A Noval Approach To Link Posts in Stack overflow

Prashanth Kumar Daram
Student Id: 1116540
Department Of Computer Science
Lakehead University

I. Abstract

***Abstract*—Posts in stack overflow make it easier to share information. The numerous number of questions and answers posted in stack overflow clearly shows the popularity and success of this learning method. This provides us a good source for researching the properties of stack overflow posts. Linking posts to potentially related posts i.e, issue knowledge acquisition will provide developers with more relevant and useful information when they searching and resolving issues. However, acquiring all the related posts is a challenging task, because real-world acquiring is time-consuming, and is mainly dependent on individual developer's expertise and skills. As a result, automatically acquiring similar posts is more efficient that can improve development performance. Identifying all the similar and related posts are formulated as guideline problems in this paper. To solve this problem, I developed three models, TF-IDF and deep learning techniques, i.e., Word Embedding and Document Embedding to solve the problem. In this project, both internal and external links analysis is also performed**

keywords*—*Stack Overflow, Unanswered Posts, Deep Learning, TF-IDF, Word Embedding, Document Embedding, Knowledge Acquisition

II. Introduction

Stack Overflow is a question-and-answer web-based platform for new, experienced, and enthusiast programmers. The website is privately owned and founded in 2008 by Jeff Atwood and Joel Spolsky as the Stack Exchange Network flagship site. This site consists of questions and answers on a variety of programming languages from different users and was developed as a more accessible alternative to previous question-and-answer platforms like Experts-Exchange.

This website acts as one good interface for users to get the answers for their posted questions and post the answers to questions and they can vote the answers up or down based on the content of the answer and update them in a wiki or Reddit in a similar way. Depending on their valuable contributions, users can gain reputation points and badges in a stack overflow. For example, an individual having an "up" vote on a question or an answer is awarded ten reputation points and earns badges for their contributions to the conventional QA platform. A sample stack overflow post is shown in figure-1.

The users can gain access to new features such as voting, commenting, and even editing other people's content based on reputation points and badges. Knowledge acquisition, which is defined as linking/acquiring similar posts is essential for developers to quickly and efficiently resolve the issues. In this paper, the pair of posts considered as linked based on the association such as dependent, equivalent, or referenced. The developers will review or access a new post after the user submitted it. The other users can look the similar posts to provide information on what they consider valuable resources and information. The post would eventually be closed with aid of relevant post information and discussions. Depending on the experience and skills of individual developers, this manual acquisition process could take a long time. The automatic tools for acquiring similar posts in stack overflow are more beneficial.

GitHub is a similar platform to Stack Overflow. GitHub provides the idea of social coding, helping developers to share knowledge, cooperate, and support their projects in a developer-friendly world. Social coding also contributes to the re-usability of code as well as the problem resolution process. Developers can efficiently participate in the discussion and contribute their work in topic reporting. As a result, various developers often reported issues and addressed them at different times, it is not unusual to see that two issues contain similar details that can be exchanged to aid issue resolution. Many open issues in a project are connected to relevant issues through URL referencing during the topic conversation, one of the manifestations.

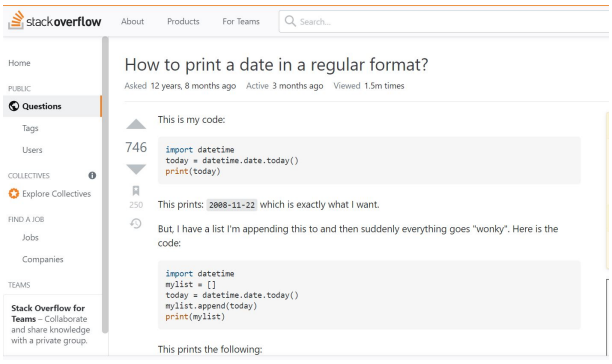

Fig. 1. Stack Overflow Post

978-1-7281-4387-3/19/$31.00 © 2019 IEEE

III. Related Work

The key issue online coding platforms facing is extracting better ways to maintain and expand the shared resources. The online coding platform designers, developers, and owners are facing a plethora of problems for managing their massive resources to ever-growing populations. The institutional rules on different aspects of a network may have an unpredictable impact on long-term outcomes. Moreover, the difficulty of this environment is shown that even minor site design decisions will have a significant impact on participation patterns. The two interconnected challenges these online flat forms facing are sustainability and representativity.

In [1], they explain linker, a novel method for automatically acquiring related issue knowledge in GitHub projects. iLinker uses deep deep learning methods with conventional information retrieval techniques (TF-IDF), word embedding, and document embedding. They compare iLinker to a state-of-the-art methodology in conventional bug tracking systems called Next Bug.

