
# *"A cloud solution for analysing patterns in NGO projects"*

**Team Number**: Team 36

**Names**: Rachel Mattoo, Mark Anson, Yansong Liu

**Date:** 13/03/2020

**Overview of progress**

After speaking to the NLP specialist, in our first meeting, we decided that the best software to use was RoBERTa. This is because although ALBERT has a higher accuracy of 89.731, according to the SQuAD ratings, RoBERTa is well-documented. We also decided to not pursue AllenNLP because it has a lower SQuAD rating. However, after discovering Hugging Face at the second meeting, we changed our mind back to experimenting with different models and accepted that ALBERT was the most likely choice.

Database 2 (Rachel – also part of my individual report):

SQL Database 2 Design: Whereas Database 1 was created based on guidelines from Matt, who was a client for another second-year team we were collaborating with, Database 2 had no such guidelines. Database 2, also known as the "knowledgebase", is based on our dataset of publicly available annual NGO reports. In order to create a database schema for Database 2, I first analysed multiple NGO reports to determine how each report was structured. The main topics highlighted were: Staff, NGO information, Sponsors, Project information and Finance. These main topics were used to sketch an initial entity-relationship (ER) diagram with just table names. Once I had determined the general structure of the database, I then analysed around ten random reports to determine how to breakdown each topic into sub-topics, which would then represent the respective relations between the tables. For example, Project information could be further broken down into Project Finance, Project Impact and Project Geo Info. Each sup-topic was further analysed to determine the fields. Designing the ER diagram required multiple sketches and revisions to ensure a maximum amount of information could be captured by the database.

Database 2 SQL Query: Once I had finalised the ER diagram, I wrote the SQL queries to create the database. It would have been wiser not to write the whole SQL script at once and instead execute it per table. However, since I wrote the SQL script at once, multiple syntax errors resulted in initial building failures. The script itself also required multiple iterations to ensure data types and length limits for varchar data types were set correctly. For example, NGO address required a longer length limit than NGO name. One error I encountered during integration testing for the ALBERT algorithm and Database 2 was that all the fields, apart from the primary/foreign keys, were not set to NULL. These fields were changed to be default NULL because it could not be guaranteed that the ALBERT algorithm would return a value. Another error encountered was that the ALBERT system could only return string values, so each field data type was changed to only store varchar.

Training/Finetuning:

The large part of the first week of these two weeks was spent on training different models a different number of times to determine if fine-tuning our model on SQuAD (a huge pre-existing reading comprehension data set of over 100,000 questions) would render a higher accuracy. Because this process was so resource and time intensive, due to time constraints we decided to stop finetuning and pursue the ALBERT model, which had the highest "exact" (answer to question has exact correct answer) accuracy.

This decision had not been taken lightly and we had tried multiple things to speed up the accuracy such as using multiple powerful GPUs to speed up our progress and leaving our program running overnight at times. We also had to consult the computer science department technology support multiple times due to frequent permission errors, which also delayed our progress. The fact we ran out of time makes sense because researchers at Google and Facebook often use extremely powerful cloud TPUs to train their BERT models, so we did not expect to be able to fine-tune our model.

