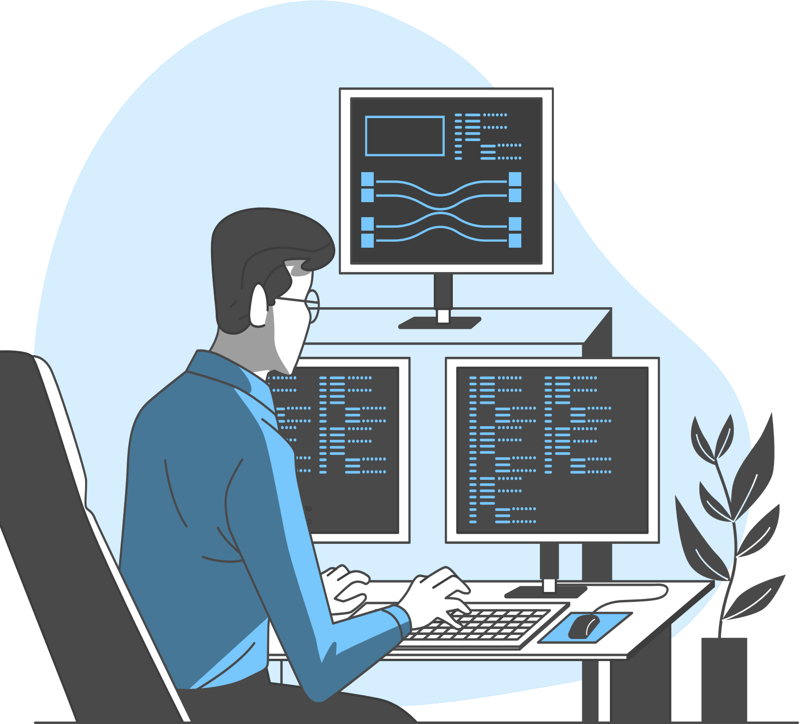
Multi-Factor Duplicate Question Detection in Stack Overflow



