



# Multi-Factor Duplicate Question Detection in Stack Overflow<sup>1</sup>

---

## Team 8 – Team **iSmart**

**Bharadwaj S** (2020122003, ECD)

**Sasanka Uppuluri** (2019102036, ECE)

**Siva Durga** (2019102038, ECE)

**Viswanadh** (2019112011, ECD)

**SMAI Course Project**  
**Course Instructor: Anoop N**  
**Mentor TA: Sireesha V**

Github Repo: [Click here](#)

1) [Zhang Y, Lo D, Xia X et al. Multi-factor duplicate question detection in Stack Overflow. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 30\(5\): 981-997 Sept. 2015. DOI 10.1007/s11390-015-1576-4](#)

# Index

---

Introduction

Dataset

Preprocessing

Similarity  
Scores

LDA

Composer

Prediction  
Phase

Training and  
Testing

Results and  
evaluations

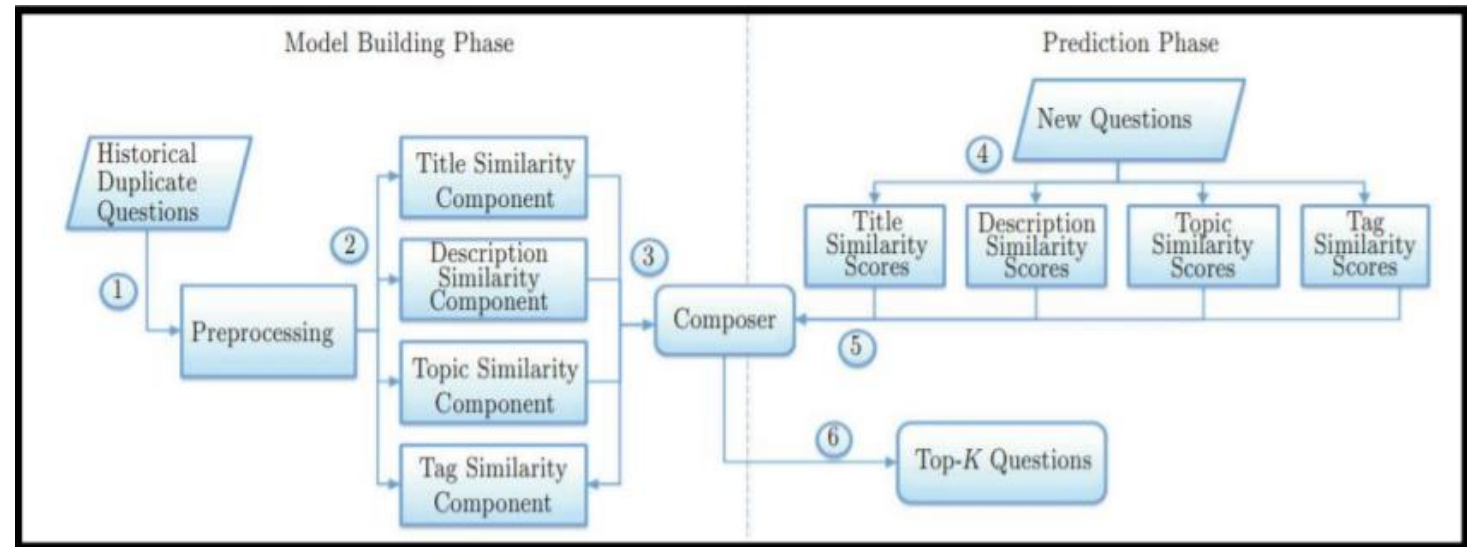
Analysis

GUI

Member  
Responsibilities

# Introduction

- The primary goal of our project is to implement DupPredictor which takes a new question as input and returns 'k' potentially duplicate/ similar questions as the output.



# Dataset

- We collected the Stack overflow question dataset from [data.stackexchange.com](https://data.stackexchange.com). It consists of 8000 duplicate and the corresponding original questions.
- Each row of the dataset primarily consists of question Id, Title, description, tags of duplicate and the corresponding original question.

