Graphle

PROJECT REPORT

Natural Language Processing (CSE4022)

Slot: E1 + TE1

Submitted in partial fulfilment for the award of the degree of
B. Tech in Computer Science & Engineering.

By

NAME	REG.NO
1. ISHAAN OHRI	18BCE0265
2. ANMOL PANT	18BCE0283

Under the guidance of
DR. SHARMILA BANU K

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

November 2020

DECLARATION

We hereby declare that the thesis entitled **"Graphle"** submitted by us, for the award of the degree of B.Tech. Computer Science and Engineering, is a record of bonafide work carried out by us under the supervision of DR. SHARMILA BANU K.

We further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Vellore
Date: 1-11-20 Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled **"Graphle"** submitted by **Ishaan Ohri (18BCE0265) and Anmol Pant (18BCE0283)** for the award of the degree of B.Tech. Computer Science and Engineering, is a record of bonafide work carried out by them under our supervision.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

The Project report fulfils the requirements and regulations of VIT and in my opinion meets the necessary standards for submission.

Signature of the Guide **Signature of HOD**

Internal Examiner **External Examiner**

ACKNOWLEDGEMENT

It is our pleasure to express with deep sense of gratitude to **DR. SHARMILA BANU K**, Associate Professor Grade 1, School of Computer Science and Engineering, Vellore Institute of Technology, for her constant guidance, continual encouragement, and understanding; more than all, she taught us patience in our endeavour. Our association with her is not confined to academics only, but it is a great opportunity on our part to work with an intellectual and expert in the field of Natural Language Processing.

We would like to express our gratitude to **G VISWANATHAN (Chancellor), SEKAR VISWANATHAN (Vice President), DR. ANAND A. SAMUEL (Vice-Chancellor), S NARAYANAN (Pro-Vice Chancellor)**, for providing an environment to work in and for his inspiration during the tenure of the course.

In jubilant mood we express ingeniously our whole-hearted thanks to all teaching staff and members working as limbs of our university for their not-self-centred enthusiasm coupled with timely encouragements showered on me with zeal, which prompted the acquirement of the requisite knowledge to finalize our course study successfully. We would like to thank our parents for their support.

It is indeed a pleasure to thank our friends who persuaded and encouraged us to take up and complete this task. At last but not least, we express our gratitude and appreciation to all those who have helped me directly or indirectly toward the successful completion of this project.

Place: Vellore
Date: 1-11-20 Name of the student

TABLE OF CONTENTS

ABSTRACT

"The brain processes visual information 60,000 times faster than text.

Graphle is a service build to make learning easy for autistic students. It takes in real time input from the teacher, and arranges it in a visual manner.

Autistic kids often receive education in a special way where concepts are delivered in a graphical way. Research says that visual memory is 60,000x faster than text/auditory memory. Inspired by graphical learning we created Graphle, a very effective visual learning tool for strengthening visual memory of our students.

With use of cutting-edge technology, Graphle can be of great use in remote education, where teachers can take their classroom boards to their students and allow Graphle simplify concepts for them.

MARKET AND USER RESEARCH

1 in 88
children in the world have autism

75%
of kids with autism have learning issues

94%
of autistic kids dislike school and face bullying

1. Some children become very distressed about going to school. Autistic children need more time, effort and patience to learn at a normal pace.

2. It was 'pot luck' to get an assistant or teacher who had some understanding of autism and how this could change every year.

3. Existing staff training provision was challenged by parents who felt that it was not possible to gain sufficient understanding in a one day course on 'special needs'.

WHAT HAVE WE BUILT?

- **Android App:** Built with native Android SDK and Java, it captures the speech of the teacher/instructor and converts it into text using Google's Speech-to-Text API and sends it to our Node.js backend.
- **Neural Network (Flask API):** The text received is processed and keywords are extracted from the same using a sequence to sequence encoder-decoder model. Both the models have been served with the help of Flask API which receive the text, pre-process it, extract keywords and send them to the Node.js Backend.
- **Clipboard:** Designed with HTML/CSS/JS and web sockets, it facilitates the display of the generated images in real-time, along with story recitation. The generated images can be shown onto the smart screen in a classroom or onto the individual screen of the student (Remote Classroom)-Sessions can be viewed on our website.
- **Node.js Backend:** It acts as the control centre for our service. It accepts the story snippet as text from the android application, sends it to the Flask API for keyword detection, gets the image URL (for the received keywords) from Microsoft Bing API and stores the entire backup on MongoDB. The story snippet along with the image URLs and audio clip's URL is sent via Pusher to all the devices connected to the session in real-time.

