Final Report

Paper Name: Multi-Factor Duplicate Question Detection in Stack Overflow
TEAM NUMBER: 24
Project ID: 26
Team Name: Hyperplane Scavengers
Team members: Gurkirat Singh, Anmol Agarwal, Shrey Gupta, Pratyush Pratap Priyadarshi
Mentor Name: Sireesha

Important Links:

- CodeBase: Link
- All files used for training and testing model: Link
- All precomputed scores: Link
- Bert embedding scores: Link
- All computed parameters: Link

Dataset Curation

- We first downloaded the large MSR dataset but it didn't have all the files so we used the official StackOverflow dump of 2021 and filtered it to make sure we considered data only till 2012.
- **Unusual result obtained while replicating the claims made in the paper**: We used the same approach as proposed by the authors of finding duplicates by looking for a [Duplicate] label at the end of the title but we found only 2 duplicate posts out of a total of 2 Million posts. We even tried **regex searching** for a variety of variations in a string like [DUplicate],[duplicate] but that got the same result.
- Finally on the basis of online literature and some StackOverflow posts we were able to find the stack exchange data dump which contained the StackOverflow data until 2021 (link to data dumps: Link). We found a file **PostLinks.xml** containing all the duplicate relationships between questions.

Database Creation

- Initially, we were using JSON to save and load data, but we soon realized that it was not the best way as it was becoming a bottleneck because of **memory issues, sorting, and random access**.
- We used a **Postgres Database** to save our data and use it for further processing.
- Later we also pushed our score to it to use for visualizations and error analysis.

Preprocessing

Cleaning Title, Body and Tags attributes

Title

- To maintain uniformity across all the titles we converted everything to lowercase.
- We removed the [Duplicate] word written anywhere in the Title.

Body

- The body contained any extra information like
- We removed all HTML tags and data inside the <code> tag as it does not add any value.

Removal of Stopwords, Stemming, Tokenization

After doing preliminary cleaning of title and body attributes to reduce noise in the text, we do additional preprocessing steps to get a cleaner and less noisy text:

1. We replace all multi spaces, tabs, and newlines with a single white space
2. The text had content like "<p>C++ is an OOP language</p>", where the "<p>" HTML tag is not an actual part of the question. As a result, we remove **all the HTML tags** while preserving the content between them. But for the "<code>" tag, we remove the content within the tags as well.
3. All the punctuation except "#" is removed to compensate for unnecessary full stops, code blocks, and different grammatical practices. "#" is left as coding languages such as **C#** are considered unique and should not be confused with the language C. As many computer science doubts are asked on the StackOverflow platform for many languages, paying special attention to these languages becomes crucial. After this, the text is tokenized by splitting on all whitespace to get individual tokens
4. We then stem all the individual tokens to get their root form and reduce the size of the vocab dictionary
5. We also remove common English stopwords. For this, we create a *stopwords.py* file that removes words that are frequent in all the English language and thus would prevent unnecessary similarity and give more focus on the specific words.
6. These tokenized texts are stored in JSON files to prevent recomputation as they run extremely slow.

DupPredictor Model

Bag of words model and Cosine Similarity

On the basis of the tokenized text, we created a bag of word embeddings for components involved in the pipeline. On the basis of this bag of words representation, the cosine similarity is used to measure the closeness b/w the two vectors which are used in the estimation of the parameters of the greedy composer.

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

LDA Model

We used the LDA topic modelling to get the probability distribution of each post. We concatenate the title and body of each post to create the final text string as many specific keywords are present in the title and the words present in the title of one post may be present in the body of another post. This allows us to get the general trend of the conversation of the post.

- We preprocessed the combined text string to get tokens of each text string. And this information is stored in the database to prevent repetitive preprocessing of the same text in later steps. Error handling was taken care of and the created dictionary was saved in case of any error as it took a lot of time for this to run.
- On the basis of this dictionary, we create the bag of word representations of each preprocessed representation. Using these Bag Of Word representations, we create the LDA Model with K = 100 topics using the Gensim LDAMulticore Library which allows multiprocessing of the LDAModel creation and thus saving time.
- Finally, on the basis of the LDA model, we created the topic distribution of each post and stored it into JSON (and later into the database) to be used for cosine similarity in the composer.

Greedy Composer

- All the parameters are then fine-tuned using a greedy approach. For each iteration, one parameter is chosen and increased in step sizes of 0.05. On the basis of each parameter set created we calculate the weighted composer sum for each duplicate question with all the questions which came before it and store the top K questions which are done using a max heap

$$\begin{aligned} Composer_{nq}(oq) = {} & \alpha \times TitleSim_{nq}(oq) + \\ & \beta \times DesSim_{nq}(oq) + \\ & \gamma \times TopicSim_{nq}(oq) + \\ & \delta \times TagSim_{nq}(oq) \,. \end{aligned}$$

- On the basis of these top K *"plausible"* questions which the duplicate question can be the duplicate of, we calculate the evaluation metric.
- On the basis of the obtained score from the evaluation criteria, we get to take the best set of params which are finally reported as the best params according to our training dataset
- After calculating all the individual components, a final score was calculated by taking a weighted sum of all the components using 4 parameters (alpha, beta, gamma and delta). For each question, we calculate the final similarity with all the posts that came before it and store it for fine-tuning the parameters later.

Results

Paper Results

The authors of the paper were interested in answering a few questions about the performance of the DupPredictor Model. We try to replicate these findings with our own dataset and the DupPredictor Model which we have trained.

Recall-rate@k

To evaluate the performance of DupPredictor the authors used the following evaluation metric:

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

Where $N_{detected}$ is the number of duplicate questions whose masters (i.eOriginal questions that are posted earlier in Stack Overflow) appear in the list of top-K questions (by DupPredictor), while N_{total} is the total number of duplicate questions used for testing.

Research Question 1: How good is DupPredictor over its 4 individual components?

Why did we do it?

To prove the novelty of the method, we needed to measure the performance of the composite DupPredictor model over its 4 individual components.

How did we do it?

To measure this, we compute recall-rate@10 and recall-rate@20 of DupPredictor when it is applied to the StackOverflow question dataset. We compare these recall rates, with the recall rates achieved using solely the title, body, topic and tag similarity components.

What result did we get?

From the table below we can observe that DupPredictor outperforms the 4 components in terms of recall-rate@10 by 69.69%,180%, 600% and 100% respectively. Similarly for recall-rate@20, we can see that the combined model outperforms the 4 components by 54.76%, 150%, 364.28% and 91.17% respectively.

DupPredictor achieves great improvements over its individual components and thus we can observe the novelty in combining multiple sources of information for duplicate detection.

Algorithm	Recall Rate@10	Improvement (%)
DupPredictor	0.56	0.0
Title similarity	0.33	69.69
Description similarity	0.2	180
Topic similarity	0.08	600
Tag similarity	0.28	100

Algorithm	Recall Rate@20	Improvement (%)
DupPredictor	0.65	0.0
Title similarity	0.42	54.76
Description similarity	0.26	150
Topic similarity	0.14	364.28
Tag similarity	0.34	91.17

Research Question 2: Effect of Varying Number of Training Questions

Why did we do it?

We needed to check whether changing the number of questions in training has drastic effects on the prediction performance.

How did we do it?

To measure this, we trained the model over 2 training set sizes, the first 100 and 300 questions. The test set consisted of the questions whose rank lie between 300 and 400 when all the duplicate questions are sorted according to their creation date. They were tested on the same testing set for uniformity. We measured the recall-rate@10 and recall-rate@20 to check our hypothesis.

What result did we get?

We observe that the composer learns different parameters for both training sets and sometimes their relationship (inequality) also changes, but overall the trend remains the same. Title and body similarity is given more preference followed by tags similarity and lastly topic similarity.

Trained on first	Alpha (Title)	Beta (Body)	Gamma (Topic)	Delta (Tags)
K = 10				
100	0.5	0.5	0.0454	0.45
300	1.0	0.6	0.3	0.6
K = 20				
100	0.4	0.5	0.13738	0.1992
300	0.9	0.85	0.3153	0.3

On the basis of the results, we can also observe that the model trained on the first 300 questions is able to generalize more on unseen data but the performance difference is extremely negligible. This means that the model is able to learn and tune its parameters even on a lesser amount of data.

Trained on first	Recall@10 Training	Recall@10 Testing	Recall@20 Training	Recall@20 Testing
100	0.74	0.55	0.78	0.62
300	0.67	0.56	0.74	0.65

Research Question 3: Does DupPredictor estimate the 4 weights of its 4 constituent components well?

Why did we do it?

DupPredictor estimated the 4 weights using a greedy random restart approach tuning one parameter at a time instead of using an optimization strategy to minimize loss as in traditional ML models. We wanted to know if these estimated parameters are accurate and well predicted.

How did we do it?

In order to analyze the performance, we randomly generate 50 sets of weights of alpha, beta, gamma and delta to compute the weighted score in the composer. The recall-rate@10 and recall-rate@20 are measured on the 100 questions test set for these 50 sets of weights and the results are compared to the best parameters generated by DupPredictor on training with the first 300 duplicate questions.

What result did we get?