RelatedPostId	OId	OTitle	OTags	OBody	DId	DTitle	DTags	DBody
0	4	4	How to convert a Decimal to a Double in C#?	<c#><floating-point>type-conversion<>double<>...	51027658	How to echo a JS variable to php?	<javascript><php>	<p>Is that possible to pass a JS variable to P...

Old represents original question Id

DId represents duplicate question Id



# Preprocessing

- We then **tokenize** the text that appears in the title and description of each question, remove common English **stop words**, and perform **'stemming'**.
- Stemming stops the commonly used articles, conjunctions and so on and reduces a word to its root form

Playing → Play  
Plays → Play  
Played → Play

} Common root form 'play'

am, are, is → be

Car cars, car's, cars' → car

Using above mapping a sentence could be normalized as follows:

the boy's cars are different colors → the boy car be differ color

# Similarity scores

- We have 4 components:
  - **Title**
  - **Tag**
  - **Description**
  - **Topic.**
- In the first three initially do **vector space modeling** (VSM), we represent the two titles as two vectors:
- $Vec_m = (wtm,1, wtm,2, \dots wtm,v)$
- $Vec_n = (wtn,1, wtn,2, \dots wtn,v).$

$$\begin{aligned} & TitleSim(TitleVec_m, TitleVec_n) \\ = & \frac{TitleVec_m \cdot TitleVec_n}{|TitleVec_m| |TitleVec_n|}. \end{aligned}$$

# Latent Dirichlet Allocation (LDA)

- LDA is a topic modelling which takes the title and description and returns the probability distribution of the topics  $k$ .
- We first perform LDA and take the probability values to be the vectors and then calculate the similarity scores.

*“Dogs like to chew on bones and fetch sticks”.*

*“Puppies drink milk.”*

*“Both like to bark.”*

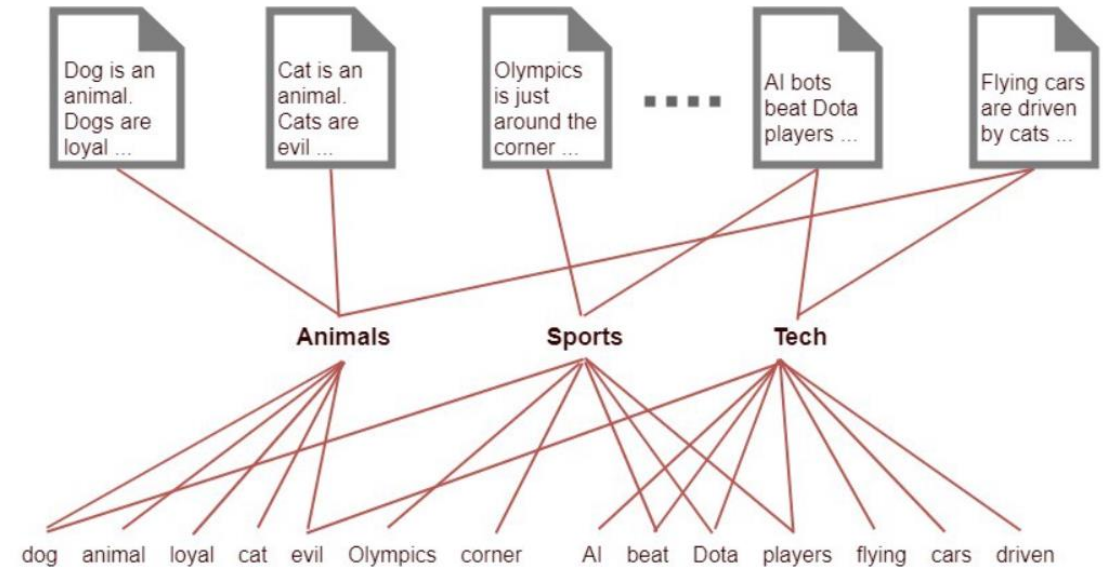


Fig: Deriving topics from documents. (Taken from towardsdatascience)

# Composer

- The composer takes all these similarity scores and calculates

$$\text{CompScore} = \alpha \times \text{TitleScore} + \beta \times \text{DesScore} + \gamma \times \text{TopicScore} + \delta \times \text{TagScore}$$

- $\alpha, \beta, \gamma, \delta \in [0, 1]$  are calculated using a greedy algorithm approach of time complexity  $O(m \times n \log n)$ .