Most software developers depend on code examples to learn how to use an API. Finding the characteristics of learning API requires an example. To improve the appropriateness of these learning mechanisms, it is necessary to identify the features of successful examples. In [2], the authors conducted research on questions and answers posted on a stack overflow, a popular programming QA website, which helps to address this problem. The answers posted in stack overflow are voted by the users, showing which one they find helpful. They found that the explanations containing examples are most valuable. Their studies have shown how API document examples are essential and it shows how API documents and examples can be created and evolved.

In [3], the authors conduct research on how important links are in source code comments. They implemented a mixed-method approach to classify the link's goals, aims, degradation, and evolutionary aspects in a large-scale analysis of about 9.6 million links to determine their prevalence. The authors discovered that links are been widespread in source code libraries. The goals of a link include licenses, software home pages, and specs, often links are often used to include metadata or attribution. The links are modified occasionally, and their targets will change. Almost 10% of links in source code comments are no longer active. They then sent a request to open-source software repositories for fixing the link. The results of their study suggest that links in the source code comments may be weak and their work suggests a way for further research into these issues.

In [4], they explore on what are the major issues on stack overflow and the reasons that could be related to new contributors move from posting questions to posting answers. User's personal characteristics such as gender, tenure, geographic area, and the characteristics of their sub-community in which they are mostly involved, such as the scale and frequency of negative social reviews have a huge impact on their willingness to post the responses. Their paper takes a first look at the barriers and threats to online user contributions by assessing and modeling these relationships.

In [5], the author's work on mutual information sharing between Android Issue Tracker and Stack Overflow posts is an example. Using the internal citation graph they suggest an automated approach that combines semantic similarities with temporal locality between Android issues and Stack Overflow posts. Their method looks at internal citations in forums to see if there are any closely associated posts or problems. It then uses the temporal similarities between issues and posts to rate associations. Extensive testing shows that their method has a precision of 62.51 percent for top 10 suggestions when recommending Stack Overflow posts to Android issues and a precision of 66.83 percent when recommending Stack Overflow posts to android issues.

IV. Dataset

In this paper, the data is taken from the Stack Exchange link. The stack exchange link contains data for all the stack exchange websites. we downloaded only stack exchange data which is required for this project. The data set contains stack overflow comments, stack overflow post history, stack overflow post links. It is shown in fig 2. Once we downloaded the dataset, we used the eTree parse to convert the XML files to CSV files. The Posts dataset consists of 30000 posts that are shared in a stack overflow. The main model is trained with 30000 posts for efficient results. Each post in the Posts dataset consists of 6 columns i.e. ID, Body, Title, PostTypeId, Tags, CreationDate. It is shown in fig 3.

sqa.stackexchange.com.7z (View Contents)	07-Jun-2021 13:23	31.4M
stackapps.com.7z (View Contents)	07-Jun-2021 13:23	12.9M
stackexchange_archive.torrent	07-Jun-2021 23:34	458.7K
stackexchange_files.xml	07-Jun-2021 23:35	103.6K
stackexchange_meta.sqlite	07-Jun-2021 18:53	1.1M
stackexchange_meta.xml	07-Jun-2021 23:34	3.4K
stackexchange_reviews.xml	07-Jun-2021 23:35	17.0K
stackoverflow.com-Badges.7z (View Contents)	07-Jun-2021 13:23	288.5M
stackoverflow.com-Comments.7z (View Contents)	07-Jun-2021 14:21	4.7G
stackoverflow.com-PostHistory.7z (View Contents)	07-Jun-2021 17:50	29.2G
stackoverflow.com-PostLinks.7z (View Contents)	07-Jun-2021 13:24	100.8M
stackoverflow.com-Posts.7z (View Contents)	07-Jun-2021 15:28	16.6G
stackoverflow.com-Tags.7z (View Contents)	07-Jun-2021 13:24	857.7K
stackoverflow.com-Users.7z (View Contents)	07-Jun-2021 13:25	733.0M
stackoverflow.com-Votes.7z (View Contents)	07-Jun-2021 14:22	1.3G
stats.meta.stackexchange.com.7z (View Contents)	07-Jun-2021 13:25	8.0M
stats.stackexchange.com.7z (View Contents)	07-Jun-2021 13:25	442.1M
stellar.meta.stackexchange.com.7z (View Contents)	07-Jun-2021 13:25	90.0K
stellar.stackexchange.com.7z (View Contents)	07-Jun-2021 13:25	2.4M