Team 18

Aarush Jain
Abhishek Sharma
Aryan Singhal
Vaibhav Agarwal

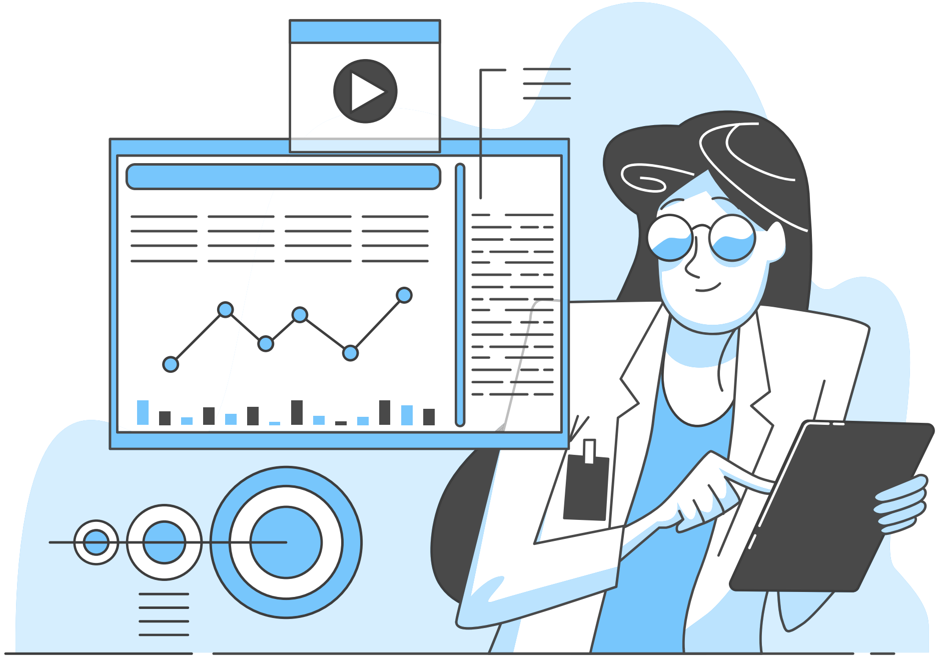
Table of Contents

01 Introduction

02 DataSet and Sources

03 Approach

04 Final Deliverable



01

Introduction

Duplicate Questions on StackOverflow

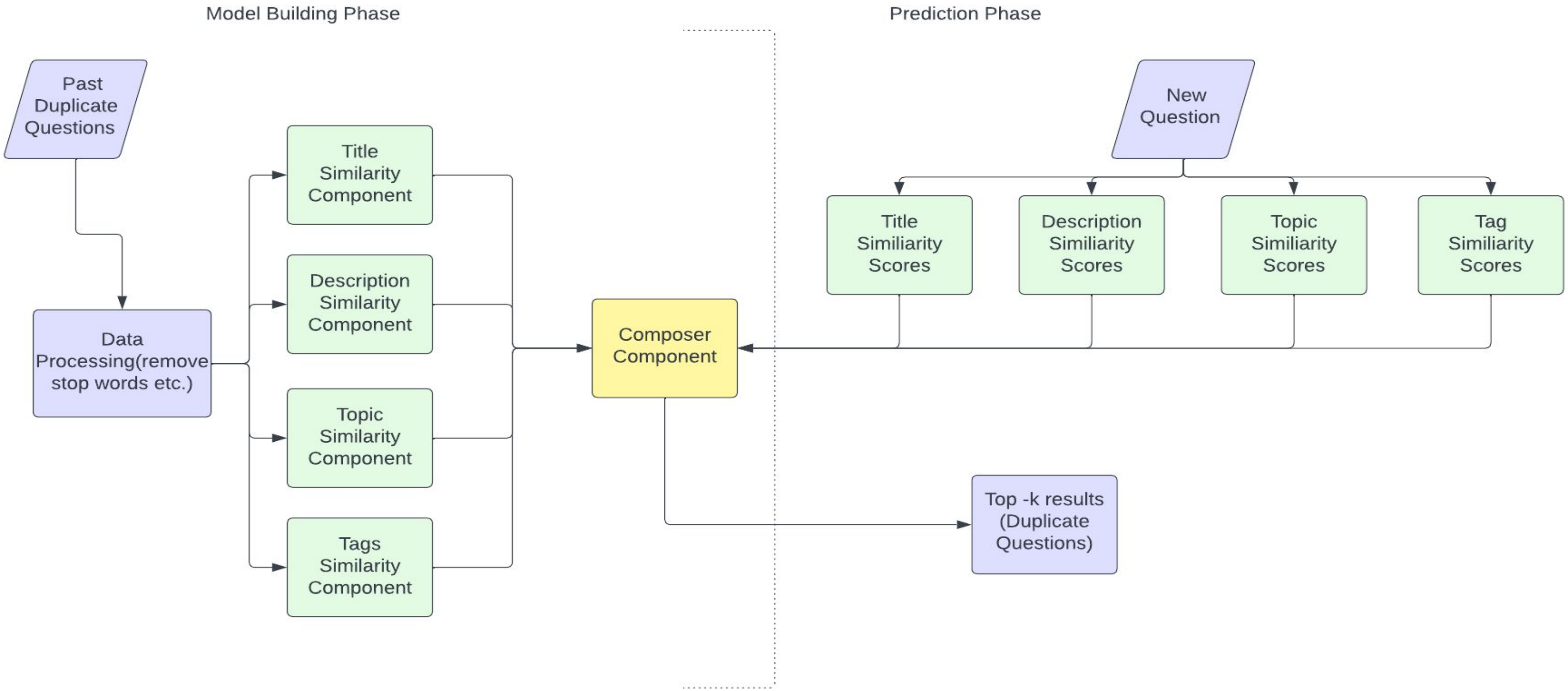
Duplicate questions make Stack Overflow site maintenance harder, waste resources that could have been used to answer other questions, and cause developers to unnecessarily wait for answers that are already available.

A typical question in Stack Overflow contains a number of fields, such as submitter, title, description, tags, and comments.

A developer needs to provide all three pieces of information when he/she submits a question to Stack Overflow. The title is a summary of the question, the description is a detailed explanation of the question, and tags are sets of words or short phrases that capture important aspects of the question

The goal is to implement **DupPredictor** which takes input a new question and gives output in form of **k** duplicate questions as the output by considering multiple factors.