We observe that the parameters generated by DupPredictos using its simplistic approach outperform all the randomly generated weights on both metrics.

alpha	beta	gamma	delta	recall-rate@10	recall-rate@20
0.9	***0.85***	***0.3152***	***0.3***	***0.56***	***0.64***
0.89	0.0687	0.29	0.2593	0.53	0.62
0.8118	0.5148	0.6062	0.3224	0.53	0.61
0.2605	0.88	0.6911	0.08	0.36	0.45
0.1749	0.4582	0.7276	0.0998	0.36	0.45

For the full results table, you can refer to this link: Results CSV

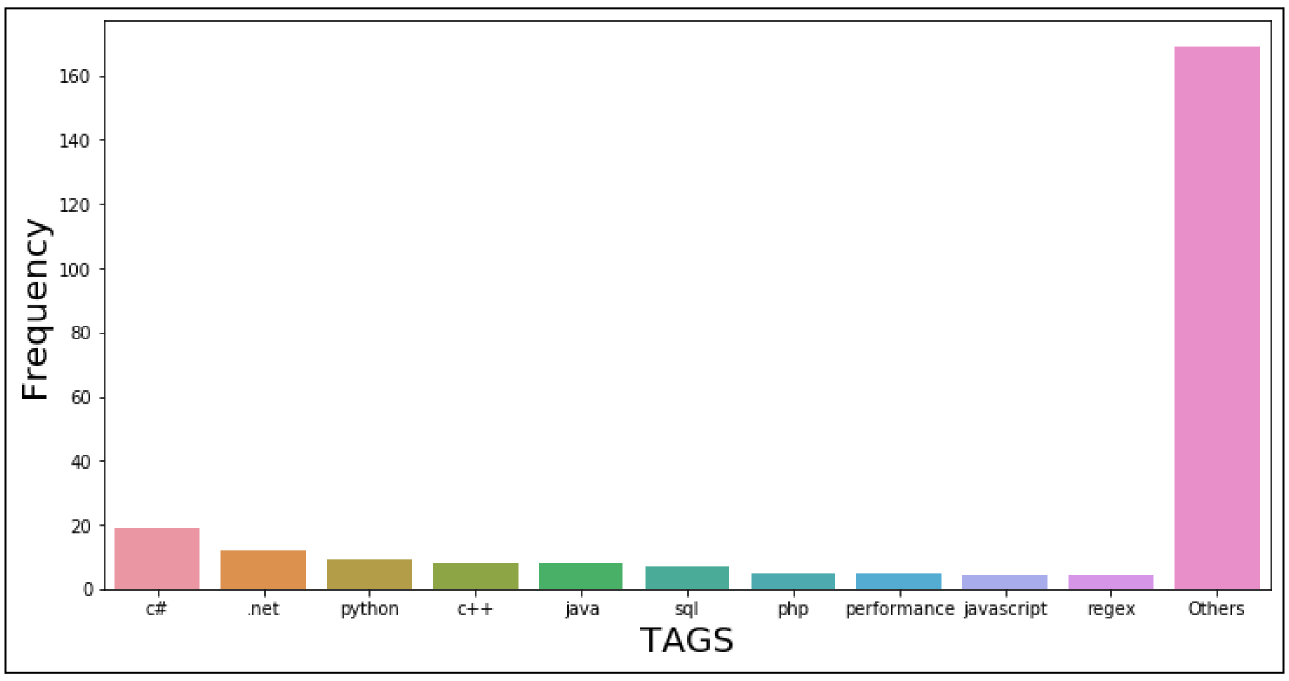
On the basis of these results, we can infer the following properties:

- Some randomly generated parameters perform very similarly to the parameters estimated by the DupPredictor model however NONE of the models outperforms DupPredictor. For example, the weights of the 2nd best model are very close to the weights predicted by DupPredictor, but it gives very little weight to the body of a question and hence it underperforms DupPredictor.
- Generally speaking, models with weights following the general relationship alpha ≥ beta ≥ delta ≥ gamma and alpha close to 1 perform the best which supports our findings as well.
- The worst performing models give too little attention to the title or too much attention to the topic causing them to miss out on important information or focus on misleading information heavily respectively.

ERROR ANALYSIS AND BASIC STATISTICS

TOTAL NUMBER OF QUESTIONS CONSIDERED = 83,350

What tags were present the most in duplicate questions?

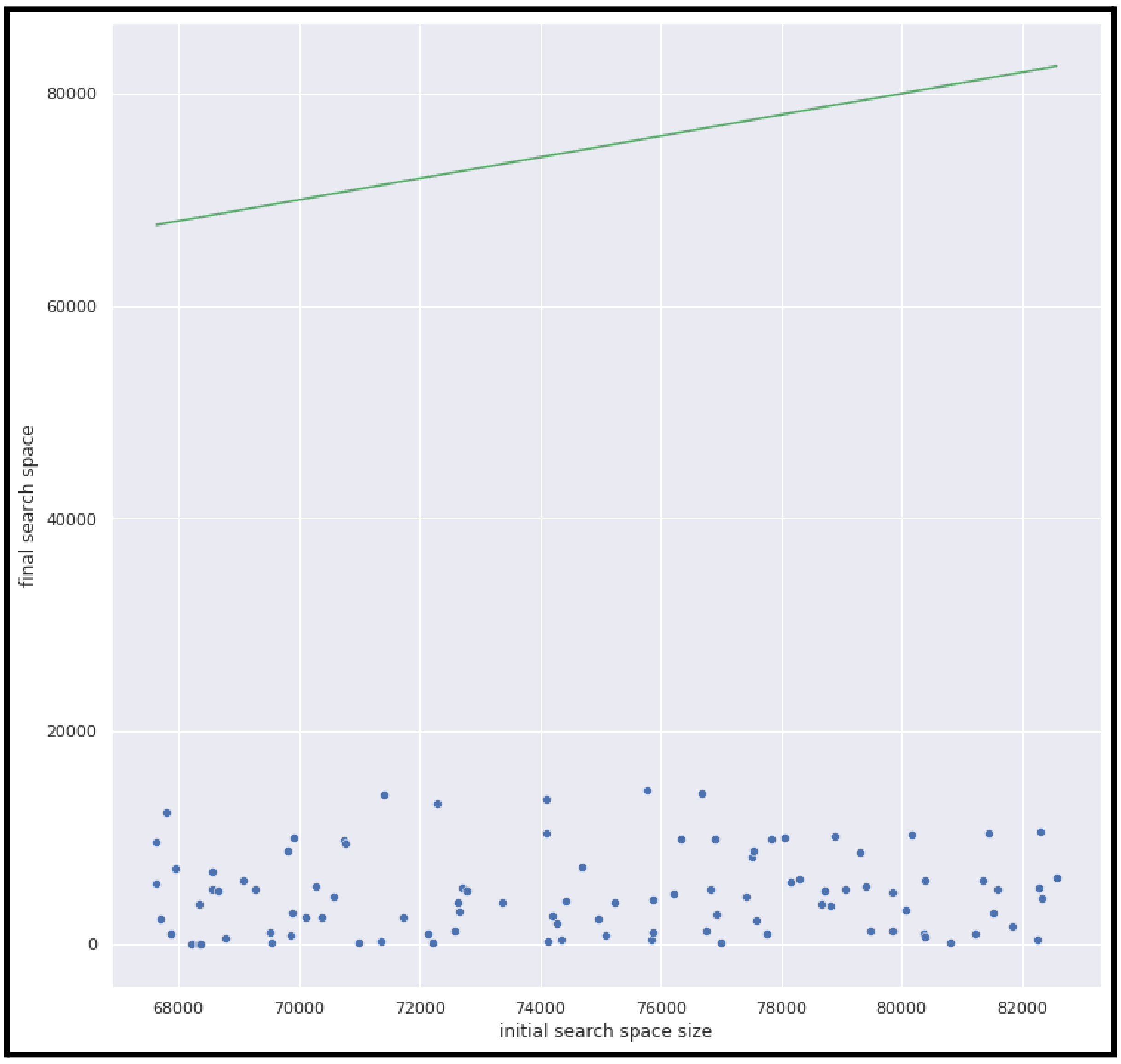


How many times did the duplicate questions have a common tag with their parents?

- Out of **100** elements in the test set, **94** of the questions had at least one tag common with their duplicates.
- The 6 cases where there was no common tag between the original question and its duplicates.

441910	476993	495244
450924	495018	513315

- This clearly indicates that **reducing the search space** based on whether the candidate duplicate has any common tags with the question Q would be highly beneficial to reduce the time needed to find a duplicate **without compromising on performance**. In the figure below, x: initial search space & y: search space size after pruning. As one will be able to see in the result, the performance increases on a pruned search space, thus **benefiting in both time and accuracy**.