Fig. 2. Data set for the model

	Id	Body	Title	PostTypeId	Tags	CreationDate
0	337	<p>I am about to build a piece of a project th...	XML Processing in Python	1	<python><xml>	2008-08-02T03:35:55.697
1	469	<p>I am using the Photoshop's javascript API i...	How can I find the full path to a font from it...	1	<python><macos><fonts><photoshop>	2008-08-02T15:11:16.430
2	502	<p>I have a cross-platform (Python) applicatio...	Get a preview JPEG of a PDF on Windows?	1	<python><windows><image><pdf>	2008-08-02T17:01:58.500
3	535	<p>I am starting to work on a hobby project wi...	Continuous Integration System for a Python Cod...	1	<python><continuous-integration><extreme-progr...	2008-08-02T18:43:54.787
4	594	<p>There are several ways to iterate over a re...	cx_Oracle: How do I iterate over a result set?	1	<python><sql><database><oracle><cx-oracle>	2008-08-03T01:15:08.507

Fig. 3. Different Columns in Post Dataset

V. Methodology

A. RQ-1:

Analysis of internal and external links in stack overflow posts (questions, comments, and answers)

The first research question was divided into three parts-

- The first one is about analyzing the percentage of links (both internal and external) in Stack Overflow posts. The results are plotted in a pie chart.
  - **Internal Link**: Internal Link is a hyperlink on a web page to another web page or resource on same website or domain
  - **External Link**: External Link is a hyperlink on a web page to another web page or resource on different website or domain

  As we can see in Fig 4., most percentages of the links are present in Answers (58.5%) followed by questions (22%) and comments (19.6%). It is understandable because users while answering the posts often refer to other links which may be internal to stack overflow or some other external websites

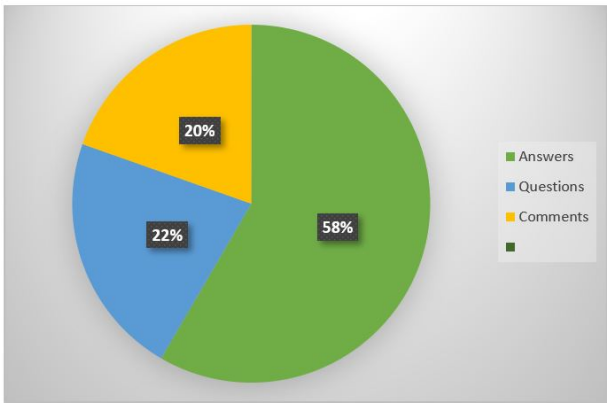

Fig. 4. Percentage of Links in SO Posts

- The second one is about the top domains in external links. We have collected all the external links in Questions, Answers, and Comments. Then using the 'urllib' library, we have extracted domains from the text input. Then top twenty domains have been filtered out and projected in a bar graph. As shown in Fig 5., the topmost domains were the Microsoft and ASP domains, whereas the least number of links were among MySQL and Codeproject domains. Microsoft domain provides documentation to a lot of other software such as Visual Studio, SQL Server, Azure, and a lot more. It may be the reason for more number of references in stack overflow posts
- The third one is about the percentage of internal and external links in Questions, Answers, and Comments. We extracted the links from each post and categorized them into 2 categories- internal and external. Then we plotted 3 bar charts for each category, i.e., Questions, Answers, and Comments. The following figures (Fig 6, Fig 7, Fig 8) shows the output. It can be seen that externals links are more in stack overflow posts compared to the internal posts

