International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Index Copernicus Value (2016): 79.57 | Impact Factor (2017): 7.296

Chatbot for Generation of Subtitles in Regional Languages

Shreyas B, Aravinth P, Kripakaran P

[1, 2, 3]College of Engineering Guindy, 12, Sardar Patel Road, Guindy, Chennai 600025, India

Abstract: ***Videos are being watched by all people regardless of their native language. Most of the video content and lectures are in English. This makes it difficult for some people to access the content due to the difference in language. Currently, there are systems that provide subtitles for video content in the same language. We have come up with an innovative solution to this problem. The objective of this project is to generate subtitles in regional languages, such as Tamil, where the user gives the video as an input through a very simple user interface such as a chatbot.***

Keywords: chatbot, speech recognition, machine learning, natural language processing

1. Introduction

With the wide spread popularization of 4G services in India, the number of internet users have spiked to 500 million in 2018 from 240 million in 2014. There are 220,000,000 smartphone users and 90% of them have data connections and a total of 138 Petabytes per month. According to a study by Google 45% of the data consumed is for videos which is a massive amount. There is a huge scope for growth in the number of smartphone users thereby making it one of the most important aspect of the Indian internet scene. Around 200 million of these users are from rural India who are non-English speakers. Video is an audio visual medium through which people understand and comprehend what is being delivered to them in an easier way and holding the viewer's attention. With the popularization of major video-sharing websites like YouTube and Dailymotion consumers have an ocean of content to view. The genres vary from entertainment to education. One major obstacle is that majority of the content is in foreign language mainly English and the upcoming users in India have a language barrier.

2. Literature Survey

The automatic subtitle generation solution by Xiaoyin Che, Sheng Luo, Haojin Yang and Christoph Meinel [1] uses the IBM Watson Speech-to-text API as their ASR tool. The transcript file is received in almost twice the duration of the audio file. Then they have implemented sentence boundary detection using a pre-trained word vector lexical model to make sure the synchronization of the subtitles is done right.
Another solution for subtitle generation by Rajeswari Sridhar, Aravind S, Hamid Muneerulhudhakalvathi and Sibi Senthur M [2] suggest that after ASR by the CMU Sphinx API they train and use two models namely the acoustic model and the linguistic model. The linguistic model is trained to capture the properties of a language and it predicts the next word in the sentence whereas the acoustic model is used to predict the pauses in human speech and the end of line in subtitles. They use a hybrid model of the combination of the two models explained above.

Stelios Piperidis, Iason Demiros and Prokopis Prokopidis [3] have designed an infrastructure for a multilingual subtitle system. They use a English ASR module to transcript the audio streams and the system adapts to the speaker`s style. This enhances the overall accuracy of the ASR system. In addition to this there is a multilingual translation subsystem that comprises of machine translation, translation memories and terminological banks. They have trained a model using audiovisual data from BBC TV programmes and transforming them into multifaceted parallel data that consists of the actual video, the transcript and English, French and Greek subtitles and relevant web extracts on the topic.

Apart from the core functionality this solution utilizes a chatbot for a convenient user interface. Godson Michael D'silva, Sanket Thakare, Sharddha More, and Jeril Kuriakose [4] propose a Real World Smart Chatbot for Customer Care using a Software as a Service (SaaS) Architecture. The proposed architecture uses ejabberd server an instant messaging designed in erlang, AWS Lambda a computing platform followed by an Amazon API Gateway to create, publish and monitor APIs and then a chatbot module. The chatbot module is a chatterbot, a conversational computer program that allows the user to interact with the machine through conversational dialogues.
A technical survey by Lorenz Cuno Klopfenstein, Saverio Delpriori, Silvia Malatini [5] examines a few advantages of using chatbot as a conversational interface as in the case of our solution. The advantages are that the bot need not be downloaded and installed it is instantly available without any extra requirements, limited data requirements and less heavy on the processing machine, and also a faster iteration in development as code is implemented on the server side completely. In the conclusion they have stated that although bots may not take over traditional applications completely, it will definitely be widely used as a software platform to deliver services to users.

Although there are several solutions to the subtitle generation problem there has seldom been solutions that have helped in generating subtitles in other languages. The solutions that we have discussed also are not available as a complete product to an user but just as a back-end module that has to implemented as a part of a larger solution.

3. Methodology

3.1 Chatbot

The chatbot works based on Natural Language Understanding. Our system uses reinforcement learning along with NLU to generate a model based on the initial training data. Each user query is processed by NLU component initially. Intent Classification and Entity Extraction is done, and a model is generated. The NLU component performs parsing on the user text and gets a meaning out of the text by extracting intent from the text and the information associated with that intent. Tokenization is performed initially, which is the process of dividing the text into tokens. For example, let "Please translate the video" be the user query. This text is broken down into tokens as: 'Please' 'translate' 'the' 'video'. The intent of the text is found, and an appropriate response is generated from the model. Intents represent a mapping between user's intent and the action to be performed by the chatbot. Entity Extraction is identifying and extracting entities from the user query. It is evident from the results that our system performs better than the rule-based pattern matching techniques.