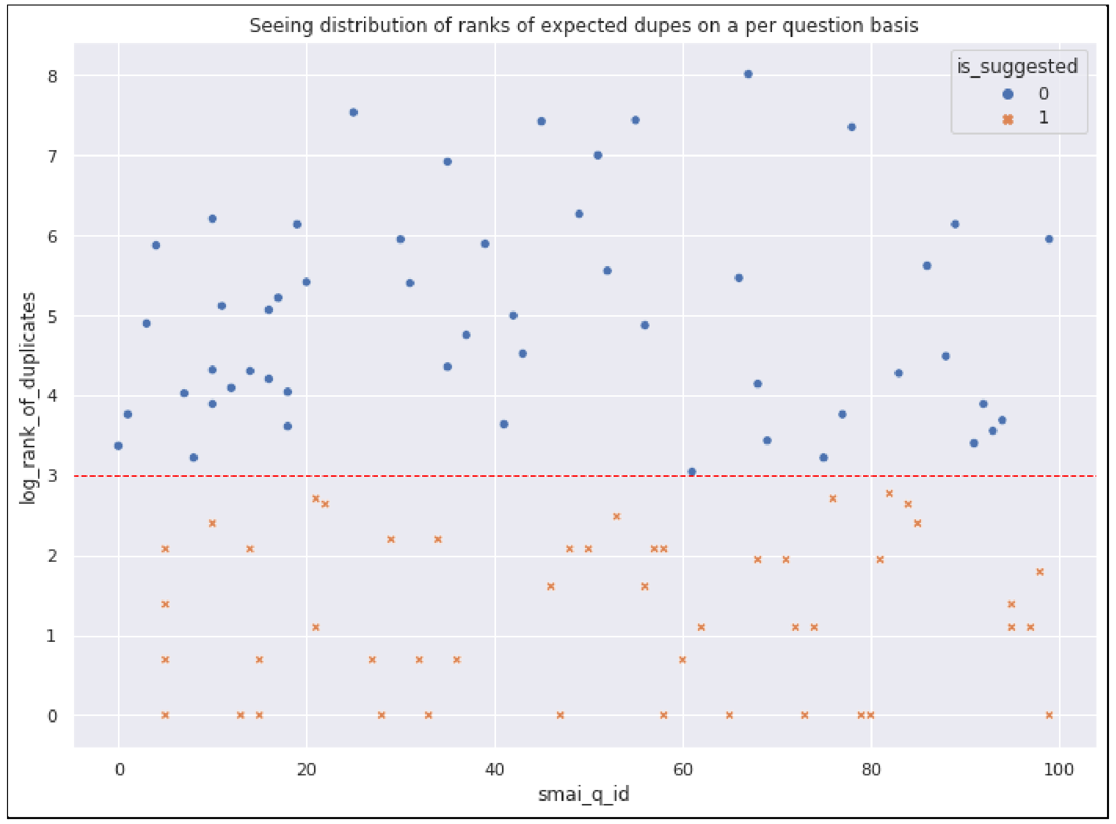


SUMMARY OF ABLATION STUDY SCORES (at Recall@20)

Model description	Training Recall@20	Test Recall@20		Mean rank of highest ranked duplicate		Best rank of a duplicate		Worst rank of a duplicate		Parameters
		Without Pruning	With pruning	Without Pruning	With pruning	Without Pruning	With pruning	Without Pruning	With pruning	
K=20 with all components	0.74	0.65	**0.68**	175.84	99.9	1	1	2365	1359	[0.9, 0.85, 0.31, 0.3, 0.0]
K =20 with all components except Title Similarity	0.61	0.53	0.53	454	287	1	1	12366	4048	[0, 0.85, 0.35, 0.64, 0.0]
K =20 with all components except Topic Similarity	0.733	0.61	0.61	229	160	1	1	3691	3613	[0.2, 0.19, 0, 0.07, 0.0]
K =20 with all components except Body Similarity	0.707	0.61	0.62	214	120	1	1	4058	1972	[1.0, 0, 0.45, 0.6, 0.0]
K =20 with all components except Tags Similarity	0.67	0.51	0.62	540	122	1	1	25818	3177	[0.9, 0.8, 0.1, 0, 0.0]
K=20 with all components + Jaccard Coeff included	**0.763**	0.65	**0.67**	**105**	**78**	1	1	2054	1197	[0.45, 0.35, 0.25, 0.29, 1.0]

Some insights based on the table above and the detailed analysis in the next section

- Pruning the search space is able to improve results and is quicker computationally
- The most significant increase in results after pruning was in the case where tag similarity was discarded. This was as pruning based on tags does a job similar to tag similarity.
- As per the ablation study, the most significant parts for detection seem to be "title" and the "tags".
- With moderators strictly enforcing titles to be precise and accurate, "title similarity" seems to be a great indicator.
- TAGS become important as it helps in restricting search to relevant languages and technologies. Eg: a question regarding retrieval queries on MySQL is likely to be similar in body and title when compared to the same question wrt MongoDB. However, the two questions being different tagged is what is allowing the model to successfully differentiate between the two.
- In the adjoining figure, x: question id ; y : rank of one of the expected duplicates. Since a question can have more than one expected duplicate, there may be multiple dots on the same vertical line. The red line depicts the threshold (20) beyond which questions are not suggested. Clearly, for some questions, one of the duplicates