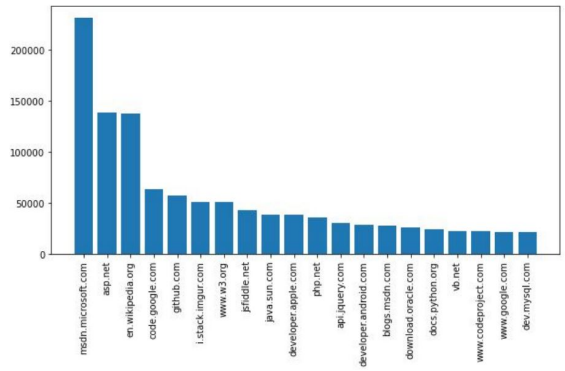

Fig. 5. Top 20 External Links

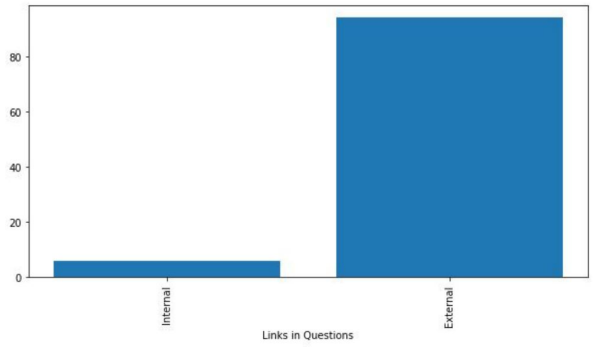

Fig. 6. Percentage of Links in Questions

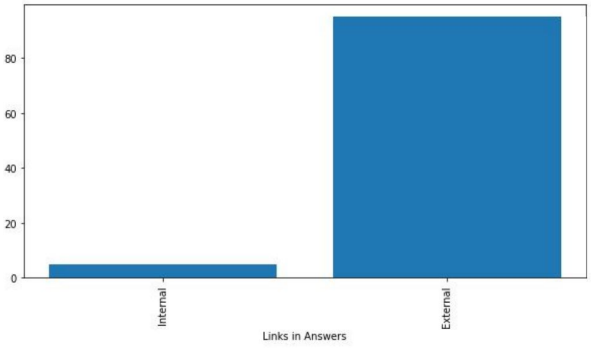

Fig. 7. Percentage of Links in Answers

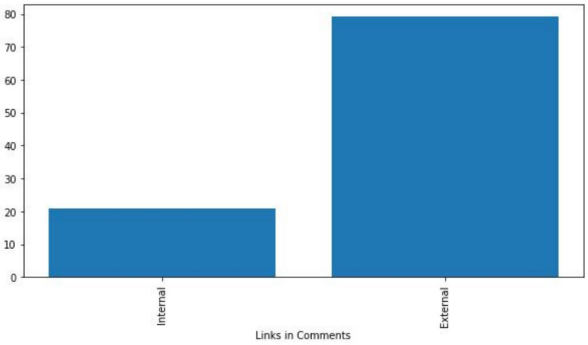

Fig. 8. Percentage of Links in Comments

B. RQ-2

Analysis of time taken for linking related posts in stack overflow

The second research question is divided into two parts-

- We have calculated the percentage of links acquired within a day for each post in the first part. We then divided the output into the following five categories-
  - less than one hour
  - 1-3 hours
  - 3-6 hours
  - 6-12 hours
  - greater than 12 hours

  The output is plotted in a bar graph as shown in Fig 9. From the image, we can infer that most of the links for posts are acquired within 1 hr, followed by the "more than 12 hrs category". On the other hand, the least percentage of links acquired was between the "6 to 12 hrs category".

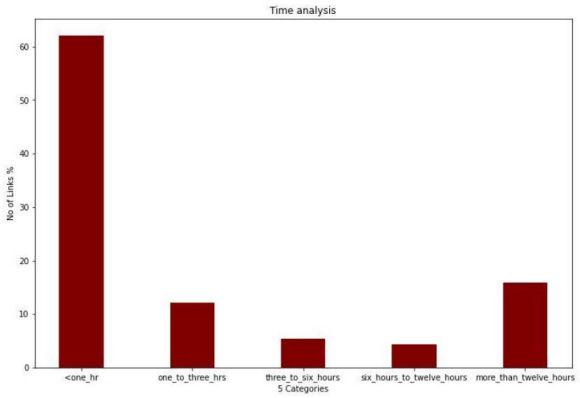

Fig. 9. Percentage of Links acquired in part-1

- In the second part of the question, we have calculated the percentage of links acquired in the whole timeline of each post. We then divided the output into the following five categories
  - less than one day
  - 1-7 days
  - 7 days - 2 weeks
  - 2 weeks - 4 weeks
  - greater than 1 month

  Here also, most links are acquired within 1 day followed by 1-7 days category as seen in Fig 10. We have excluded the less than one day category to show the remaining categories better, as shown in Fig 11

C. RQ-3

Building a model that returns the topmost related posts for the unanswered questions in Stack Overflow.

The architecture of the model for the third research question is shown in Fig 12.