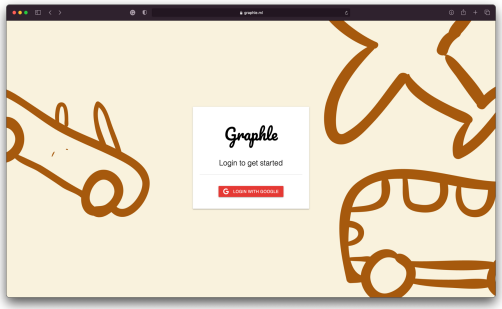
HOW IT WORKS

Our easy to use Android app helps you to start ***Graphling***with a single click

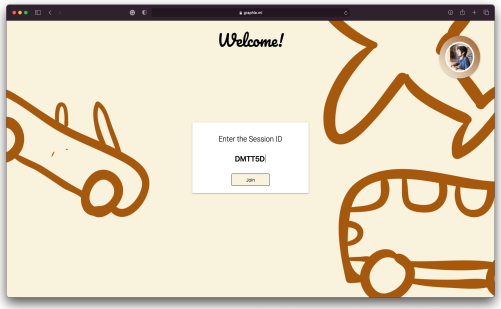
Create you ***Graphle Session***in a single click with minimal session info and rapid verification from our system

Teacher: *"A red bird is sitting on the tree..."*

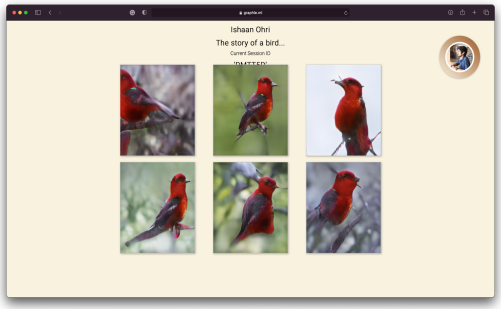
Our ***Google***powered ***Speech-to-Text*** engine send your speech to our magical image generator, which generates handcrafted image cliparts related to your teaching in real-time, thus providing you and your students with an enhanced learning experience



Thanks to ***Google OAuth***, our StoryBoard provides a single click identity management and access to your lessons.