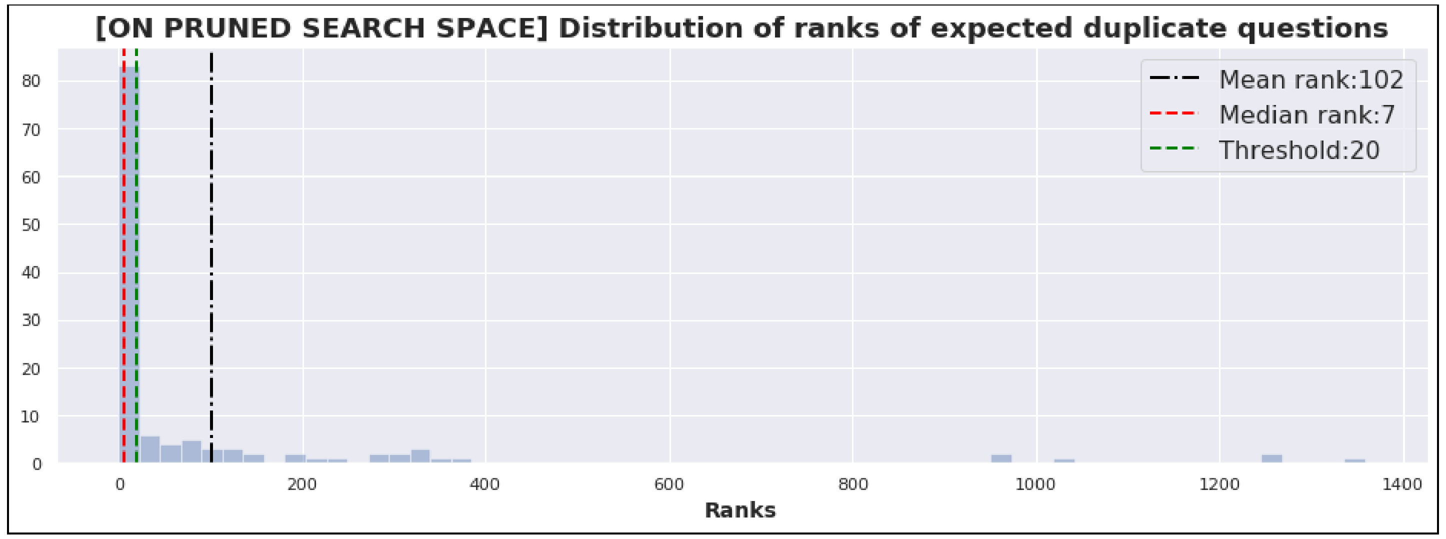


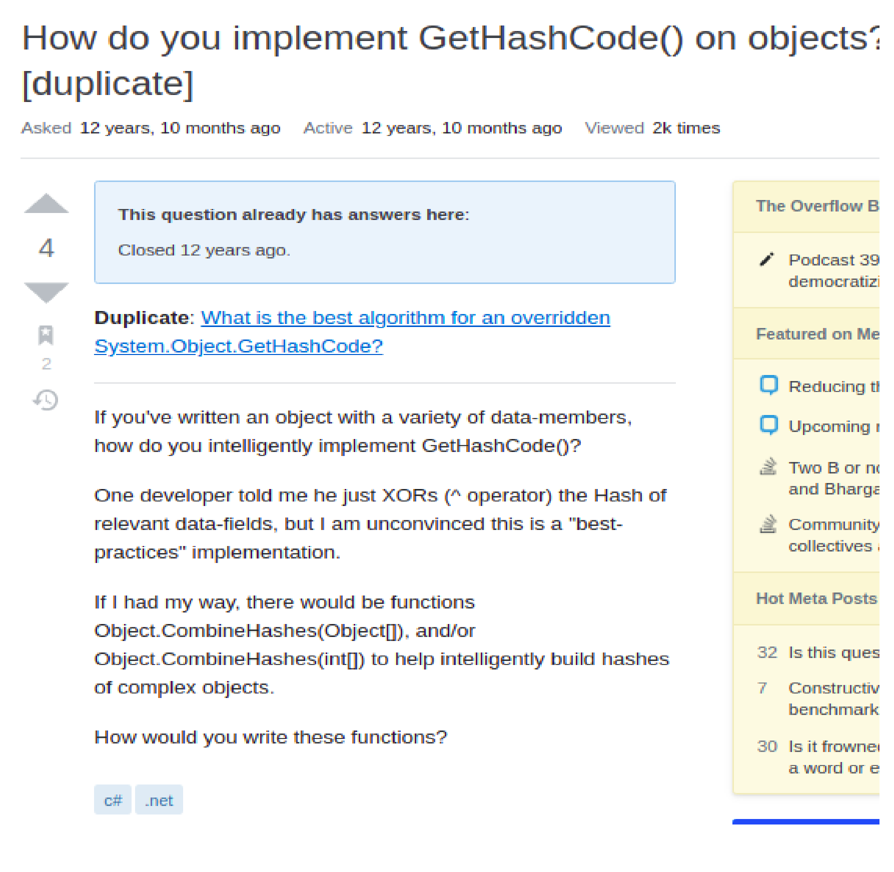
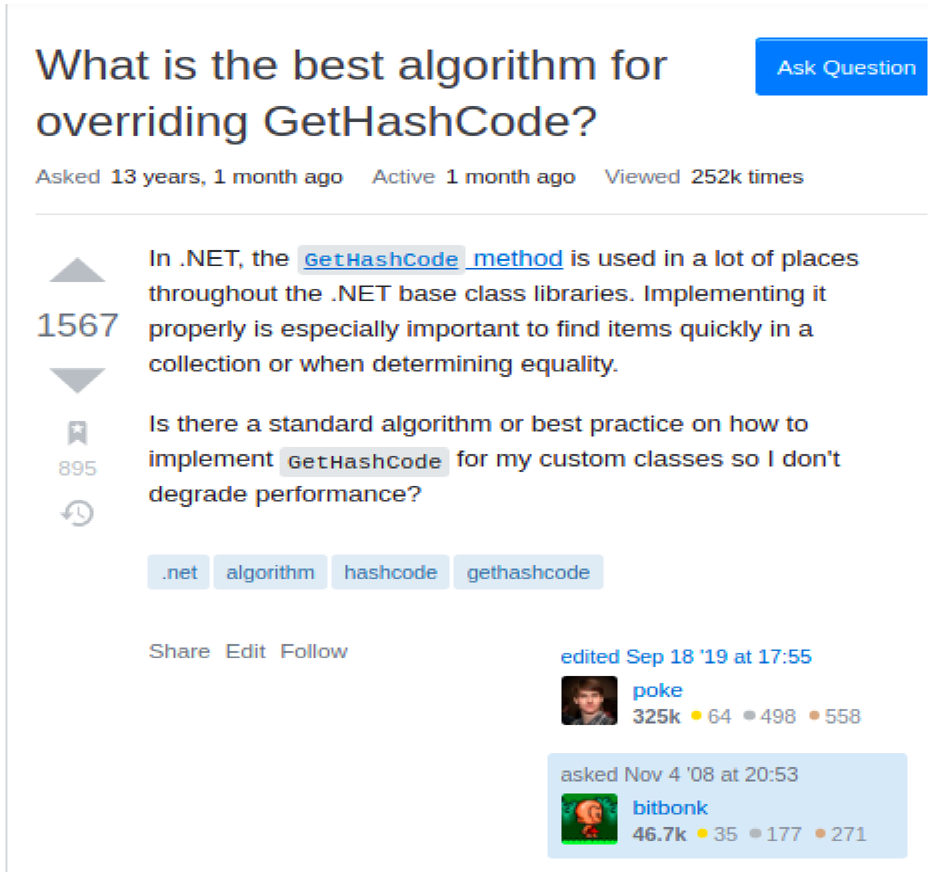
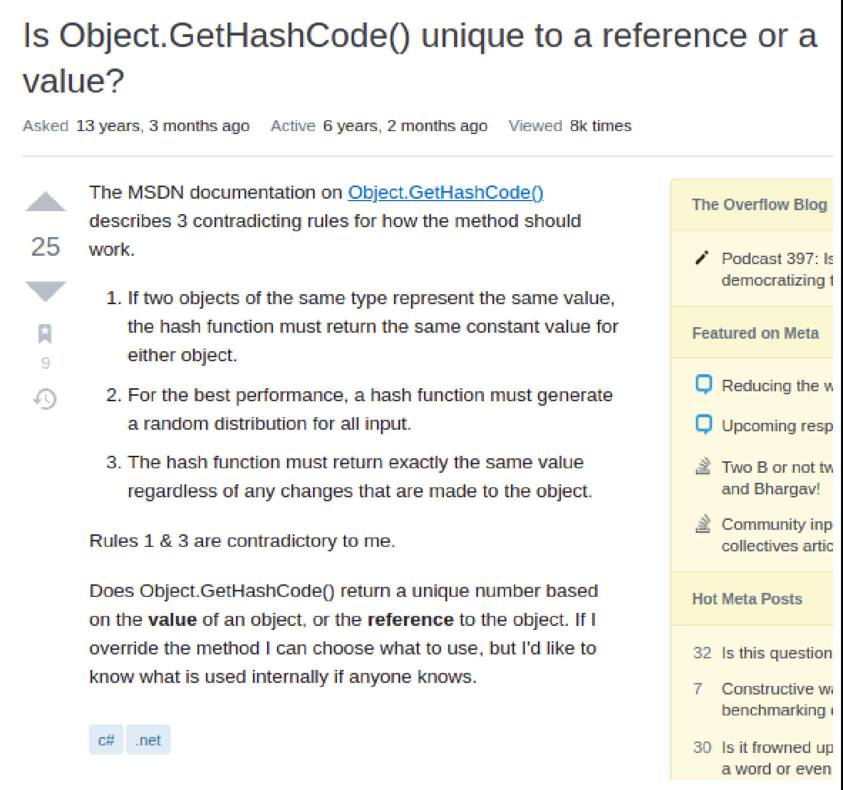
ALL 4 components (title + body + tags + topics) involved

In general, the 0.68% accuracy seems decent.

Params: [0.9, 0.85, 0.3125, 0.3] i.e. high importance to title and body text

- qid for whom we want a duplicate: 441529
- Link to qid: LINK
- **Actual best candidate:** 263400
    - scores: [0.51, 0.11, 0.5 , 0.4]
    - final score: 0.85 | rank 68
- **Chosen best candidate:** 34505
    - Scores: [0.5, 0.5, 0.7, 1]
    - final score: 1.4287 | rank : 1
- We see that the q_chosen beats q_expected in all 4 criteria.
- **Conclusion:** The model is misled by the use of common words like "objects" and "hash function" in q_asked and q_chosen. Even though q_expected is clearly a better match; but the model fails to capture the fact that the q_asked is actually discussing an algorithm and naively allots q_chosen as the higher score (despite it being very different) just because of common words. This can be improved by considering synonyms or word embedding such that phrases like "intelligently built" etc can be linked to words like "algorithm".