- The third research question mainly focuses on developing an automated model that would recommend related posts

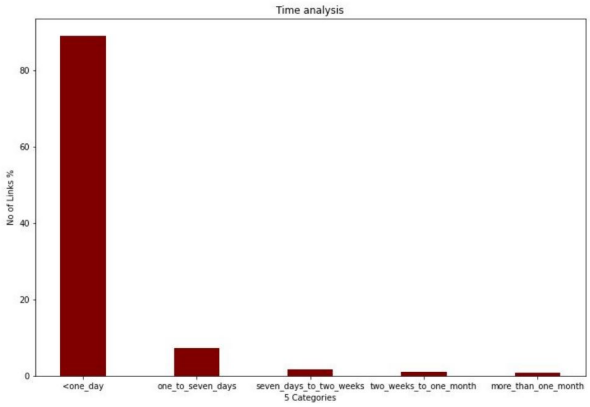

Fig. 10. Percentage of Links acquired in part-2a

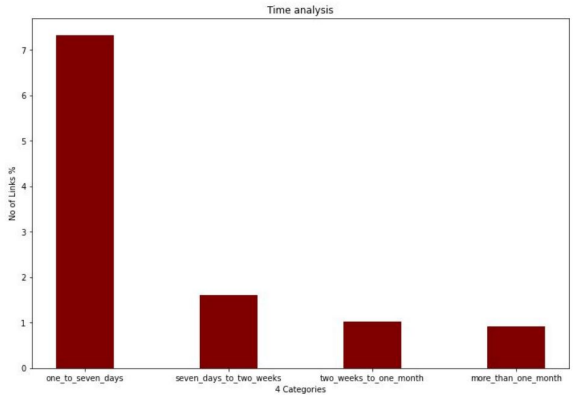

Fig. 11. Percentage of Links acquired in part-2b

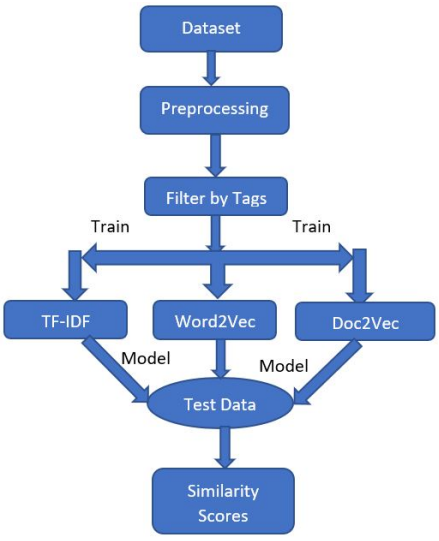

Fig. 12. Architecture of model

to unanswered questions on stack overflow. In this model, for training and testing, we used the questions only related to the python tag specifically. It is because Python is one of the faster-growing languages over the past years and the number of people using this language has been increasing dramatically over the years. It can be shown in fig 13. In this paper, we implement the model only for pythons tags and we can see how well the model can perform for a single tag, and later we can implement it for all other languages.

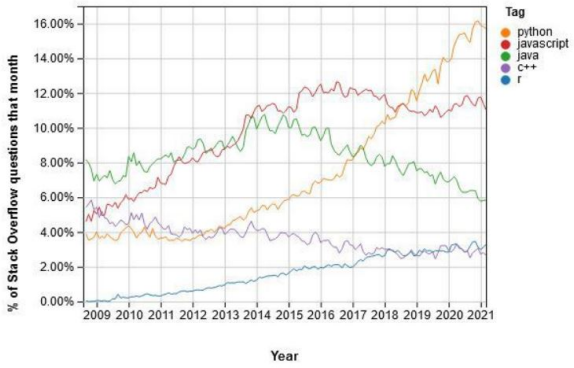

Fig. 13. Trends of Languages

VI. Proposed Model

Main Steps Involved in this Model:

- **Pre-Setup (Importing all the required packages)**
- **Dataset Processing**
- **Building the model**

A. Pre-Setup

Firstly, the coding part has done in the Google CO-LAB platform, which is provided by Google. The Google Co-lab usually uses GPU because it has a very fast processing time. Secondly, import all the required libraries that are necessary to work with this model. The libraries used in this model are Numpy, Pandas, Sklearn, genism.

- **Pandas**: Pandas is used for reading the CSV dataset
- **Sklearn**: It is used for calculating TF-IDF vector values and finding cosine similarity
- **Numpy**: It is used for performing mathematical calculations on multi-dimensional arrays
- **NLTK**: It is called as Natural Language Tool Kit. NLTK is written in Python language. It is a large number of libraries and programs for statistical processing of Natural Language
- **Stop Words**: Stop words are English common words that do not add anything to our sentiment frequencies. Stop words library from nltk helps to remove stop words in our document
- **WordNetLemmatizer**: Lemmatization is the process of converting a word into a root form. Wordnet is a large and publicly available lexical database for English. It establishes relationships between the words which are semantically structured
- **Porterstemmer**: Stemming is a process of converting a word into root form by removing suffixes and prefixes such as -ed, -ize, -s,-de, mis. It is a normalization technique. Porterstemmer is an nltk library that helps to perform stemming on a given text
- **Tfidf Vectorizer**: It performs the transformation of text into feature vectors that can be used as input for the estimator