Dialogue management is seen as a classification problem. We use Hybrid Code Networks [6] to resolve this problem. Hybrid Code Networks combine Recurrent Neural Networks and domain specific knowledge as software to generate responses. After tokenization, a bag of word vectors is formed. Entity extraction module identifies entities from these word vectors. The text and entities are then passed to the entity tracking code, provided by the domain-specific developer, which maintains entities. This code returns an 'action mask', which indicates actions that can be performed at current timestamp. These feature components are combined to form a feature vector. This feature vector is passed to RNN which computes a hidden state vector. This hidden state vector is then passed on to a dense layer with softmax activation function. The output dimension is equal to the number of actions possible. The result is normalized to a probability distribution from which an action is selected.

3.2 Transcript Generation

The generation of subtitles for videos encapsulates three steps. The first phase is extraction of audio from the .mp4 input video file. For speech extraction from video, we use the FFMPEG command line tool. FFMPEG stands for Fast Forward Moving Pictures Expert Group and it is one of the most popular frameworks used for audio and video manipulations. We use a python interface to run commands in the terminal – which is achievable thanks to the *subprocess* module of python. The subprocess module allows us to create and start new applications from our python program. We use the subprocess.call statement to handle the audio extraction by specifying the input file, the codec (encoding and decoding) type which is audio in our case, the number of audio channels and the compression level required as part of the ffmpeg command. The output obtained is a .wav audio file that represents an accurate extraction of audio part from the input .mp4 video file. The next step is speech recognition using the Google Speech to Text API. But it would not be possible to process the entire audio file at a stretch, as the API cannot recognize large files at one instance. The audio is therefore, split up into a number of segments and fed into the Google API. The conversion of the audio file into a number of small-sized blocks is done with the help of AudioSegment command of Python, which is a simple and high level interface for audio manipulations.

Also, the AudioSegment command only accepts .wav files as input, which conveniently, is the output from the first part where we extract the audio. The final step would be the generation of subtitles as .srt file. The list of words along with the timestamps in which they occur from the previous module is fed to the third module to eventually obtain a subtitle file (.srt). Proper synchronization is a must and to realize this, we use the synchronous speech recognition technique provided by Google Speech to Text. Note that this method can only be used for short length audios, another reason why we split the original audio into a number of smaller portions. For consecutive sentences, Silence utterances can be used as delimiters. We use VLC media player, because this supports automatic subtitle file generation without any manual involvement.

3.3 Transcript Translation

The control now transfers to the culminating module where we translate the transcript obtained from English to Tamil or any other regional language. To preserve the subtitle format with the timing constraints we have to carefully parse through the complete srt file and translate the transcript to a regional language that does not the affect the timing synchronization. We use Yandex [7] API an open source API to translate the transcript from English to Tamil. Since the subtitle cannot be directly translated we have to write an algorithm which maintains the time synchronization and at the same time the time frame has to also be maintained.

4. Results & Discussion

4.1 Chatbot Results

The chatbot was also evaluated through random conversations and topics to test if the chatbot`s NLU recognizes the intent. We had given around 100 different messages and the correct intent was recognized 85% of the times and this is appreciably high since certain domains were no trained and hence the intent was not recognized by the chatbot.

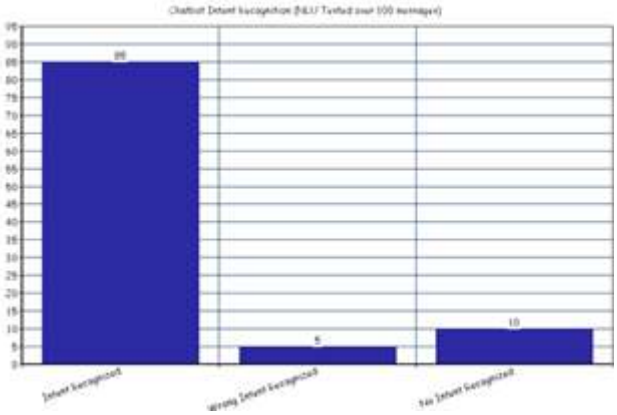

Figure 1: Chatbot Intent Recognition Accuracy

4.2 Transcript Generation and Translation Results

Table 1 below shows the correlation between bitrate and size of the video before and after extraction of audio from it in addition to depicting the accuracy of the recognition with respect to the total number of words that were uttered as part of the video. For example, the video apple_intro.mp4 has an initial size of 8.6 MB and a Bitrate of 1458 kbps. After the audio extraction, there is a reduction in these values to 6.32 MB and 1289 kbps respectively. The total time taken for the extraction is around 0.5 microseconds. We observe that the truncation in values before and after audio extraction remains consistent among all 5 videos. As for the recognition of words in the video correctly, we obtain results north of 91% and the accuracy seems to be particularly high if the number of words voiced over in the video is also high, clearly evident from video 5 where we have an accuracy of 97.5 of recognition over a sample space of 612 words, with 597 of them being recognized correctly.