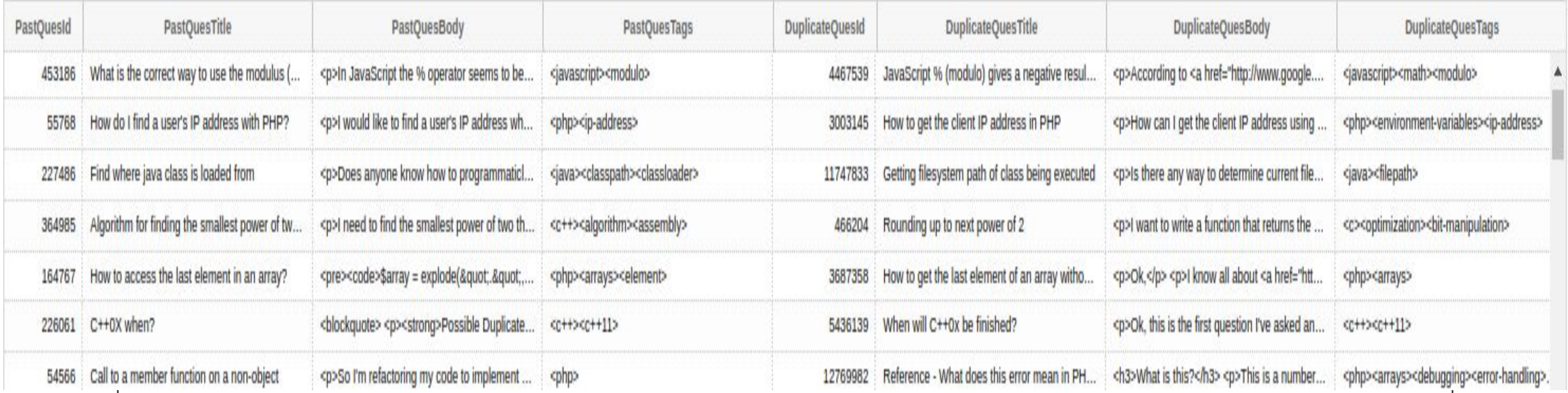
The framework of the algorithm :


Model Building Phase
Prediction Phase
Past Duplicate Questions
Data Processing(remove stop words etc.)
Title Similarity Component
Description Similarity Component
Topic Similarity Component
Tags Similarity Component
Composer Component
New Question
Title Similiarity Scores
Description Similiarity Scores
Topic Similiarity Scores
Tag Similiarity Scores
Top -k results (Duplicate Questions)

02

Dataset and Sources

- Dataset for duplicate questions on StackOverflow is collected from StackExchange.
- https://data.stackexchange.com/stackoverflow/queries
- It has around 50k entries which includes **previous question(Id,Title,Description,Tags)** with **Duplicate Question(Id,Title,Description,Tags).**



PastQuesId	PastQuesTitle	PastQuesBody	PastQuesTags	DuplicateQuesId	DuplicateQuesTitle	DuplicateQuesBody	DuplicateQuesTags
453186	What is the correct way to use the modulus (...	<p>In JavaScript the % operator seems to be...	<javascript><modulo>	4467539	JavaScript % (modulo) gives a negative resul...	<p>According to <a href="http://www.google....	<javascript><math><modulo>
55768	How do I find a user's IP address with PHP?	<p>I would like to find a user's IP address wh...	<php><ip-address>	3003145	How to get the client IP address in PHP	<p>How can I get the client IP address using ...	<php><environment-variables><ip-address>
227486	Find where java class is loaded from	<p>Does anyone know how to programmaticl...	<java><classpath><classloader>	11747833	Getting filesystem path of class being executed	<p>Is there any way to determine current file...	<java><filepath>
364985	Algorithm for finding the smallest power of tw...	<p>I need to find the smallest power of two th...	<c++><algorithm><assembly>	466204	Rounding up to next power of 2	<p>I want to write a function that returns the ...	<c><optimization><bit-manipulation>
164767	How to access the last element in an array?	<pre><code>$array = explode(&quot;.&quot;,...	<php><arrays><element>	3687358	How to get the last element of an array witho...	<p>Ok,</p> <p>I know all about <a href="htt...	<php><arrays>
226061	C++0X when?	<blockquote> <p><strong>Possible Duplicate...	<c++><c++11>	5436139	When will C++0x be finished?	<p>Ok, this is the first question I've asked an...	<c++><c++11>
54566	Call to a member function on a non-object	<p>So I'm refactoring my code to implement ...	<php>	12769982	Reference - What does this error mean in PH...	<h3>What is this?</h3> <p>This is a number...	<php><arrays><debugging><error-handling>.