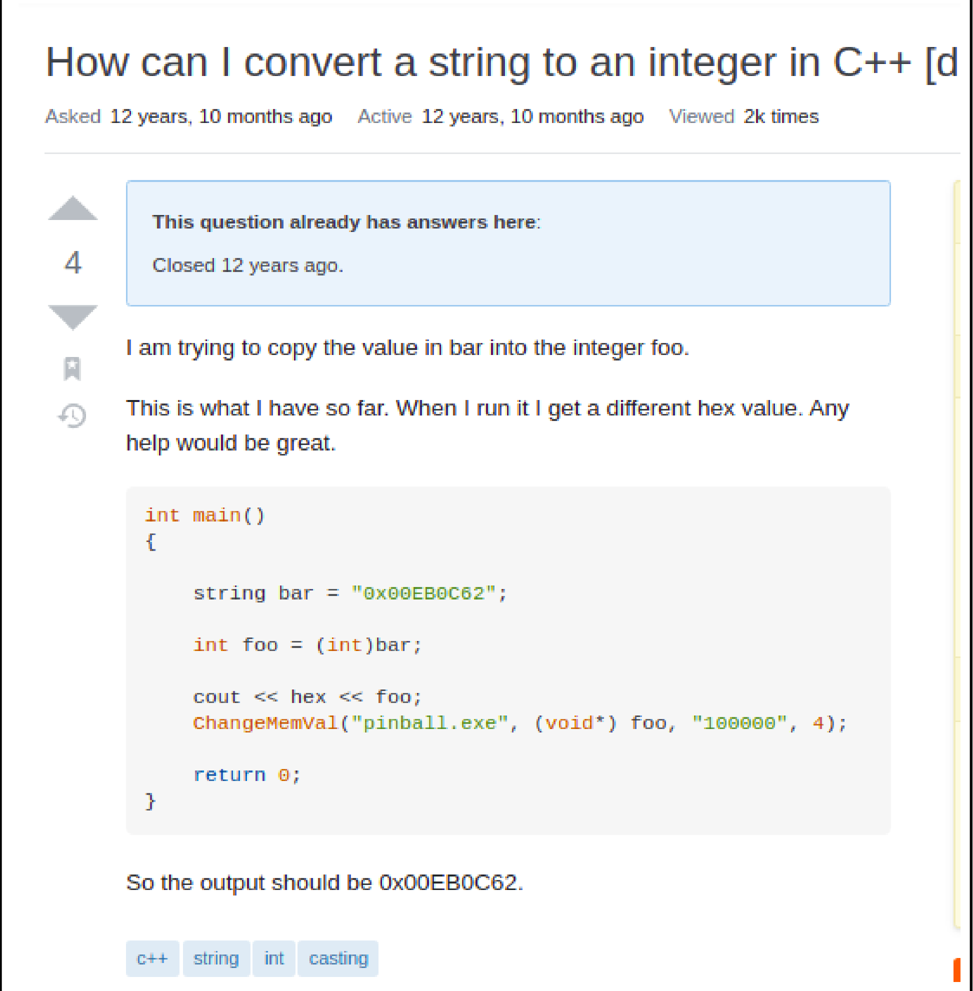
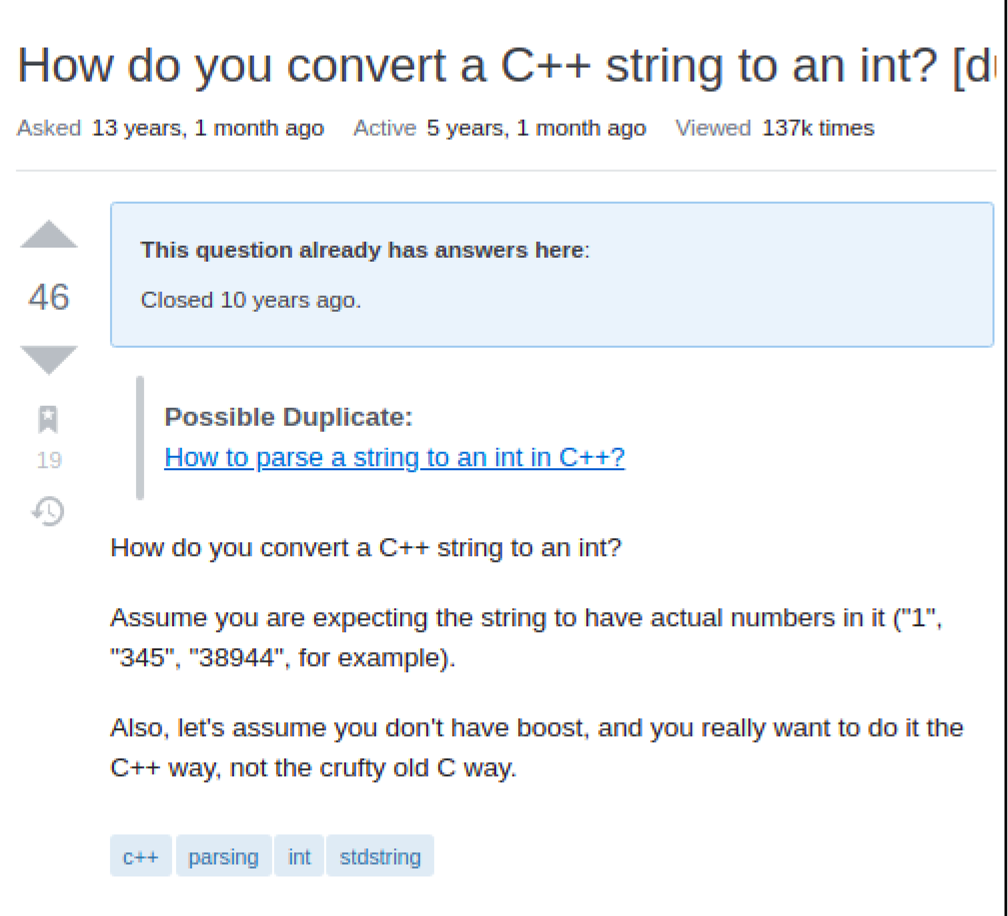
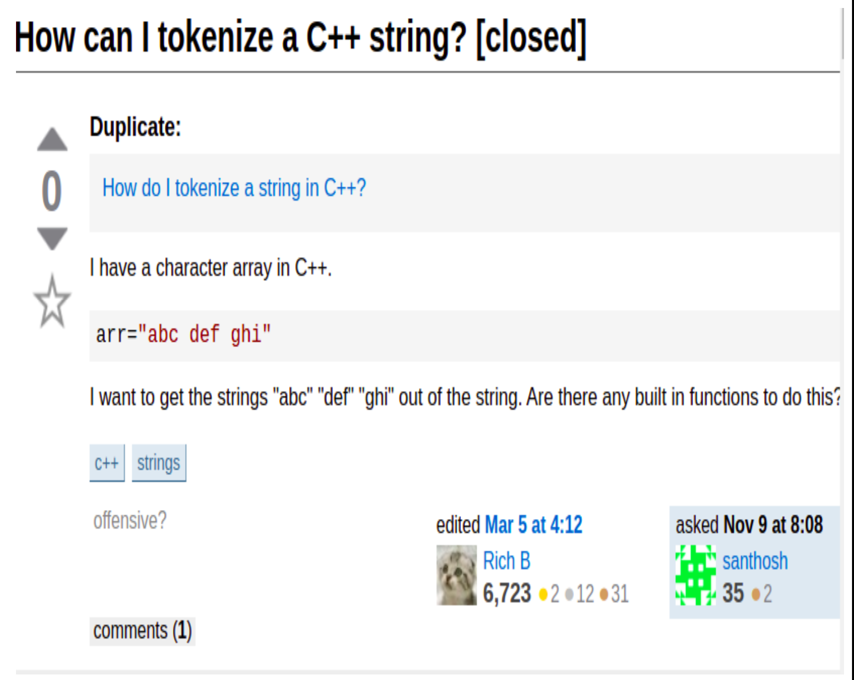
question asked	Expected best	Chosen best
 How do you implement GetHashCode() on objects? [duplicate] Asked 12 years, 10 months ago Active 12 years, 10 months ago Viewed 2k times 4 2 **This question already has answers here:** Closed 12 years ago. **Duplicate**: What is the best algorithm for an overridden System.Object.GetHashCode? If you've written an object with a variety of data-members, how do you intelligently implement GetHashCode()? One developer told me he just XORs (^ operator) the Hash of relevant data-fields, but I am unconvinced this is a "best-practices" implementation. If I had my way, there would be functions Object.CombineHashes(Object[]), and/or Object.CombineHashes(int[]) to help intelligently build hashes of complex objects. How would you write these functions? c# .net The Overflow B Podcast 39 democratizi Featured on Me Reducing tl Upcoming r Two B or no and Bharga Community collectives a Hot Meta Posts 32 Is this ques 7 Constructiv benchmark 30 Is it frowne a word or e	 What is the best algorithm for overriding GetHashCode? Ask Question Asked 13 years, 1 month ago Active 1 month ago Viewed 252k times 1567 895 In .NET, the `GetHashCode` method is used in a lot of places throughout the .NET base class libraries. Implementing it properly is especially important to find items quickly in a collection or when determining equality. Is there a standard algorithm or best practice on how to implement `GetHashCode` for my custom classes so I don't degrade performance? .net algorithm hashcode gethashcode Share Edit Follow edited Sep 18 '19 at 17:55 poke 325k ● 64 ● 498 ● 558 asked Nov 4 '08 at 20:53 bitbonk 46.7k ● 35 ● 177 ● 271	 Is Object.GetHashCode() unique to a reference or a value? Asked 13 years, 3 months ago Active 6 years, 2 months ago Viewed 8k times 25 9 The MSDN documentation on Object.GetHashCode() describes 3 contradicting rules for how the method should work. 1. If two objects of the same type represent the same value, the hash function must return the same constant value for either object. 2. For the best performance, a hash function must generate a random distribution for all input. 3. The hash function must return exactly the same value regardless of any changes that are made to the object. Rules 1 & 3 are contradictory to me. Does Object.GetHashCode() return a unique number based on the **value** of an object, or the **reference** to the object. If I override the method I can choose what to use, but I'd like to know what is used internally if anyone knows. c# .net The Overflow Blog Podcast 397: Is democratizing t Featured on Meta Reducing the w Upcoming resp Two B or not tw and Bhargav! Community inp collectives artic Hot Meta Posts 32 Is this question 7 Constructive w benchmarking 30 Is it frowned up a word or even

ALL components (Except Title Similarity) involved

In general, the 0.53% accuracy WITHOUT using the title is reasonable.

Params: [0 , 0.85, 0.35, 0.64] i.e. without the title, the content now shifts attention to TAGS.

- qid for whom we want a duplicate: 508191 | Link to qid: LINK
- **Actual best candidate:** 200090
    - scores: [0.67 , 0, 0.95 , 0.5]
    - final score: 0.65 | rank 62
- **Chosen best candidate:** 275733
    - Scores: [0.4, 0.129, 0.88, 0.7]
    - final score: 0.87 | rank : 1
- q_chosen beats q_expected in "body_score" which has a high weightage of 0.85.
- **Conclusion:** Due to removal of title, q_expected loses out. Also, because the body in q_asked is almost entirely empty (since we remove the code before inputting to model), there are hardly any words in the "filtered body" of q_asked to give any meaningful insights for matching. However, since the coefficient for the body is high, q_chosen overtakes q_expected. The model where title similarity exists is able to CORRECTLY detect this test case.

question asked	Expected best	Chosen best
		

How can I convert a string to an integer in C++ [d

Asked 12 years, 10 months ago   Active 12 years, 10 months ago   Viewed 2k times

4

This question already has answers here:

Closed 12 years ago.

I am trying to copy the value in bar into the integer foo.

This is what I have so far. When I run it I get a different hex value. Any help would be great.

```
int main()
{

    string bar = "0x00EB0C62";

    int foo = (int)bar;

    cout << hex << foo;
    ChangeMemVal("pinball.exe", (void*) foo, "100000", 4);

    return 0;
}
```

So the output should be 0x00EB0C62.

c++   string   int   casting

How do you convert a C++ string to an int? [d

Asked 13 years, 1 month ago   Active 5 years, 1 month ago   Viewed 137k times

46

This question already has answers here:

Closed 10 years ago.

19

Possible Duplicate:
How to parse a string to an int in C++?

How do you convert a C++ string to an int?

Assume you are expecting the string to have actual numbers in it ("1", "345", "38944", for example).

Also, let's assume you don't have boost, and you really want to do it the C++ way, not the crufty old C way.

c++   parsing   int   stdstring

How can I tokenize a C++ string? [closed]

0

Duplicate:

How do I tokenize a string in C++?