Table 1: Margin specifications

S.no	Input	Initial Size (MB)	Initial Bitrate (kbps)	Size after audio extraction (MB)	Final Bitrate (kbps)	Time Taken for extraction (in seconds)	Total words	Correctly recognized words	Accuracy of recognition
1	apple_intro.mp4	8.6	1458	6.32	1289	0.0005	256	235	91.7
2	nptel_compiler.mp4	18.3	1892	15.71	1526	0.0027	398	370	92.9
3	nptel_graphics.mp4	35.3	1608	28.8	1100	0.0018	425	401	94.5
4	iit_lecture.mp4	50.8	2056	45.1	1874	0.0030	476	454	95.3
5	ted_talk_1.mp4	62.8	1787	51.9	1365	0.0017	612	597	97.5

Table 2: Subtitles Generated in Tamil

Videos	Subtitles Generated in Regional Language (Tamil)	
	Source Language (English)	Target Language (Tamil)
Video 1	Today we are going to learn an important concept in mathematics.	இன்று நாம் கணிதத்தில் ஒரு முக்கியமான கருத்தை கற்றுக் கொள்ள போகிறோம்.
Video 2	Science is something we experience every day.	விஞ்ஞானம் நாம் இதை அனுபவிக்கும் ஒன்று
Video 3	Mathematics plays a major role in all subjects.	அனைத்து பாடங்களிலும் கணிதம் முக்கிய பங்கு வகிக்கிறது.
Video 4	This circle has a diameter of 4cm.	இந்த வட்டமானது 4cm விட்டம் கொண்டது.
Video 5	The earth revolves around the sun.	சூரியனைச் சுற்றி பூமி சுழல்கிறது.

We ran the complete solution over 5 videos to check for the translation results that were being generated. The translation results that were generated were accurate for the most parts. It was not satisfactory only when poetic language with figures of speech were used. We chose 5 videos that were basically lecture video for schools. It consisted of lectures on concepts of Mathematics and science. We observed an accuracy of almost 80% where the subtitles in the target language was a grammatically correct translation ad with appropriate vocabulary used.

The major linguistic challenges that we had overcome to generate the subtitles in the target language can be seen from the line: “Mathematics plays a major role in all subjects.” which when translated to Tamil, positions of the verb and subject are changed. The line in Tamil puts “all subjects” at the starting of the sentence followed by Mathematics. This is due to the difference in structure of the languages. Therefore from the analysis we observed that this solution was generating subtitles in regional languages with linguistic accuracy.

5. Conclusion

Past few years have witnessed an extensive use of video content. An extremely large number of educational videos are available at our disposal. At the same time, certain individuals with auditory problems or people with language gap cannot understand the context/idea behind such videos because there is no text transcription available. Hence, it is important to find solutions for making such content accessible for most people. Hence, an automated system is proposed and created. For easy access, a visually appealing UI was created by using a novel method for developing a chatbot using reinforcement learning. This system is going to help in various ways i.e. social, educational. It will help students with word identification, meaning, acquisition i.e. to get complete and better understanding of the videos. It will help people with auditory problem to enjoy videos. And there is no need to manually download subtitle files of the video from the Internet which can be a cumbersome process.

6. Future Scope

Further improvement on the project would focus on:

- Providing Braille Printing facility
- Incorporating multiple languages and providing a choice for the user to select the desired language for subtitle generation
- Integration with a video player like VLC
- Plugin for YouTube
- Subtitles for Live videos/broadcast

References

[1] X. Che, S. Luo, H. Yang and C. Meinel, "Automatic Lecture Subtitle Generation and How It Helps," *2017 IEEE 17th International Conference on Advanced Learning Technologies (ICALT)*, Timisoara, 2017, pp. 34-38.

[2] R. Sridhar, S. Aravind, H. Muneerulhudhakalvathi and M. Sibi Senthur, "A hybrid approach for Discourse Segment Detection in the automatic subtitle generation of computer science lecture videos," *2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, New Delhi, 2014, pp. 284-287.

[3] Piperidis, S., Demiros, I., & Prokopidis, P. (2005). Infrastructure for a multilingual subtitle generation system.

[4] G. M. D'silva, S. Thakare, S. More and J. Kuriakose, "Real world smart chatbot for customer care using a software as a service (SaaS) architecture," *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, 2017, pp. 658-664.

[5] Klopfenstein, Lorenz & Delpriori, Saverio & Malatini, Silvia & Bogliolo, Alessandro. (2017). The Rise of Bots: A Survey of Conversational Interfaces, Patterns, and Paradigms. 555-565.

[6] Williams, J.D., Asadi, K., & Zweig, G. (2017). Hybrid Code Networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. ACL.
[7] Borisov, A., Dlougach, J., & Galinskaya, I. (2013). Yandex School of Data Analysis Machine Translation Systems for WMT13. WMT@ACL.