached. Finally, it connects the sentences in the written order and outputs them as a summary. At the same time, it also outputs the sentence with the highest score as the superimposed text. We also obtain the URL of an image used in a blog entry. When multiple images are used, only the first image is used in our research.

The information obtained above is converted into an APE script [8]. The APE script is an XML-based file with tags. A conversion example is shown in Figure 3. Table 2 shows excerpts of the tags and their functions. These tags are defined in a TVML template described in the next section, and the tags are defined differently depending on the program genre. However, there are some common tags such as <talk> of announcer's talking.

3.3 Making TVML Template

TVML template is a text file that the APE engine in Figure 2 refers to when it converts an APE script into a TVML script. The TVML template consists of a set of rules for replacing the XML tag of the APE script. Each tag in the APE script is replaced with a series of TVML commands written in the TVML template.

The TVML template is made manually. The process uses the result of our previous study described in section 2.2 and is done

as follows.

1) Select a TV program to imitate.

2) Imitate the TV program by writing a TVML script.

3) Edit the obtained TVML script to create a TVML template.

We chose a general TV news program because our target blog is for introducing general information. The TV program that we used is "NHK News 845" (as explained in section 2.2) which is an ordinary daily news program broadcasted at night. This program is done in a routine presenting several articles, starting with announcer pre-swing, followed by edited movies or static images with narration, and returning to announcer's wrap up.

As explained in section 2.2, mimicked CG animation with TVML is shown in Figure 1 and it was done based on the analysis result shown in Table 1. In this phase as shown in Figure 1, temporary CG character, CG studio set, music, sound effect, artwork, etc. which are similar to the TV programs are used. These items will be replaced with other artworks in a manner described in the next section.

Once the TVML script mimicking the original TV program has been made, a segment in the script will be replaced according to the contents that need to be identified, and then the segment is turned into a template. For example, let's take an example that you make a template from the script in Figure 4-(a). These three lines of TVML make gesture movement by motion capture data (character: mocap()), make a CG character speak with a synthesized voice with a subtitle (character: talk()), and delete the subtitle after the talk has finished (captionout()). The <talk> tag in the APE script is defined as <talk> in the TVML template (see Figure 4-(b)). The "$$" segment in the TVML template is replaced with the content of the <talk> tag in the APE script. Also, a value of the attribute "name" of the tag <talk> is put into the "$name$" segment. In addition to the above, some functions are prepared, such as initialization, ending process, setting default value when the element is omitted.

Through the above process, a TVML template corresponding to the chosen TV program is made. Once you create this template, CG animation can be mass-produced with this production pattern by giving a different APE script each time.

Table 2 Tag and its function of APE script (excerpt).

Tag	Function
<apescript>	Indicates that the document is an APE script. Specifies APE name.
<talk>	Makes a character talk
<bow>	Makes a character bow
<pause>	Makes a pause for specified seconds
<supername>	Superimposes a text
<superarticle>	Superimposes a text with effect
<superupperright>	Superimposes a text at upper right corner
<mainimage>	Displays an image file at full size

```
<apescript ape="Koukaton">
 <supername>こうかとん(Koukaton)</supername>
 <pause>1</pause>
 <bow></bow>
 <pause>2</pause>
 <talk>おはようございます、東京工科大学ニュースです。</talk>
 <supernameoff></supernameoff>
 <pause>1</pause>
 <superarticle subname="学会発表">嘘をつくゲームAI</superarticle>
 <talk>この「カタン」の面白さは、他プレイヤーとの「交渉」にあります</talk>
 <talk>交渉により、他プレイヤーと資源を交換することができるのですが、このときの駆け引きのうまさが勝敗に重要な要因となりますし、この「カタン」の醍醐味でもあります</talk>
 <superarticleoff></superarticleoff>
 <pause>1</pause>
 <mainimage>fig2.png</mainimage>
 <superupperright secondline="他プレイヤーとの「交渉」にあります">この「カタン」の面白さは、</superupperright>
 <pause>1</pause>
 <talk>中澤君は、コンピューターゲームでの「カタン」では、この交渉の部分が人間同士のプレイに比べてあまり面白くないと考えました</talk>
 <talk>その要因として、コンピューターとのコミュニケーションでは「嘘」が出てこないからではないかということに着目し、コンピューター側が嘘をついて交渉に臨むゲームAIを提案しています</talk>
 <talk>以下の図は、彼が作成したゲームAIシミュレーションの実行の様子です</talk>
 <pause>1</pause>
 <superupperrightoff></superupperrightoff>
 <pause>1</pause>
 <talk>おもしろメディアニュースをお送りしました</talk>
 <bow></bow>
 <pause>2</pause>
</apescript>
```

Figure 3 An example of APE script.

```
character: mocap(name=Bob, motion=talkgesture, wait=no)
character: talk(name=Bob, text="Hello guys, what's happening?")
captionout()
```

(a) Original TVML script

```
<talk>
character: mocap(name=$name$, motion=talkgesture, wait=no)
character: talk(name=$name$, text="$$")
captionout()
</talk>
```

(b) TVML template obtained from (a)

Figure 4 How to make TVML template from the original TVML script.