I have a character array in C++.

```
arr="abc def ghi"
```

I want to get the strings "abc" "def" "ghi" out of the string. Are there any built in functions to do this?

c++   strings

offensive?

edited Mar 5 at 4:12
Rich B
6,723 ●2 ●12 ●31

asked Nov 9 at 8:08
santhosh
35 ●2

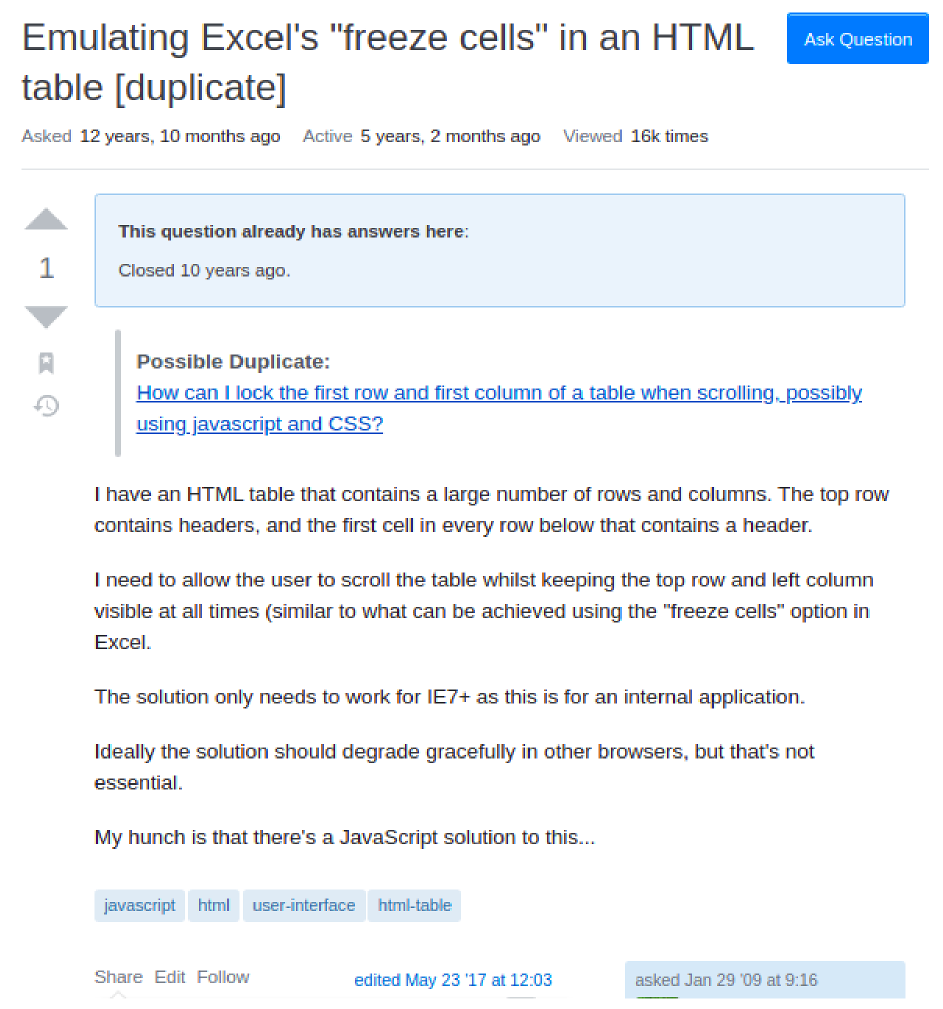
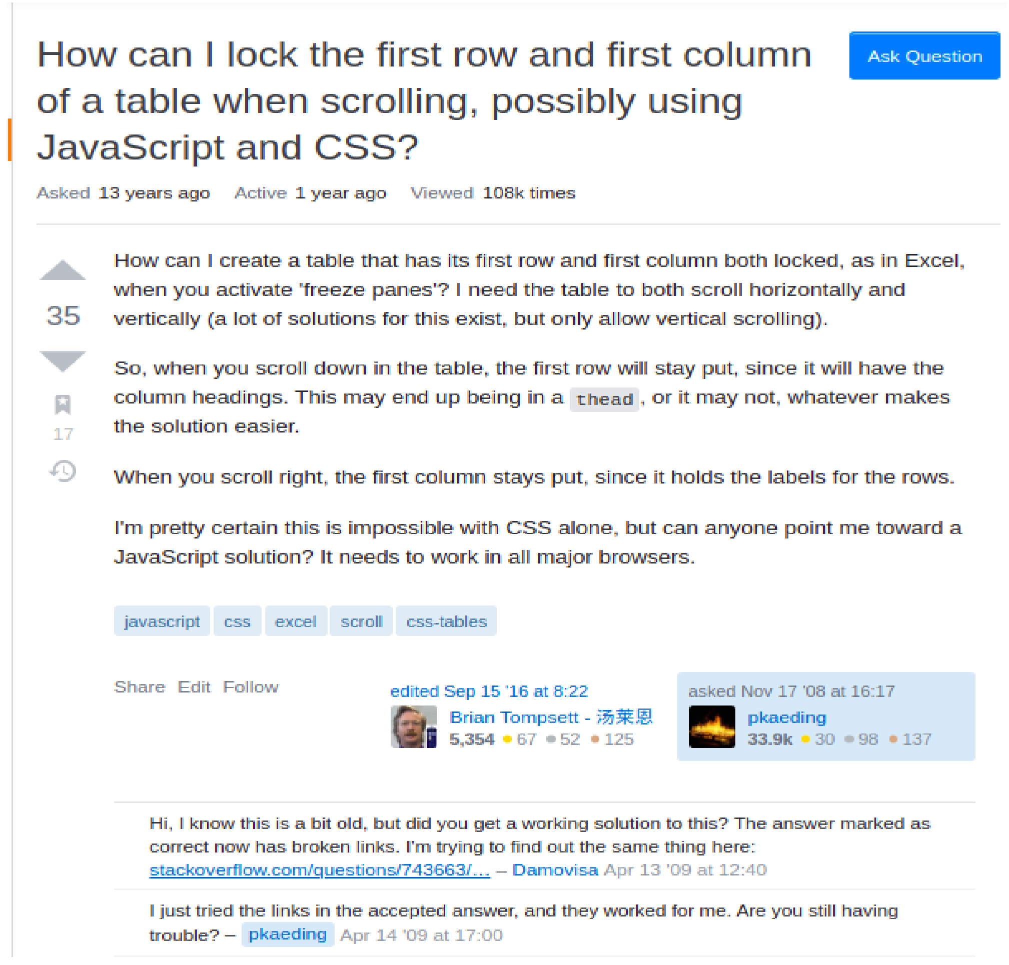
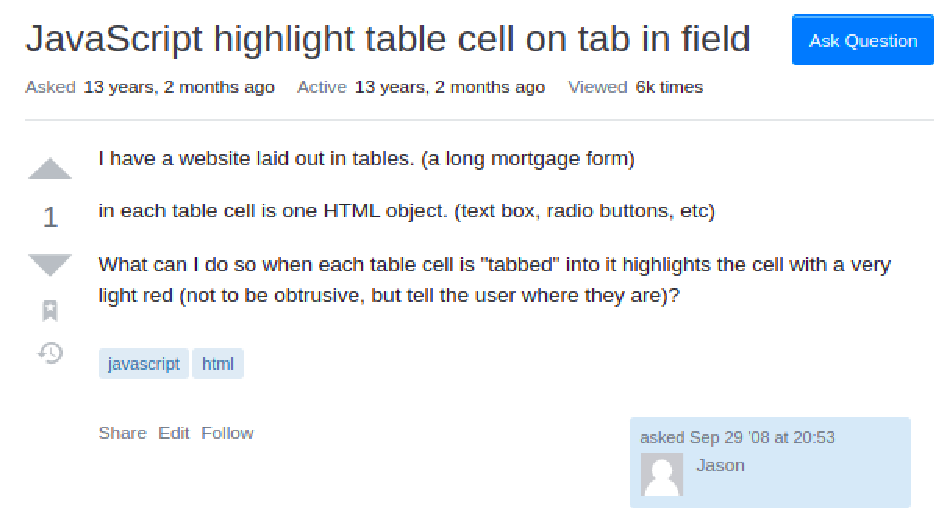
comments (1)

ALL components (Except Description Similarity) involved

In general, the 0.62% accuracy WITHOUT using the body shows that BODY does not play a major role in helping the detector..

Params: [1, 0, 0.5 , 0.6] with emphasis on TITLE and TAGS.

- qid for whom we want a duplicate: 490973 | Link to qid: LINK
- Actual best candidate: 296020
    - scores: [0.15, 0.6, 0.7 , 0.4]
    - final score: 0.8 | rank 42
- Chosen best candidate: 150606
    - Scores: [0.36, 0.26, 0.65, 0.7]
    - final score: 1.11 | rank : 1
- Conclusion: The question wants an answer on how to FREEZE cells in EXCEL using HTML/CSS. Q_chosen gives an answer which is **NOT EVEN REMOTELY related** to EXCEL. lLearly, the model fails to capture the "excel" factor and hence, blunders. The BODY of q_expected and q_ask match to a large extent to give a score of 0.6 . But since coeff for body score is ZERO in this model, the test case gets misdetected by the model.

question asked	Expected best	Chosen best
		

Emulating Excel's "freeze cells" in an HTML table [duplicate]

Ask Question

Asked 12 years, 10 months ago Active 5 years, 2 months ago Viewed 16k times

1

This question already has answers here:

Closed 10 years ago.

Possible Duplicate:
How can I lock the first row and first column of a table when scrolling, possibly using javascript and CSS?

I have an HTML table that contains a large number of rows and columns. The top row contains headers, and the first cell in every row below that contains a header.

I need to allow the user to scroll the table whilst keeping the top row and left column visible at all times (similar to what can be achieved using the "freeze cells" option in Excel.

The solution only needs to work for IE7+ as this is for an internal application.

Ideally the solution should degrade gracefully in other browsers, but that's not essential.

My hunch is that there's a JavaScript solution to this...

javascript html user-interface html-table

Share Edit Follow   edited May 23 '17 at 12:03   asked Jan 29 '09 at 9:16

How can I lock the first row and first column of a table when scrolling, possibly using JavaScript and CSS?

Ask Question

Asked 13 years ago Active 1 year ago Viewed 108k times

35

17

How can I create a table that has its first row and first column both locked, as in Excel, when you activate 'freeze panes'? I need the table to both scroll horizontally and vertically (a lot of solutions for this exist, but only allow vertical scrolling).

So, when you scroll down in the table, the first row will stay put, since it will have the column headings. This may end up being in a `thead`, or it may not, whatever makes the solution easier.

When you scroll right, the first column stays put, since it holds the labels for the rows.