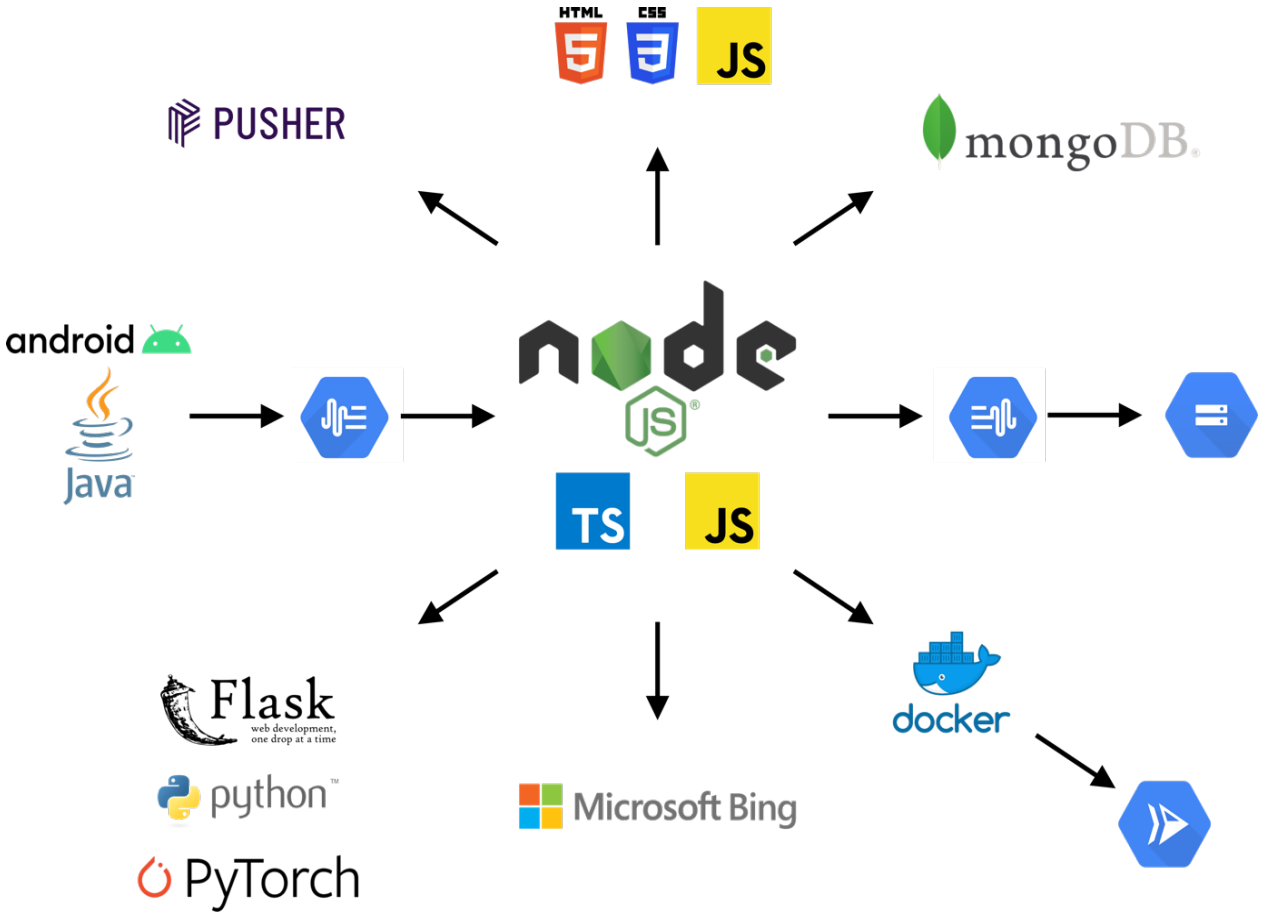


Click the button, enter your Session ID and learn like a ***PRO***



Voila!

TECH STACKS



GRAPHLE GITHUB REPOSITORY

https://github.com/IshaanOhri/NLP-Project

WEEKLY EXPERIMENTS GITHUB REPOSITORY

https://github.com/IshaanOhri/Natural-Language-Processing-Tasks.git
https://github.com/anmolpant/Natural-Language-Processing-Tasks.git

PROJECT WEBSITE

https://graphle.ml

ALGORITHM

The basis of our application lies in the sequence to sequence neural network that performs keyword extraction from the text it receives from the backend post speech to text conversion. The reason a sequence to sequence model has been used to find the most important words from a sentence instead of using the conventional pre-existing algorithms like RAKE and TF-IDF, that could also give us an indication of the importance of words in a particular block of text, is the fact that these algorithms are suitable for particularly longer pieces of text, where the frequency of a particular word has a role to play in deciding its importance. Here, we were dealing with relatively short text, spanning across one or maybe two sentences and hence the conventional approaches were rendered useless.

Hence we went about the problem by adopting a language translation using a sequence to sequence neural network approach. Where the keywords were assumed to be a separate language and an encoder-decoder model was trained to convert English -> Keywords.

The Encoder

Our encoder uses a BERT or Bidirectional Encoder Representations from Transformers model. It accepts a single element of the input sequence at each time step, processes it (performs tokenization, numericalization and forms word embeddings), collects information about that element and propagates it forward. It returns an intermediate vector that contains information about the entire input sequence to help the decoder make accurate predictions. The information contained in the input text is basically encapsulated as internal state vectors (or tensors) by the encoder and this intermediate output is then passed on to the decoder part of our model.

The Decoder

Given the entire sentence in English, it predicts the output at each time step. It generates the output phrase (the one containing the keywords) word by word,

for which it must recognise the starting and end of each sentence. Hence the output sequence is prepended with <s> and <eos> tags at the start and end respectively. The initial states of the decoder are set to the final states of the encoder as the decoder is trained to generate the output based on the information gathered by the encoder. A teacher forcing technique is used so that the input at each time step is the actual output and not the predicted output from the last time step.

The loss is lastly calculated on the predicted output and the errors are back propagated to update the model weights. The code and the dataset used can be checked out in the Graphle – Neural Network directory in the above mentioned GitHub repository.