3.4 Generation of TVML Script with APE

Once the TVML template described in the previous section has been made, the template and other necessary data files such as CG character modeling data, CG studio set modeling data, image data, sound data, etc. are packaged to create a single APE [8]. The APE functions as a converter to produce a TVML script from a given APE script.

APE has its name and can be used by designating the name. In terms of the data structure on any storage device, we have a folder (APE folder) with the same name as the name of APE containing a TVML template and various necessary data. The APE engine reads the APE name written in the attribute of <apescript> tag in the APE script and converts the APE script to a TVML script using the APE of that name. This APE folder is portable. It can be placed on the local hard drive of the PC or the server. Once you create an APE folder, you can bring it anywhere to use it.

3.5 Building the Application

We implemented all the functions described above to make an application. Currently, it connects the following two processes manually: (1) to crawl HTML data from the blog, perform web scraping and natural language processing, extract summarize sentence and a message string for superimposing and an image URL, and (2) to generate a TVML script from an APE script from the results of (1) and play it with a TVML engine. The reason why we separated the two processes is just an implementation matter. We manually copy and paste the output from the process-(1) to the input of the process-(2).

For process-(2), we used Windows10 PC as a platform and produced the application using the TVML SDK [11, 12] implemented on the Unity Game Engine. On the other hand, the process-(1) was done in another computer. All the data elements extracted and processed from the blog post, are made automatically. And the TVML template, CG data, sound data, etc. that apply to the data elements need to be prepared in advance as described in sections 3.3 and 3.4.

Figure 5 shows the target blog and Figure 6 shows the application screen playing back the blog post. The player application has a user interface that a user can select pre-stocked data on the application by clicking a button at the upper left to see it as CG animation. Figure 7 shows some screenshots of the generated animation.

Figure 5 Target blog for the experiment.

Figure 6 User interface and playback screen of the application.

4 Experiments and Discussion

In this chapter, the experiments conducted in two parts are described. The first is an experiment to extract a summary, a message string, a title, and an image by web scraping and natural language processing to the HTML of a blog article, and its evaluation. The second is an experiment to produce CG animations visualized by giving the information obtained in the first experiment to the developed application.

4.1 Summary and Message String

The purpose of this experiment is to extract a title, a summary, a message string, and an image from a Japanese blog article. The message string is used for superimposing the string on the screen as often used in a real TV program.

First, 53 articles are chosen from the data of the blog site from which the titles and the texts and the image URLs of the articles are extracted by Web scraping. Then a text file containing the title, the text, and the image URL is created for each blog post. Ultimately 53 text files were created to be used as input for natural language processing software. The natural language processing module processes each text file by the method described in section 3.2 and outputs a “summary” and a “message string”. The title and the image URL (only an image that appeared first is used) also go through this module but without being processed.

The experiment and evaluation of extracting the summary and the message string are described below.

4.1.1 Summary Extraction Experiment

As a result of extracting the summaries of 53 articles, we successfully obtained the summary text of fewer than 200 characters for all articles. By reading the output summaries by us, it was confirmed that a certain degree of contextual consistency was ensured. However, although the sentences are connected in the original stroke order, the adjacent sentences were not always extracted as the most important ones.

4.1.2 Extraction of Message String and Its Evaluation

Using the summary text obtained in section 4.1.1, the most important sentence is extracted in the form of “section + predicate”, which becomes a message string. With this method, the message string for each of the 53 articles was automatically obtained.

Next, an experiment was conducted to evaluate whether the obtained message string was valid or not. This was done by comparing a human-created string with an automatically obtained string, and the title of the original blog post. This time, we used 15 examinees, given the 53 summaries, and asked them to randomly

Table 3 Results of comparison of 22 message strings.

	Both match	At least one matches
Comparison of manually created message string and blog title	14 (64%)	20 (90%)
Comparison of automatically and manually created message strings	8 (36%)	12 (55%)

create a message string. As a result, 22 message strings created by the two examinees were finally selected as the target strings (two different message strings were created for one article). Table 3 shows the result of comparing this target character string with the automatically generated message string, and the title of the blog.

Looking at the numbers for "at least one matches" in the table, the message strings manually extracted by the examinees from the summaries match with the title of the blog article at a high rate of 90%. Therefore, we ensure the manual summarization was well done and mostly valid. However, the matching of the automatically obtained message strings and the manually made message strings remains at 55%. This rate is not very high, so it suggests in the future, we should improve the automatic extraction method to increase the matching rate.

4.2 Experiments for Making News Contents

In this experiment, we gave the TVML script, which imitated the TV program and the target blog to examinees that make artworks suitable for the feeling. The news contents created by them are an opening title picture, a jingle (short music in the opening), CG studio set, decorative artwork used for superimposing, sound effects, ending titles, etc.