03 Approach

01 Data Preprocessing

02 Similarity Scores (4 Types)

03 LDA (Latent Dirichlet Allocation) for latent topics Similarity Score

04 Final Prediction

01 Data Preprocessing

After collecting past questions , we first extract the title , description and tags from each question.

We **tokenize** the text that appears in the title , description of each question and remove common English **Stop words** (‘a’,’the’,’is’,’are’ ,’my’,’I’ and etc.), punctuations and special characters.

Perform **Stemming** (extract the base form of the words by removing affixes from them) by **Porter Algorithm**

e.g. Eating → Eat

Wrote → Write

Marks, Marked, Marking → Mark

02 Similarity Scores

After preprocessing is done , Four Scores are computed that capture the similarity of the questions.

Approach for Title , Tag and Description Similarity Component is same .

For two questions m and n , two bags represent two bags of words that extracted from their respective Component(title,tag and desc.) as Component_m and Component_n .

1. Two bags are merged and union words are eliminated which contain x words.
2. WE represent both components with vectors using Vector Space Modeling.
   Component_m = (wtm,1, . . . , wtm,v) and Component_n = (wtn,1, wtn,2, . . . , wtn,v).
   where wtq,i = relative term frequency of ith word in question q’s component.

$$wt_{q,i} = \frac{n_{q,i}}{\sum_v n_{q,v}}.$$

$n_{q,i}$

denotes the number of times the ith word of Component_u appears in the component of the question q

02 Similarity Scores

$\sum_v n_{q,v}$

denotes the total number of occurrences of all words in the component of question q, where v is the index of a word in that component_u.

Similarity using Cosine Similarity:

$$TitleSim(\boldsymbol{TitleVec}_m, \boldsymbol{TitleVec}_n) = \frac{\boldsymbol{TitleVec}_m \cdot \boldsymbol{TitleVec}_n}{|\boldsymbol{TitleVec}_m||\boldsymbol{TitleVec}_n|}.$$

Similarity Tag and Description Scores are calculated

03 LDA(Latent Dirichlet Allocation)

It is a tool for topic modelling which classifies or categorizes the text into a document and the words per topic.

It assumes that documents with similar topics will use a similar group of words. This enables the documents to map the probability distribution over latent topics and topics are probability distribution.

2 parts in LDA:

- Words that belong to a document that we already know
- Words that belong to a topic or the probability of words belonging into a topic, that we need to calculate

Algorithm

Go through each document and randomly assign each word in the document to one of k topics (k is chosen beforehand).

For each document d, go through each word w and compute :

- p(topic t | document d)
- p(word w | topic t)

Update the probability for word w belonging to topic t

03 LDA(Latent Dirichlet Allocation)

How to use LDA for calculating similarity score for the mode?

Consider a set of topic distributions T corresponding to the set of all questions. Let Td = (pd_1, pd_2, . . . , pd_t) refer to the topic distribution corresponding to question d, where pd_j denotes the probability of question d to belong to topic j.

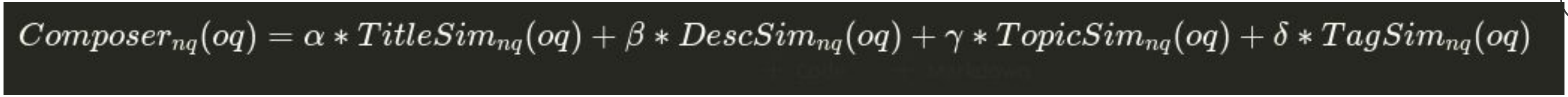
t = topic number of LDA model , determined by measuring perplexity of the mode.

Test a set of topic number and choose the one with best perplexity.

Similarity is calculated by cosine similarity as calculated for previous components.

04 Final Prediction

Composer Component :



$$Composer_{nq}(oq) = \alpha * TitleSim_{nq}(oq) + \beta * DescSim_{nq}(oq) + \gamma * TopicSim_{nq}(oq) + \delta * TagSim_{nq}(oq)$$

The composer takes all these similarity scores and calculates .