ERRORS ENCOUNTERED

1. Lack of a proper dataset to work on

The very first problem we were faced with was the absence of a dataset containing single line sentences and the keywords present in them, on which we could train our model on. After trying multiple online datasets but to no avail, to address this predicament, we performed web scraping on a popular blogging website (https://lifehacker.com/) and scrapped the titles of their articles as sentences and the SEO tags enlisted in the article as their keywords. The SEO tags which were not present in the title line were then deleted. This is how we obtained the dataset to train our model on.

2. Keyword Extraction from Short Text

After obtaining a dataset, the next challenge was to figure out how to extract keywords from a given sentence as the conventional RAKE and TF-IDF based approaches were heavily reliant on the frequency of a particular word in a piece of text, to figure out its importance. For that we decided to use a sequence to sequence language translation based approach, assuming the English sentences to be our base language and the keywords as a separate language altogether that lacked supporting verbs, conjunctions, etc. The keywords were our target

language and a Seq2Seq translation model was implemented to perform keyword extraction.

3. Encoder

Initially we tried to code an encoder to convert our English sentences to tensors that could then be used by our decoder for keyword extraction but the primary problem we faced was the fact that due to the sentences having different lengths, the tensors had no fixed length and hence the decoder could not interpret the output of the encoder, let alone make sense and act on it. To solve this we decided to use the BERT encoder, and then fine tune it. As BERT, given by Google, uses state of the art pre-processing techniques, it finds the longest tensor. All the other tensors are prepended with zeroes so that each and every tensor is of the same length, which solved our problem of the encoder returning tensors of irregular length.

4. Using channels and web sockets to ensure real time service

The issue was to ensure that the service works in real time, i.e. the fetched URL from the Bing API along with the other required metadata is sent across to all the connected screens in a synchronous manner. It is required that as the instructor speaks, the required data is sent immediately without any significant latency and ensuring that it's sent to each and every connected screen. In order to tackle the problem, we used web sockets implemented via an industrial standard service, PUSHER. It ensures real time transmission of data to channels via custom channel IDs.

5. Proper authentication of connecting users

Every user who is attempting to connect to the service must be authenticated from the backend in-order to ensure proper security. No random user should be able to login into the website nor should be able to join any of the existing sessions. For login authentication, we use industrial standard Google OAuth, which issues proper tokens to the user and every request made by the user is authenticated by the backend. For connecting to any existing service, every user is issued a joining token by the backend which is limited to the scope of the session.

These were the major errors/roadblocks we faced whilst working on the project.

REFERENCES

1. https://pytorch.org/docs/stable/index.html
2. https://www.crummy.com/software/BeautifulSoup/bs4/doc/
3. https://towardsdatascience.com/extracting-keywords-from-short-text-fce39157166b
4. https://lifehacker.com/
5. https://monkeylearn.com/keyword-extraction/
6. https://www.youtube.com/watch?v=ojUGNyWChHg
7. https://www.youtube.com/watch?v=v9X3j-2p4yA
8. https://www.youtube.com/watch?v=Lbi34KzFUas
9. https://github.com/spro/practical-pytorch
10. https://medium.com/dair-ai/neural-machine-translation-with-attention-using-pytorch-a66523f1669f
11. https://www.rileynwong.com/blog/2019/4/3/implementing-a-seq2seq-neural-network-with-attention-for-machine-translation-from-scratch-using-pytorch
12. https://github.com/spro/practical-pytorch/issues/135
13. https://github.com/spro/practical-pytorch/issues/108
14. https://github.com/JRC1995/TextRank-Keyword-Extraction
15. https://medium.com/dissecting-bert/dissecting-bert-part-1-d3c3d495cdb3
16. https://medium.com/dissecting-bert/dissecting-bert-appendix-the-decoder-3b86f66b0e5f
17. https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346
18. https://link.medium.com/l5O6jf093ab
19. https://pypi.org/project/pytorch-pretrained-bert/0.4.0/
20. https://pytorch.org/tutorials/beginner/transformer_tutorial.html
21. https://discuss.pytorch.org/t/why-cant-i-import-masked-cross-entropy/11737
22. https://pytorch.org/tutorials/beginner/saving_loading_models.html
23. https://pusher.com/docs/channels/using_channels/presence-channels

24. https://pusher.com/docs/channels/using_channels/authorized-connections
25. https://pusher.com/docs/channels/using_channels/connection
26. https://docs.docker.com/ci-cd/github-actions/
27. https://mongoosejs.com/docs/api.html#model_Model.find
28. https://dev.to/chuck_huey/setup-a-typescript-project-with-eslint-prettier-editorconfig-and-husky-13aa