B. Dataset Processing

- **Text Normalization**: The First step is to import all the libraries required for stop words removal, punctuation removal, stemming, and lemmatization. We import stop words library from the nltk corpus to remove the stop words. Next, we import the Punkt library from nltk corpous to remove punctuations. Next, we import WordNetLemmatizer, PorterStemmer. Lemmatization and Stemming are the normalization techniques to covert the word into its base/root form. Stemming converts the word into its base form by removing the suffixes and prefixes of the word. Lemmatization uses dictionary of words for converting the word into its root form. In the second step, after performing text normalization to the body and title of each post. We append two more columns tokenized_title and combined text. The combined text column is a combination of both the data from the body and title column, whereas the tokenized_title is a combination of pre-processed text from both body and title columns. It can be shown in fig-14.

	Id	Body	Title	PostTypeId	Tags	CreationDate	tokenized_title	combined_text
0	337	<p>I am about to build a piece of a project th...	XML Processing in Python	1	<python><xml>	2008-08-02T03:35:55.697	[xml, processing, python, p, build, piece, pro...	XML Processing in Python <p>I am about to buil...
1	469	<p>I am using the Photoshop's javascript API t...	How can I find the full path to a font from it...	1	<python><macos><fonts><photoshop>	2008-08-02T15:11:16.430	[find, full, path, font, display, name, mac, p...	How can I find the full path to a font from it...
2	502	<p>I have a cross-platform (Python) applicatio...	Get a preview JPEG of a PDF on Windows?	1	<python><windows><image><pdf>	2008-08-02T17:01:58.500	[get, preview, jpeg, pdf, window, p, cross-pla...	Get a preview JPEG of a PDF on Windows? <p>I h...
3	535	<p>I am starting to work on a hobby project wi...	Continuous Integration System for a Python Cod...	1	<python><continuous-integration><extreme-progr...	2008-08-02T18:43:54.787	[continuous, integration, system, python, code...	Continuous Integration System for a Python Cod...
4	594	<p>There are several ways to iterate over a re...	cx_Oracle: How do I iterate over a result set?	1	<python><sql><database><oracle><cx-oracle>	2008-08-03T01:15:08.507	[cx_oracle, iterate, result, set, p, several, ...	cx_Oracle: How do I iterate over a result set?...

Fig. 14. Pre-Processed Combined Text

C. Building the model

In this project, we used three different models

- **TF-IDF:** TF-IDF stands for Term frequency-inverse document frequency. It used in information retrieval systems and also content-based filtering mechanisms (such as a content-based recommender)
- **Word2Vec:** The word2Vec algorithm learns word associations from a vast corpus of text using a neural network interface. Once learned, a model like this can identify synonyms and recommend alternate terms for a sentence. The Continuous Bag-of-Words model and the Skip-gram model are the two main models described in word2Vec
- **Doc2Vec**: Doc2Vec is an NLP tool that creates a numeric representation of a document. It represents documents

as a vector and is a generalization of the word2Vec method. PV-DM and PVDBOW are two main methods in Doc2Vec.

Parameters used in TF-IDF [6]:

- **analyzer:** This parameter is used to choose whether the feature should be made of word or character n-grams.
- **ngram range (tuple (min n, max n), default=(1, 1)):** This parameter decides the lower and upper boundary of the range of n-values for different n-grams to be extracted. All values of n such that min n¡= n ¡= max n will be used. For example, an ngram range of (1, 1) means only unigrams, and (2, 2) means only bigrams. It only applies if the analyzer is not callable
- **min df (float or int, default=1):** When building the vocabulary, ignore terms with a strictly lower document frequency than the given threshold. If float in the range of [0.0, 1.0], the parameter represents a proportion of documents, integer absolute counts
- **stop_words:** A string is passed to check stop list, and the appropriate stop list is returned. English is currently the only supported string value. There are several known issues with English, and you should consider an alternative.

Parameters used in Word2Vec[7]:

- **size/vector_size(int,optional):** Dimensionality of word vectors
- **window (int, optional):** Maximum distance between the current and predicted word within a sentence.
- **min count (int, optional):** Ignores all words with total frequency lower than this value
- **workers (int, optional):** Uses these many worker threads to train the model (=faster training with multicore machines).
- **sg (0, 1, optional):** The first involves predicting context terms using a center phrase, while the second involves predicting the word using context words. The skip-gram model was used in our project. The SkipGram model learns to infer a term from its surroundings. Even though it takes more time for training than CBOW, it works well with a small amount of the training data and represents well even rare words or phrases.