# Prediction Phase

- Now that we have the values of the coefficients, we can now enter the prediction phase where we calculate the scores with respect to that new question and return the top k answers.

## How to convert char to integer in C?



Can any body tell me how to convert a char to int?

```
char c[]={'1',':','3'};

int i=int(c[0]);

printf("%d",i);
```

When I try this it gives 49.

[c](#) [char](#) [int](#) [conversion](#)

[edit](#) [close](#) [flag](#)

**This question is a duplicate of which other question?**

[convert char](#)

[625010 - converting char\\*\\* to char\\* or char](#)

[606075 - How to convert char \\* to BSTR?](#)

[620446 - Convert char for bit data to integer in DB2](#)

[806724 - How to convert char to integer in C?](#)

[78125 - Why can't I convert 'char\\*\\*' to a 'const char\\* const\\*' in C?](#)

[159442 - What is the simplest way to convert char\[\] to/from tchar\[\] i...](#)

[162303 - Convert Char\[\] to a list<byte> \(c#\)](#)

[347949 - Convert std::string to const char\\* or char\\*](#)

[420394 - Detecting spanish accents in chars and converting them to no...](#)

[690675 - How to convert 'const char \\*' to 'const char \\*\\*' ?](#)

# Training & Testing

- To **Train** the *dupPredictor* model, we assumed our original questions as the questions dataset, and out of all the duplicate questions, we used the first 300 questions as train data for training the composer model parameters, (Title, Tag, Body, and Topic similarity scores).
- For **testing** the *dupPredictor* model, we take the first 1200 duplicate questions of the remaining dataset as test data and tested the composer model obtained in the training model.

# Results & Evaluation metrics

- Evaluation Metric – **Recall rate@k**: denotes the ratio of number of original questions in the top – k similar questions to the total number of questions.

$$\text{recall-rate@}k = \frac{N_{\text{detected}}}{N_{\text{total}}}.$$

- For computing composer scores, we used pseudo-greedy algorithms, and the optimal composer scores obtained from it are as follows
  - $\alpha = 0.682, \beta = 0.64, \delta = 0.15, \gamma = 0.06$
- After tweaking the scores obtained from the pseudo random algorithm to maximise the accuracy, we get
  - $\alpha = 0.8, \beta = 0.51, \delta = 0.37, \gamma = 0.04$
- The recall-rate obtained with the above composer scores in the testing phase is **Recall-rate@20 = 0.725** [in the testing phase].

## Analysis

- Contribution of individual components in the dupPredictor model

Type	Recall-Rate@5	Recall-Rate@10	Recall-Rate@20
Title only	0.30666	0.375	0.445833
Description	0.2525	0.306666	0.37666
Tag only	0.261666	0.331666	0.398333
Combined	0.455	0.585	0.725

# Additional data sets

- To test the model, we take a new dataset which contains questions related to physics, taken from physics.stackexchange.
- Unlike, the stackoverflow dataset, which contains both coding blocks and english words, this dataset contains mostly english words.
- We notice that the  $\alpha > \beta > \delta > \gamma$  does not hold in this case.
- Instead, we have  $\beta$  to be the **highest value** meaning that it gives more weightage to the description component.

# Analysis

- For the dataset with physics questions, we noticed the following

	$\alpha$	$\beta$	$\delta$	$\gamma$	Recall-Rate@20
Stack overflow	0.8	0.51	0.37	0.04	0.725
Physics questions	0.4804	0.9271	0.52995	0.00944	0.77

- We can notice that the recall rate for stack overflow is slightly less because it contains mostly coding blocks and not english words, so information is lost in description due to preprocessing the coding block and thus resulting in lesser weightage for description component.





# GUI

- A simple python tkinter<sup>1</sup> based GUI was developed where a question is taken as input through multiple fields such as "Title", "Body" and "Tags".
- It outputs top K similar questions to the given question depending on their similarity scores.

