An Overview of the Dataset

- 162 Queries and 1 Base Document
- Queries given in (.CSV) format and Base Document in (.pdf)
- Base Document has multiple tables and mathematical formulae.
- Queries did not have tabular entries but a few had linear formulas with special symbols like gamma, alpha etc.

Formulation of Approach

- Representing features in the form of vectors for both; queries and base document.
- Calculating the distance between features for each query with the base document.

Modelling

There are two type of similarities possible in case of matching the texts,

- **Lexical Similarity** : This measures the closeness of occurrence of words.
  Jaccard Similarity calculates the similarity (Jaccard index) as size of intersection divided by size of the union of two sets.
- **Semantic Similarity** : This measures the texts similarity based on the 'meanings' of the sentences. Word embedding is used to create the feature vector and then cosine similarity is used to calculate the matching score. While the match percentage might be different from the ground truth due to methodology or size of dataset, the comparative ratio of most similar and dissimilar queries should be the same.

For example for P161 query, both the above methods give a high matching score.

Contents & Instructions

- Semantic_Similarity.ipynb - Ipython Notebook trained on Google Colab,
  `To reproduce the results, select 'Runtime' and 'Run all' the cells. You will need the 'cc.en.300.bin' file. Please download it using a bash terminal (run ./download_model.py en) and set the paths accordingly.`
  `I have included the colab notebook link as well in the email.`
- Semantic_Similarity.csv - (.CSV) file with results in a separate column.
- Lexical_Similarity.ipynb - Python Notebook trained on Jupyter Notebook
  `Open using Jupyter Notebook and 'Kernel - Restart and Run All'`
- Lexical_Similarity.csv - (.CSV) file with results in a separate column.