Parameters used in Doc2Vec [8]:

- **documents (iterable of list of Tagged Document, optional):** Input corpus can be simply a list of elements, but for larger corpora, consider an iterable that streams the documents directly from disk/network
- **dm (1,0, optional):** Defines the training algorithm. If dm=1, 'distributed memory' (PV-DM) is used. Otherwise, a distributed bag of words (PV-DBOW) is employed.
- **vector size (int, optional):** Dimensionality of the feature vector
- **window (int, optional):** The maximum distance between the current and predicted word within a sentence can be chosen using this parameter
- **alpha (float, optional):** To decide the initial learning rate of the model.
- **min count (int, optional):** The model will ignore all words with total frequency lower than this
- **sample (float, optional):** This is used to set the threshold for configuring which higher-frequency words are randomly down-sampled, the useful range is (0, 1e-5).

VII. CODE

The programming language used in this project is python because python supports libraries for data processing, data conversions, and model building. The libraries used in this project are Numpy, Scklearn, NLTK, Pandas. The information about libraries, model creation, and the different parameters used in the model creation are explained in the methodology section. The project is divided into three sub-parts one for each research question. The first part is link analysis in posts and comments and the second part is time analysis for linking posts and the third part is building the model that recommends answers to unanswered questions. The code is uploaded in GitHub and the link is shared in the below reference

VIII. RESULTS

We downloaded the unanswered posts data from the stack overflow dataset, then later we have chosen ten unanswered questions with post id and title, it can be shown in below fig 15. For all ten unanswered posts in the list, we ran the models to get ten topmost related answers with their PostID's and similarity scores. Then later we recommended the post which has the topmost similarity score. We used different hyperparameters and tuned the model to get better output. The output of the three models can be shown in Figures 16, 17, and 18.

```
337 - XML Processing in Python
469 - How can I find the full path to a font from its display name on a Mac?
502 - Get a preview JPEG of a PDF on Windows?
535 - Continuous Integration System for a Python Codebase
594 - cx_Oracle: How do I iterate over a result set?
683 - Using 'in' to match an attribute of Python objects in an array
742 - Class views in Django
766 - Python and MySQL
773 - How do I use itertools.groupby()?
972 - Adding a Method to an Existing Object Instance
1171 - What is the most efficient graph data structure in Python?
```

Fig. 15. Unanswered Posts

```
Highest score for Post Id 337 is = 0.5602681983303459
Highest score for Post Id 469 is = 0.47302601673938777
Highest score for Post Id 502 is = 0.41127725075008614
Highest score for Post Id 535 is = 0.4647230425239056
Highest score for Post Id 594 is = 0.42707757837824106
Highest score for Post Id 683 is = 0.4119239741093581
Highest score for Post Id 742 is = 0.3749356990365732
Highest score for Post Id 766 is = 0.47424417298691657
Highest score for Post Id 773 is = 0.39268621842216617
Highest score for Post Id 972 is = 0.5063109556579887
Highest score for Post Id 1171 is = 0.5027743386644267
```

Fig. 16. TF-IDF Output

```
Highest score for Post Id 337 is = 0.5556448101997375
Highest score for Post Id 469 is = 0.5496886968612671
Highest score for Post Id 502 is = 0.6351563930511475
Highest score for Post Id 535 is = 0.5437466502189636
Highest score for Post Id 594 is = 0.7417551875114441
Highest score for Post Id 683 is = 0.49396514892578125
Highest score for Post Id 742 is = 0.5088351964950562
Highest score for Post Id 766 is = 0.5553408265113831
Highest score for Post Id 773 is = 0.5059433579444885
Highest score for Post Id 972 is = 0.6138319969177246
Highest score for Post Id 1171 is = 0.5723389387130737
```

Fig. 17. Word2Vec Output

```
Highest score for Post Id 337 is = 0.6675436496734619
Highest score for Post Id 469 is = 0.6557201147079468
Highest score for Post Id 502 is = 0.7186626195907593
Highest score for Post Id 535 is = 0.6886473894119263
Highest score for Post Id 594 is = 0.7548345327377319
Highest score for Post Id 683 is = 0.6817964315414429
Highest score for Post Id 742 is = 0.6681376695632935
Highest score for Post Id 766 is = 0.6960107088088989
Highest score for Post Id 773 is = 0.6520678400993347
Highest score for Post Id 972 is = 0.7824229001998901
Highest score for Post Id 1171 is = 0.6779126524925232
```