As mentioned earlier, the target blog is Japanese, and the contents are about media science, computer graphics, conference reports, etc. About 1,500 articles are collected from the blog archive, and natural language processing is carried out for the chosen 53 articles to extract 200 words of speech text, a superimposing message string, and one image URL. Those results are manually saved in a simple text file. The application reads this text file, converts it into an APE script just by simple replacement, and obtains the final CG animation. The results of the news content production are shown in Figure 7. The automatically summarized sentences and a message string in news content were mostly acceptable.

The following two problems were seen in the sentence of superimposing.

1) The superimposed text is obtained by choosing a sentence that appears in the blog entry with the highest score. It was observed that some superimposed texts do not appear to be appropriate. It suggests that it is still necessary to process the sentence further to use it as a superimposed text.

2) Some of the sentences are too long to fit into the superimposing. Also, regarding the image file, there was a case where it did not match the content of the narration, because the system displays the first image that appeared in the blog post.

Figure 7 Screenshots of generated CG animation.

5 Conclusion

In this paper, we have described a system that automatically converts blog entries into CG animations. The main points of our proposed system are the following.

1) Natural language processing of blog content, summarizing it to a length suitable for CG animation, extracting a superimposed message string, and an image URL.

2) A mechanism to imitate a TV program as a reference, converting it into TVML to produce CG animation by using a template made in regards to the sense of the original blog as a source.

There are two main features in our result.

1) The system has great potential for producing new kinds of content, mainly if it is consumed on mobile devices by young generations that do not have the patience to read blog entries and prefer to watch short videos.

2) Mobile device owners can choose to read a blog entry or watch it. They can watch entire news content which increases the number of news consumed.

Our future work is the following.

1) The application does not integrate all the elements of the entire process. We plan to integrate them in the near future.

2) The association between images and text is necessary when it comes to displaying the image at an appropriate position in the story and making an appropriate sentence for superimposing.

3) And it is also important to obtain more flexible and natural CG animation. In particular, there are issues such as designing CG characters that move naturally and humanly and having fluctuations every time so that viewers do not get bored.

References

[1] Hayashi, M., *TVML (TV program Making Language) - Automatic TV Program Generation from Text-based Script* -, In Proceedings of Imagina'99, pp. 119-133, 1999.
[2] Hayashi, M., Inoue, S., Douke, M., Hamaguchi, N., Kaneko, H., Bachelder, S, and Nakajima, M., *T2V: New Technology of Converting Text to CG Animation*, ITE Transactions on Media Technology and Applications, Vol. 2, No. 1, pp. 74-82, 2014.
[3] Nadamoto, A. and Tanaka, K., *Complementing Your TV-Viewing by Web Content Automatically Transformed into TV-Program-Type Content*, In Proceedings of the 13th Annual ACM International Conference on Multimedia (ACM Multimedia2005), pp. 41-50, 2005.
[4] Douke, M., Hayashi, M. and Makino, E., *Automatic Generation of Television News Shows from Given Program Information Using TVML*, The Journal of the Institute of Television Engineers of Japan, No. 7, pp. 1097-1103, 2000.
[5] Hayashi, M., Bachelder, S., and Nakajima, M., *Automatic CG Talk Show Generation from the Internet Forum*, In Proceedings of SIGRAD2016, pp. 43-45, 2016.
[6] Hayashi, M., Shishikui, Y., Bachelder, S., and Nakajima M., *An Attempt of Mimicking TV News Program with Full 3DCG - Aiming at the Text-Generated-TV System* -, 2016 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), pp. 1-5, 2016.
[7] Shishikui, Y., Nakamura, G., and Hayashi, M., *Assessing viewer satisfaction of CG programs as a substitute for real TV programs*, International Workshop on Advanced Imaging Technology (IWAIT) 2020, Vol. 11515, pp. 385-388, International Society for Optics and Photonics, 2020.
[8] Hayashi, M., Douke, M., and Hamaguchi, N., *Automatic TV Program Production with APEs*, In Proceedings of the Second International Conference on Creating, Connecting and Collaborating through Computing, pp. 20-25, 2004.
[9] Tsuruta, N., Teraoka, T., Kondo, K., and Hayashi, M., *TV Show Template for Text Generated TV*, 2018 International Workshop on Advanced Image Technology (IWAIT), pp. 1-3, 2018.
[10] Ouyang, Y., Li, W., Lu, Q., and Zhang, R., *A Study on Position Information in Document Summarization*, In Proceedings of the 23rd International Conference on Computational Linguistics: Posters (COLING 2010), pp. 919-927, 2010.
[11] Hayashi, M., Bachelder, S., Grippon, M., and Nakajima, M., *Interactive TV by Text-To-Vision - Application Using TVML SDK on UNITY* -, In Proceedings of 2013 International Conference on Cyberworlds, pp. 373-373, 2013.
[12] *TVML Homepage*, accessed Jan. 12, 2021, http://tvmllab.com/