JD CREAToR:

This is a modular structure of the implementation that will be followed.

Module -1 : divides into sections

Module – 2 : construction of dictionary from database

moule – 3: storage of these modules in database.

Module – 4: preparation data for statistical inferencing

module – 5: develop an aggregated matrix establishing features between tags found

module – 6: implement statistical inferencing on matrix made

module – 7: give score to each ( tag X tag )

module – 8: display results

module – 9 implement gradient descent for retraining

# Module 1

## Objective:

This module will use the entire job description document and segregate them into different sections such that all the sections can be differently analysed for data.

## input:

Input to this module will be a bunch of JD’s in the form of

* text files,
* unformatted plain text
* one job description in one file, all stored in a folder.

## Output:

output will be different sections of a JD stored in different columns of a spreadsheet of which the first column will be the complete JD text and next few columns represent the sections in the JD

* sections will be limited to a few specific names,
* these sections will be identified on the content they hold
* if something in a JD if not identified it will be pushed to a column with column heading as (Extra)

# Module 2

## objecive:

This module will create a dictionary that will be Looked upon each time a word in a JD is read for comparison.

## input:

Excel sheet where columns can be as many but one column should have all the words that are possible to occur in a job description and also are keywords.

## output:

A giant dictionary having structure of a recursive dictionary (dictionary in a dictionary) where each individual word is a key and its value is again a dictionary with their key as its preceding word.

# module 3

## objective:

Module 2 will have a few instances which will run the same code but for different sections and will save their tags differently. Module 3 basically will match the JD’s that are having words exactly same, short forms or acronyms of the words present in the dictionary of words made from module 2.

## input:

* JD sections one by one, these sections will be read from the data that has been segregated in module-1.
* Dictionary made in section 2.

## output:

* Input data with their corresponding tags will be stored in the Database.
* User interface will be created which will show the text of the jd that has been processed on a canvas or a text box on the screen.
  + Words marked in RED – words that perfectly matched the ones in the DB
  + Words marked in BLUE – words that have case issue (lower or upper) else matched correctly
  + Rest of the words will be unmatched or will be stop words.

## future additions could be:

* Words occurring frequently still are unmatched can be prompted on the screen and manual entry of those words can be made in the dictionary (i.e. MODULE – 2)
* User can pic words or strings on their own and add them as input to MODULE-2.

# module 4

## objective:

Develop clean data for statistical inferencing of data collected.

## input:

JD text along with corresponding tags that were collected in module 3.

## output:

Matrix with **Rows-** JD text and **columns–** Tags that were found yet (all of them)

The value of a column for a particular row will be the occurrence of that TAG (column heading) in the JD being read.

# module 5

## objective:

Developing an aggregate matrix that establishes a relationship between two tags, or provides with some statistical data that helps us find a relevance score for all pair of tags found.

## input:

Output of module 4 will be the input to this module.

## output:

A 3-dimensional matrix where rows and columns are tags initially initialized to zero. And the third, the depth axis will be the distance from row[word] from column[Word] in a JD.

The value of the depth axis at depth **(D)** will be incremented when an occurrence of the two words with distance **(D)** occur.

## Future additions could be:

Many changes in this module are possible, its pure statistics and any method that will be found more reliable for measuring the closeness can be implemented in this module.

# module 6

## objective:

This module will cluster the input data stream/ tag stream into various groups and predictions for those groups will be differently looked for.

For example:

* The JD creator adds 5 words from the core software technology domain, and then he adds a word which is from a different domain, that holds less association with the software ones, suppose (marketing) or (sales).
* Then this module will cluster those words in separate groups and look for their predictions separately.
* Suppose 10 predictions are required in total, then it will suggest 5 from domain-1 and 5 from domain-2.

## input:

The input comes from the UI front-end.

## Output:

Its output is passes to MODULE 8 for predictions.

Output will most likely be in the form of list of list. Where each list in the mega list represent 1 cluster.

# module 7

## objective:

Scoring of the tags in association with one another to find associations and alliances of tags.

This module will be used to make actual predictions when a word will be entered by the user, this module will suggest the maximum scoring individuals(tags) as the predicted output.

## input:

* Statistically processed data matrix on which further operation will be made, (either static or dynamic).
* Matrix formed in MODULE 4 can also be required depending on the scoring formula that will be derived.

## output:

A matrix with rows as tags and columns as tags and value of matrix[row][column] = (percentage association)/ nearness score.

# module 8

## objective:

This module will use the output of MODULE 7 and MODULE 6 for processing predictions.

The words in list provided by module 6 will be read and row of that tag will be looked upon in the matrix of MODULE 7, the output will be sorted on the basis of score in the matrix.

In case of more than one word in the list, the scores will be added and then the results will be presented.

## input:

Output of MODULE 6, MODULE 7

## output:

List of predictions where the top word in the list represent highest scoring tag.

# module 9:

**THE AI MODULE, YET TO BE CONSTRUCTED.**