1) <https://docs.python.org/3/library/tkinter.html>



---

Input Format:



"Enter the title" - text



"Enter the body" - text or HTML body



"Enter the tags" - tags separated by ',' or ' '  
(whitespace) or '<>'



"Enter k-value" - A integer value

Similar Question detector

Enter the Title:

Enter the Body:

Enter the Tags:

Enter k-value:

----- Question 0-----

Score---> 0.4652

Title ---> What Linux shell should I use?

Body ---> <p>I've used bash, csh, and tcsh. But I asked <a href="http://web.archive.org/web/20151121164806/http://stackoverflow.com/questions/198723/whats-in-your-cshrc" rel="nofollow noreferrer">this question</a>, and Jonathan informed me that csh isn't to be trusted. So what Linux shell is good for development. and why?</p>

Tags ---> <linux><shell>

-----

----- Question 1-----

Score---> 0.3972

Title ---> Receive and send emails in python

Body ---> <p>How can I receive and send email in python? A 'mail server' of sorts.</p>

<p>I am looking into making an app that listens to see if it receives an email addressed to foo@bar.domain.com, and sends an email to the sender.</p>

<p>Now, am I able to do this all in python, would it be best to use 3rd party libraries? </p>

Tags ---> <python><email>

-----

----- Question 2-----

Score---> 0.3903

Title ---> Standalone Python applications in Linux

Body ---> <p>How can I distribute a standalone Python application in Linux?</p>

<p>I think I can take for granted the presence of a recent Python interpreter in any modern distribution. The problem is dealing with those libraries that do not belong to the standard library, i.e. wxPython, scipy, python cryptographic toolkit, reportlab, and so on.</p>

<p>Is there a working Linux counterpart to, say, py2exe (which, by the way, I have never tried)? Is there a free, opensource one?</p>