I'm pretty certain this is impossible with CSS alone, but can anyone point me toward a JavaScript solution? It needs to work in all major browsers.

javascript css excel scroll css-tables

Share Edit Follow   edited Sep 15 '16 at 8:22 Brian Tompsett - 汤莱恩 5,354 ● 67 ● 52 ● 125   asked Nov 17 '08 at 16:17 pkaeding 33.9k ● 30 ● 98 ● 137

Hi, I know this is a bit old, but did you get a working solution to this? The answer marked as correct now has broken links. I'm trying to find out the same thing here: stackoverflow.com/questions/743663/… – Damovisa Apr 13 '09 at 12:40

I just tried the links in the accepted answer, and they worked for me. Are you still having trouble? – pkaeding Apr 14 '09 at 17:00

JavaScript highlight table cell on tab in field

Ask Question

Asked 13 years, 2 months ago Active 13 years, 2 months ago Viewed 6k times

1

I have a website laid out in tables. (a long mortgage form)

in each table cell is one HTML object. (text box, radio buttons, etc)

What can I do so when each table cell is "tabbed" into it highlights the cell with a very light red (not to be obtrusive, but tell the user where they are)?

javascript html

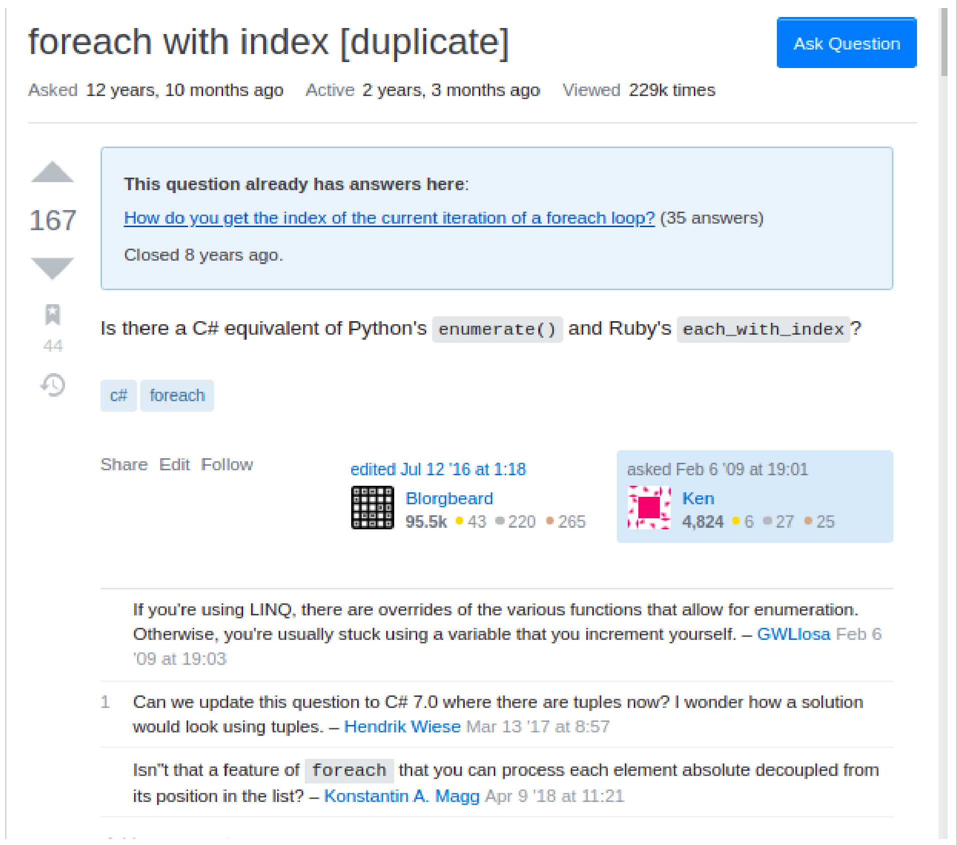
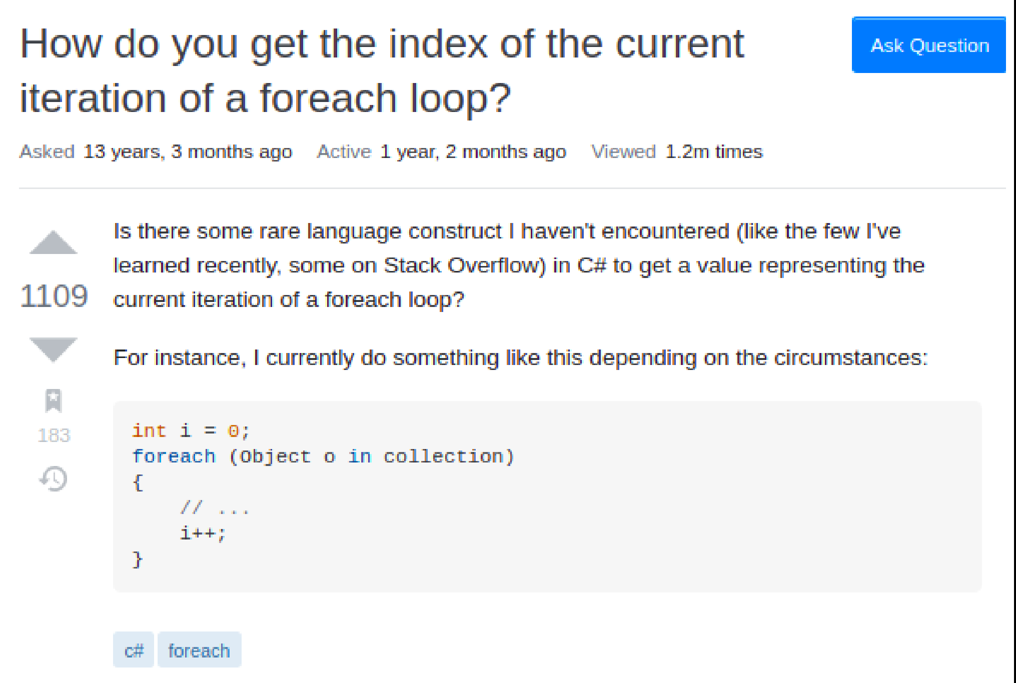
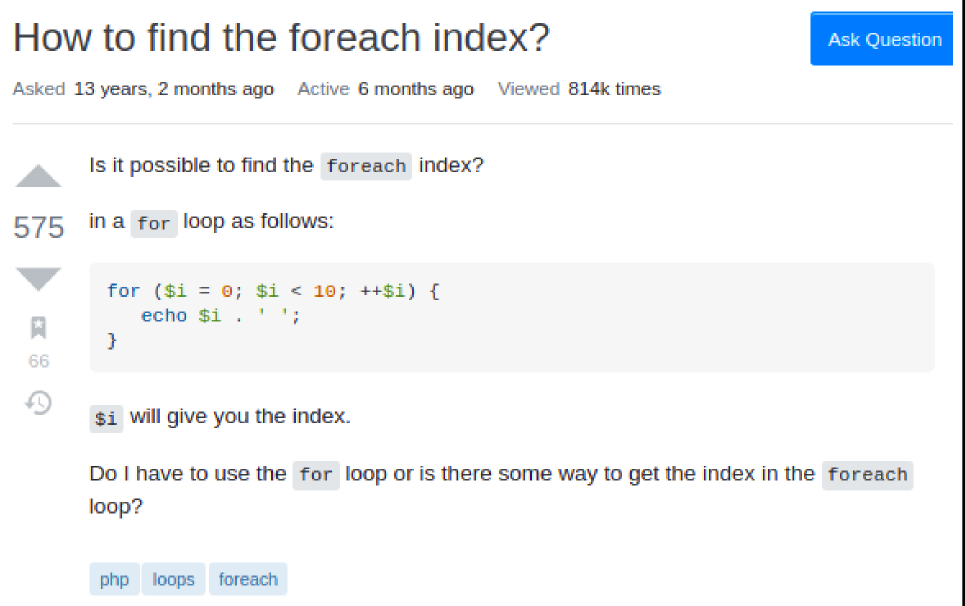
Share Edit Follow   asked Sep 29 '08 at 20:53 Jason

ALL components (Except TAGS Similarity) involved

In general, the accuracy without using tags is 0.51%. But in the pruned search space, tags are automatically taken care of and hence, leads to an accuracy of 0.62% (very minor drop) when tags are not considered.

Params: [0.9 , 0.8 , 0.1 , 0] with emphasis on TITLE and BODY.

- **qid for whom we want a duplicate:** 521687 | Link to qid: LINK
- **Actual best candidate:** 43021
  - scores: [0.5, 0.08, 0.35 , 1]
  - final score: 0.77 | rank 28
- **Chosen best candidate:** 141108
  - Scores: [0.8, 0.25, 0.64, 0.4]
  - final score: 0.999 | rank : 1
- **Conclusion:** q_ask is concerned with foreach in C#. Q_expected is related to foreach in C# but q_chosen is related to foreach in PHP. Both of them are not filtered as both of them have the “foreach” tag common with q_ask. Q_expected has exactly the same tags as q_asked but q_chosen is predicted (even though it **deals with a totally different language**). This is due to the zero value of coefficients of TAGS_SCORE.

question asked	Expected best	Chosen best
		

foreach with index [duplicate]

Ask Question

Asked 12 years, 10 months ago Active 2 years, 3 months ago Viewed 229k times

167

This question already has answers here:

How do you get the index of the current iteration of a foreach loop? (35 answers)

Closed 8 years ago.