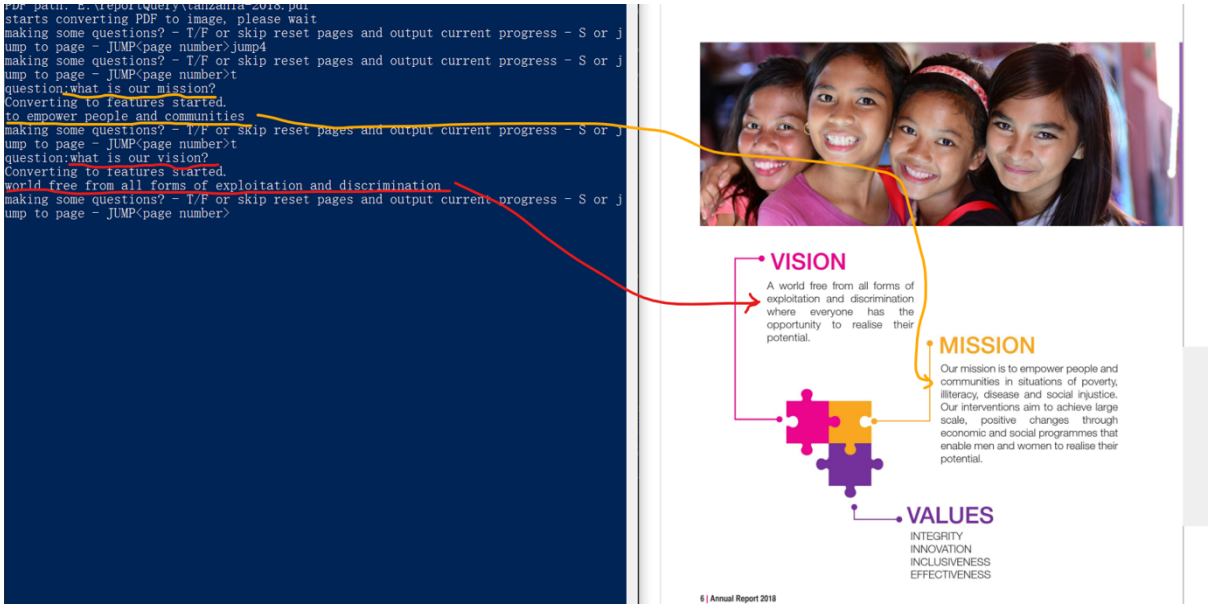
<u>BERT algorithm in progress:</u>

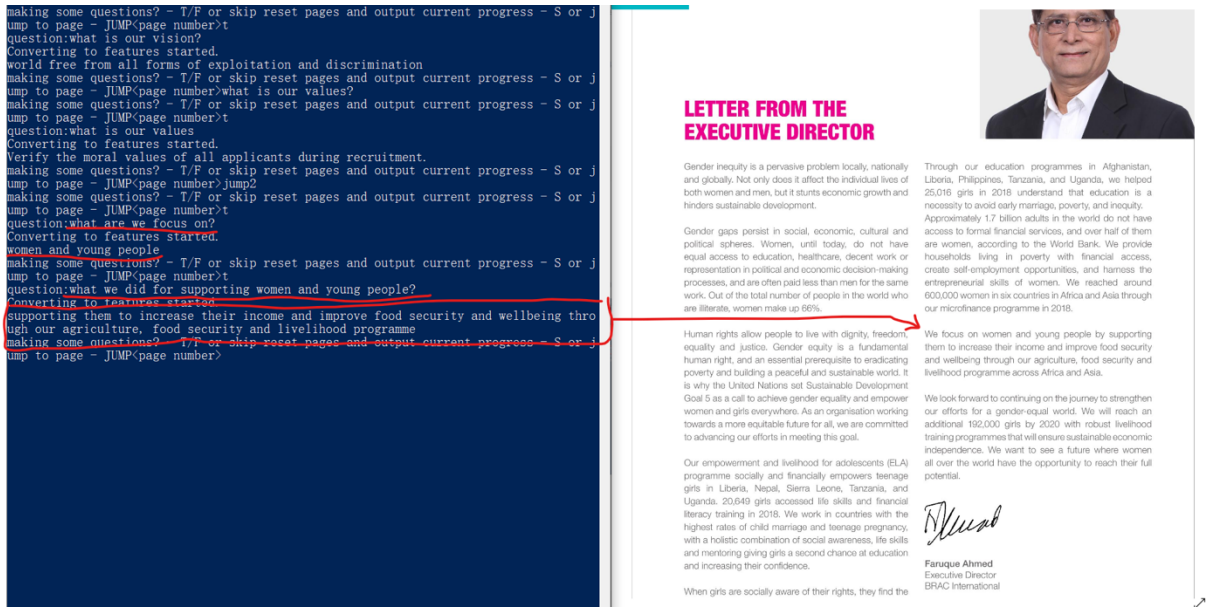
Since we managed to implement ALBERT on one page of a pdf, so essentially, we could ask a question to the text and the model would return the answer for us based on the text. Here are a few examples we demonstrated to our TA Sheena to show how BERT models allow us to query text accurately:

**<u>List of tasks completed and whether project is running on time</u>**

The project is running on time and we are on track to meet the deadline.

- Research on different BERT models (documented on website)
- Experimentation on fine-tuning different BERT models
    - RoBERTa, ALBERT, BERT
    - Fine-tuned each model a different number of times (1,2,3)
        - To check if there was an improvement in accuracy
        - On the SQuAD dataset






**Plan for coming 2 weeks**

- Complete BERT algorithm
  - This is the pdf extraction tool
  - As per our user requirements
- Link BERT with Database 2
  - Using SQL database connections
- Successfully deploy project
- Start on documentation for BERT algorithm
- Finish extra deliverables
  - On top of source code
- Finish Database 1
  - Ask master's team if they require any final iterations