α,β,γ,δ ∈ [0,1] are calculated using greedy approach with time complexity O(m n log(n))

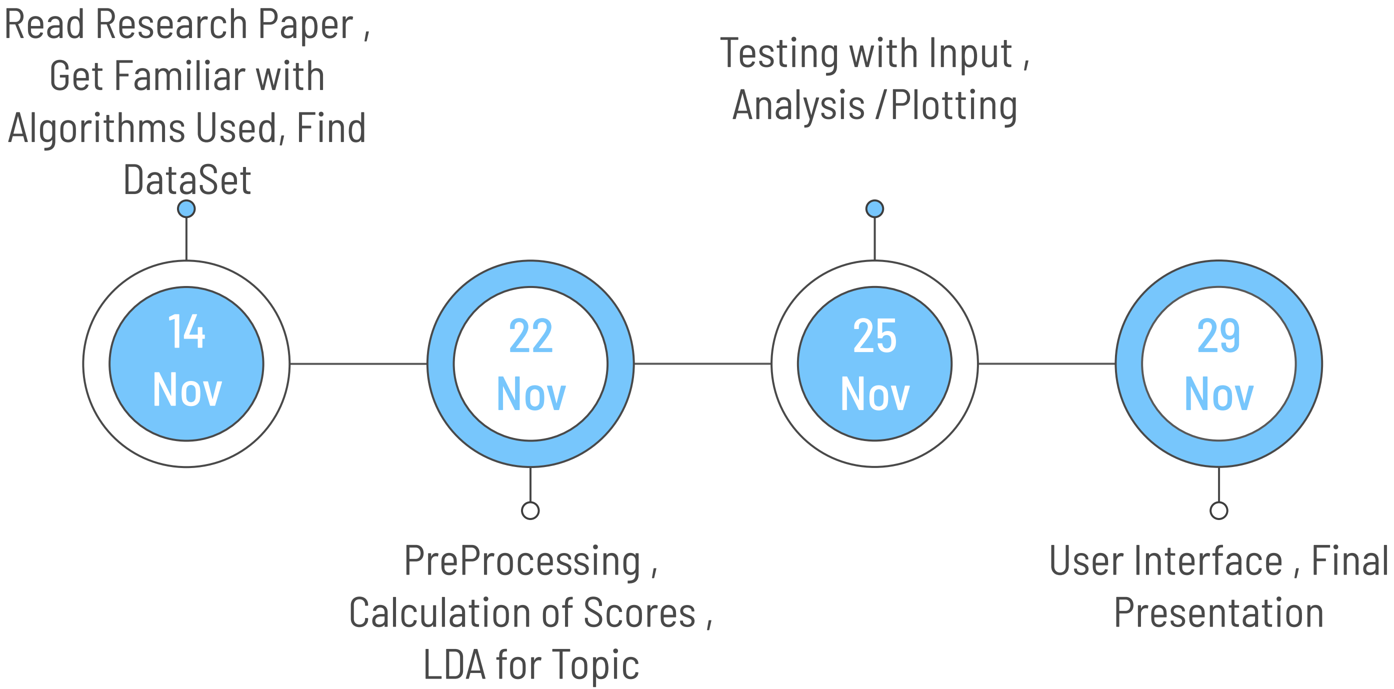
For every m duplicate questions , duplicacy is calculated using for every n past questions , then for top-K questions , we sort all the output which can be done using priority queue to reduce the time complexity.

04

Final Deliverable

- Effect of each Component in Decision Making
- Analysis of 4 parameters with different Recall Rates(Recall-rate@k measures the percentage of duplicate questions whose masters are successfully retrieved in the list.)
- User Interface where user can write the input question and choose the value of k and can see the output as top-k duplicate questions on Stackoverflow.
- Effect of varying the number of duplicate questions in the training set on the effectiveness of DupPredictor

Timeline



14 Nov

Read Research Paper , Get Familiar with Algorithms Used, Find DataSet

22 Nov

PreProcessing , Calculation of Scores , LDA for Topic

25 Nov

Testing with Input , Analysis /Plotting

29 Nov

User Interface , Final Presentation

Thanks!