Fig. 18. Doc2Vec Output

From the following results, we can observe that Doc2Vec similarity scores are higher compared to TF-IDF and Word2Vec. The scores are ranging between 0.65 and 0.78. On the other hand, similarity scores of TF-IDF are in the range between 0.39 and 0.56, and the Word2Vec similarity scores in between 0.49 and 0.74. The TF-IDF having the least scores compared to Word2Vec and Doc2Vec because it has some disadvantages, it might be a reason for the least similarity scores. A few disadvantages of using the TF-IDF model are

- It computes document similarity directly in the word-count space, which may be slow for large vocabularies
- It assumes that the counts of different words provide independent evidence of similarity
- It makes no use of semantic similarities between words

The comparison of similarity score of all models are shown in fig-19

IX. Limitations

The criteria that we have chosen to build this model is too restrictive, as we used the posts only having python's tags for the training and testing model. By considering more posts related to different tags may increase the consistency of the results but it is known that different posts have different characteristics, which may cause some problems. In the pre-processing step of our model, we used WordNetLemmatizer for transforming words to their root form. In future work, we can use different stemmers such as Lancaster stemmer for converting words into their root form and we can compare the performances of each model. Another concern would be the impact of duplicate questions in the training dataset.

Post ID	TF-IDF	Word2Vec	Doc2Vec
337	0.5602	0.5556	0.6675
469	0.473	0.5496	0.6557
502	0.4112	0.6351	0.7186
535	0.4647	0.5437	0.6886
594	0.427	0.7417	0.7548
683	0.4119	0.4939	0.6817
742	0.3749	0.5088	0.6681
766	0.4742	0.5553	0.696
773	0.3926	0.5059	0.652
972	0.5063	0.6138	0.7824
1171	0.5027	0.5723	0.6779

Fig. 19. comparison of 3 models

X. Future Work

From our study results, we can still implement the better technique and good ideas for future research

- **Future work should include the analysis of the unanswered questions.** There are many questions in stack overflow which may be incomplete or not having accurate information about the problem. The research can be work on Why do questions at Stack Overflow remain unresolved for a long time?
- **Future work should further investigate the impact of duplicate questions in training set upon the performance of the model.** It may be possible that the question may be incomplete or not having accurate information about the problem. Why do questions at Stack Overflow remain unresolved for a long time?
- **Future work should look forward to implementing a plug-in for stack overflow website which will recommend related posts immediately once a new question has been posted.** As we can see from the results of the second research question, some posts take more time to get related links. So it would be helpful for the developers if there's a plug-in kind of thing that will recommend a set of related posts as soon as a question has been posted or while posting a question

XI. conclusion

In this research, we have analyzed the links in the stack overflow posts. We explore three different research questions. Here we used different information retrieval techniques such as TF-IDF and deep learning techniques word embedding and document embedding. In our evaluation, the results have shown that Doc2Vec performs well compared to the other two techniques. Furthermore, we find that tag-specific training corpus can improve the performance of deep learning techniques, and different training and testing sets have negligible effects on the performance. Our future work will further enhance the approach by considering more information about lexical,

semantic, user behavior, and popularity associated with the questions.

REFERENCES

[1] Yang Zhang, YiwenWu, Tao Wang, Huaimin Wang, "iLinker: a novel approach for issue knowledge acquisition in GitHub projects", World-WideWeb (2020) 23:1589–1619.

[2] Seyed Mehdi Nasehi, Jonathan Sillito , Frank Maurer, and Chris Burns, "What makes a Good example? A Study of Programming QA in StackOverflow ", 2012 28th IEEE International Conference on Software Maintenance (ICSM).

[3] Hideaki Hata, Christoph Treude, Raula Gaikovina Kula and Takashi Ishio, "9.6 Million Links in Source Code Comments:Purpose, Evolution, and Decay", 2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE).

[4] Timur Bachschia, Anik o Hann akb, Florian Lemmericha,and Johannes Wachs, "From Asking to Answering : Getting More Involved on Stack Overflow",arXiv:2010.04025v1 [cs.CY] 8 Oct 2020

[5] Huaimin Wang, Tao Wang, Gang Yin, and Cheng Yang, "Linking Issue Tracker with QA Sites forKnowledge Sharing across Communities", IEEE Transactions on Services Computing, VOL. 11, NO. 5, September/October 2018

[6] "https://scikit-learn.org/stable/modules/generated/sklearn.feature extraction.text.TfidfVectorizer.html"

[7] "https://radimrehurek.com/gensim/models/word2vec.html"

[8] "https://radimrehurek.com/gensim/models/doc2vec.html"

[9] "https://github.com/Prashanthkumar17/A-Novel-Approch-to-Link-Stack-Overflow-Posts"