Tags ---> <python><linux>

-----

Input Parameters for a question

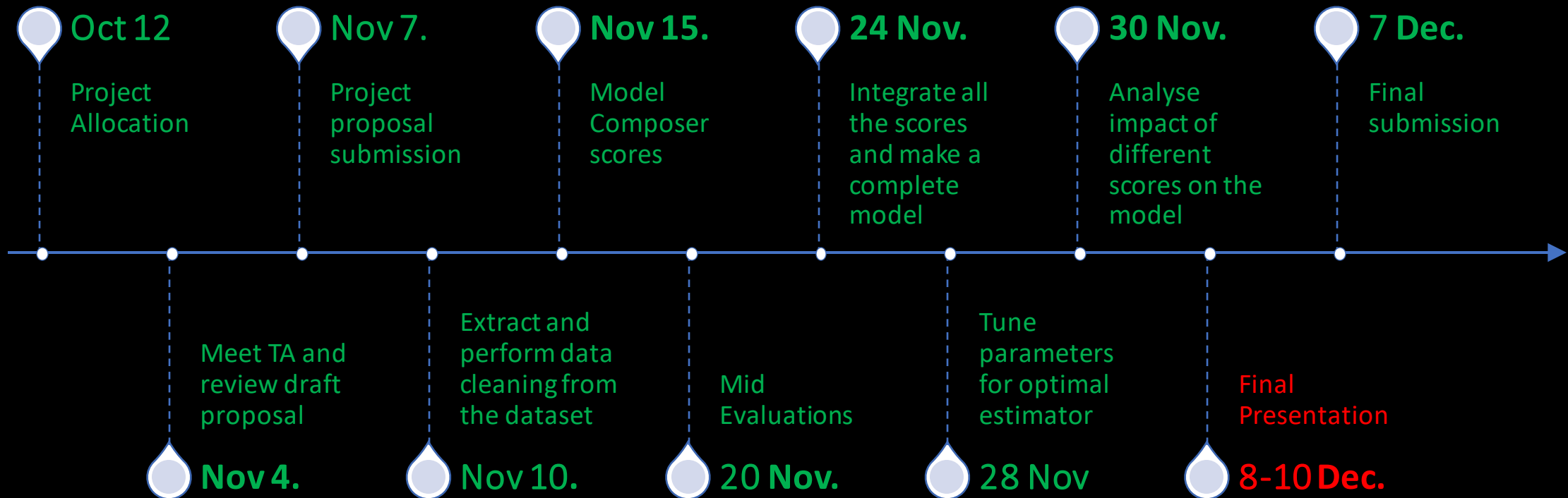
Test the model by clicking Submit

Detected Similar questions

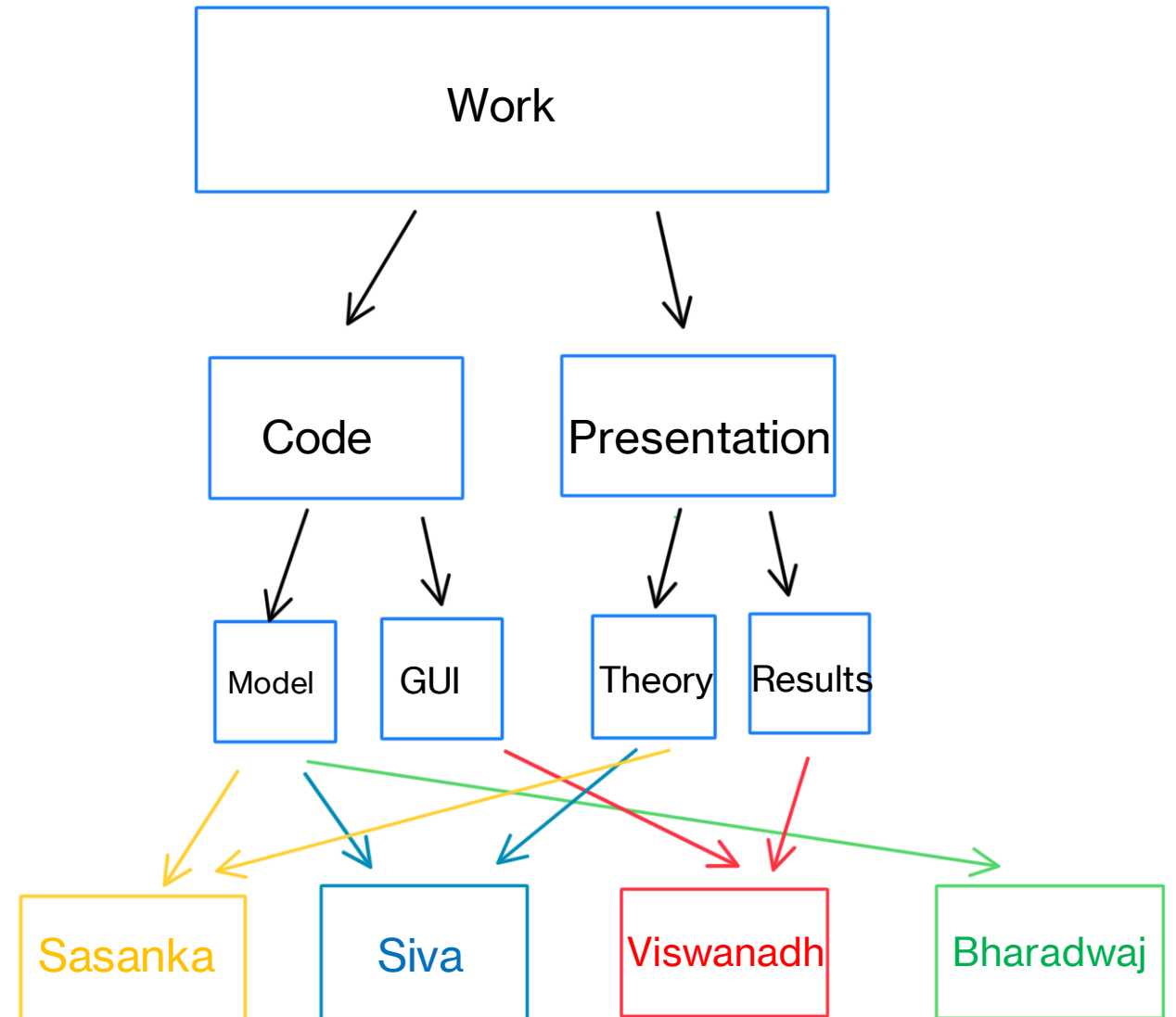
Fig: Preview of GUI

# Timeline

Work Completed  
Yet to be done



# Member responsibilities



# **Thank You!**