Is there a C# equivalent of Python's `enumerate()` and Ruby's `each_with_index`?

c# foreach

44

Share Edit Follow

edited Jul 12 '16 at 1:18
Blorgbeard
95.5k ● 43 ● 220 ● 265

asked Feb 6 '09 at 19:01
Ken
4,824 ● 6 ● 27 ● 25

If you're using LINQ, there are overrides of the various functions that allow for enumeration. Otherwise, you're usually stuck using a variable that you increment yourself. – GWLlosa Feb 6 '09 at 19:03

1 Can we update this question to C# 7.0 where there are tuples now? I wonder how a solution would look using tuples. – Hendrik Wiese Mar 13 '17 at 8:57

Isn"t that a feature of `foreach` that you can process each element absolute decoupled from its position in the list? – Konstantin A. Magg Apr 9 '18 at 11:21

How do you get the index of the current iteration of a foreach loop?

Ask Question

Asked 13 years, 3 months ago Active 1 year, 2 months ago Viewed 1.2m times

1109

Is there some rare language construct I haven't encountered (like the few I've learned recently, some on Stack Overflow) in C# to get a value representing the current iteration of a foreach loop?

For instance, I currently do something like this depending on the circumstances:

```
int i = 0;
foreach (Object o in collection)
{
    // ...
    i++;
}
```

183

c# foreach

How to find the foreach index?

Ask Question

Asked 13 years, 2 months ago Active 6 months ago Viewed 814k times

575

Is it possible to find the `foreach` index?

in a `for` loop as follows:

```
for ($i = 0; $i < 10; ++$i) {
   echo $i . ' ';
}
```

66

`$i` will give you the index.

Do I have to use the `for` loop or is there some way to get the index in the `foreach` loop?

php loops foreach

Further Exploration

We attempted to further improve the best results obtained using the methodology proposed in the paper. For improving accuracy we mainly experiment with 2 techniques: Increasing the number of features while calculating the composer score and further improving the predictions of the greedy composer with the help of NLP in the hopes of capturing the semantic and lexical meaning of the questions.

Jaccard coefficient as a similarity metric

Why did we do it?

Our hypothesis was that since the **composer is treating each of its similarities as separate entities**, it might miss cases where some words that are present in the title for one question may be present in another section for another question. To incorporate these cases inside the model we used the Jaccard coefficient.

How did we do it?

We combined the processed title, body and tags as a single bag of words for all the questions. Then we computed the Jaccard score between each duplicate question and a question candidate of being the master of the duplicate question.

$$J(A,\ B)\ =\ \frac{|A \cap B|}{|A \cup B|}$$

What result did we get?

We trained on the first 300 questions including the fifth parameter as Jaccard similarity.

alpha	beta	gamma	delta	Jaccard	recall-rate@20 Training	recall-rate@20 Testing
0.45	0.35	0.25	0.2941	1.0	0.763	0.65

We found that the composer predicts Jaccard to be a very important parameter (with a value of 1) while predicting a candidate question to be a duplicate's master. One reason for this might be that the Jaccard score looks at each entity i.e. title, body & tags together in a holistic view thereby providing much more information about the pairs in consideration. We can also state that the Jaccard score somewhat represents the title, body and tags similarity components together.

But we observe that the testing recall does not increase significantly. From this, we can conclude that, generally, all the three entities are disjoint and do not provide any significant extra information when combined together.

Re-ranking using BERT embeddings

Why did we do it?

After Jaccard, we felt that increasing the number of features more would only increase the complexity of the model and will not improve results significantly. Also applying a computationally heavy approach like embeddings generation using Word2Vec, or BERT is **not feasible for fast training and training time will take a hit**. In this scenario, reranking only the smaller pruned rank list created using DupPredictor is easier and much more computationally feasible.

How did we do it?

The DupPredictor was initially trained on the first 300 duplicate questions (Best Params reported above). For each duplicate question, we calculate the **top 30 predictions on the basis of the parameters estimated from the DupPredictor Model.** The actual master questions which have not been predicted in the **top 30** were then appended at the end of the predictions list and this list is the initial ranking list for a duplicate according to DupPredictor.

Now, the title and body texts are cleaned and then Bert embeddings are calculated for each text phrase using a pre-trained hugging face transformer. For reranking the questions, the weighted cosine similarity metric is calculated using the formula as described below:

$$BertScore\ =\ \alpha \times CosineSim(BERT(Q1_{Title}),\ BERT(Q2_{Title}))\ +\ \beta \times CosineSim(BERT(Q1_{Body}),\ BERT(Q2_{Body}))$$

Here Q1 and Q2 represent the candidate and duplicate questions for which we calculate the BERT embeddings. The weights alpha and beta for calculating this score are found on the basis of observations and the findings from the DupPredictor model. Thus, we set the value of α **= 0.6** and β **= 0.5** giving more weight to the title (DupPredictor gives more weightage to Title generally). Finally, all the predictions are re-ranked using this newly calculated BertScore.

What result did we get?

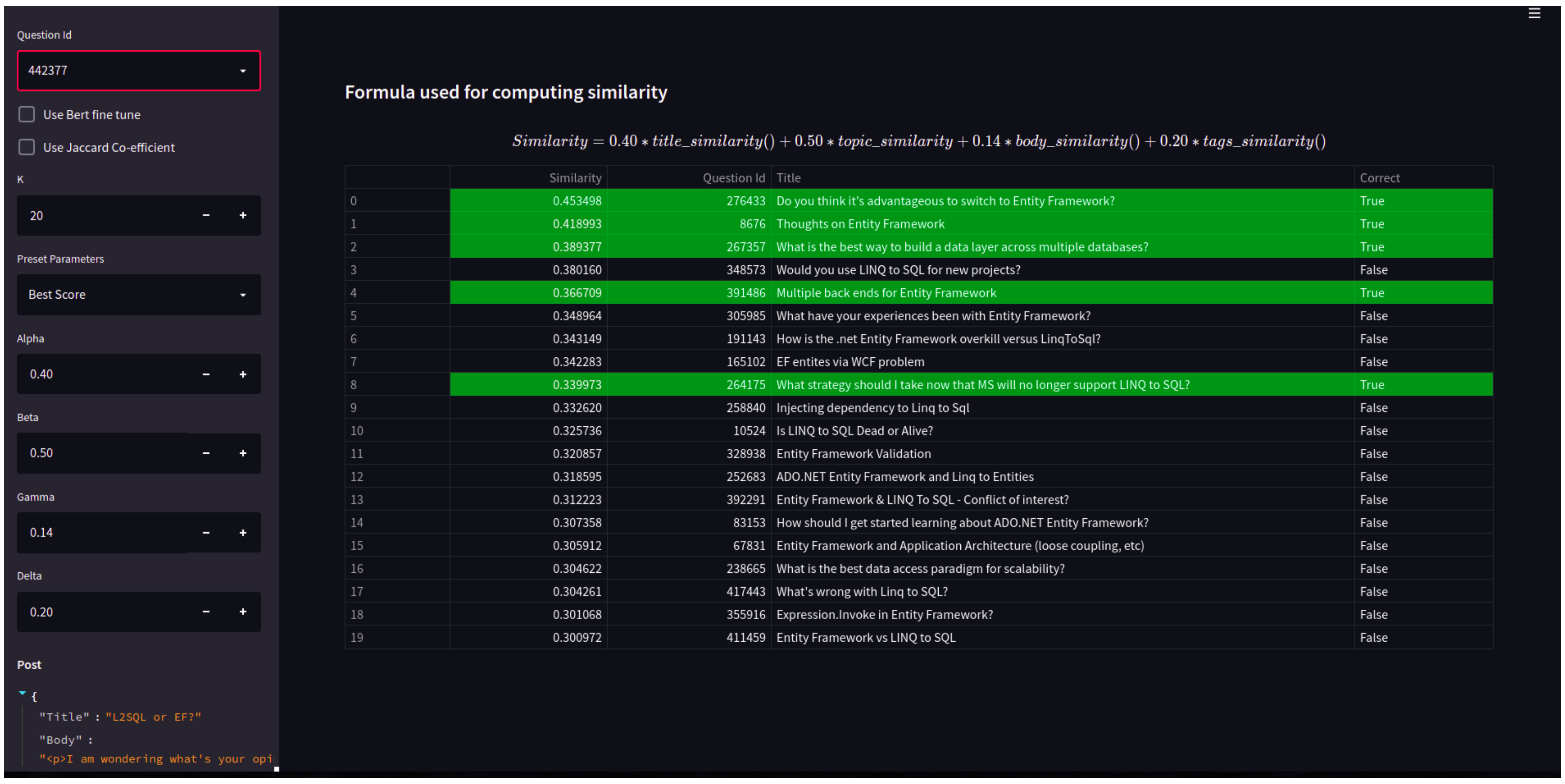
We observe that re-ranking on the basis of BERT embeddings helps the model a lot.

- Firstly, we improved the recall-rate@20 of the DupPredictor from **0.65 to 0.88** on the test set, which is a very significant improvement for quite minimal computation. From this, we can conclude that BERT's power to understand the semantic and lexical meaning of the sentences greatly helps to improve the quality of the predictions.

- Secondly, we observe that for **29** duplicate questions, we are able to improve the richness of the top 20 predictions by including **32** more correct candidate questions which were initially in the top 30 but not in the top 20 predictions.
- Surprisingly, we are also able to include **2** questions in the top 20 which were initially not even in the top 30.

Website to view the working of our model

We have created a website to easily understand the importance of each score/similarity and play around with the various parameters in the composer to understand how the ranking of the various candidate questions change.



Question Id: 442377
Use Bert fine tune: ☐
Use Jaccard Co-efficient: ☐
K: 20
Preset Parameters: Best Score
Alpha: 0.40
Beta: 0.50
Gamma: 0.14
Delta: 0.20

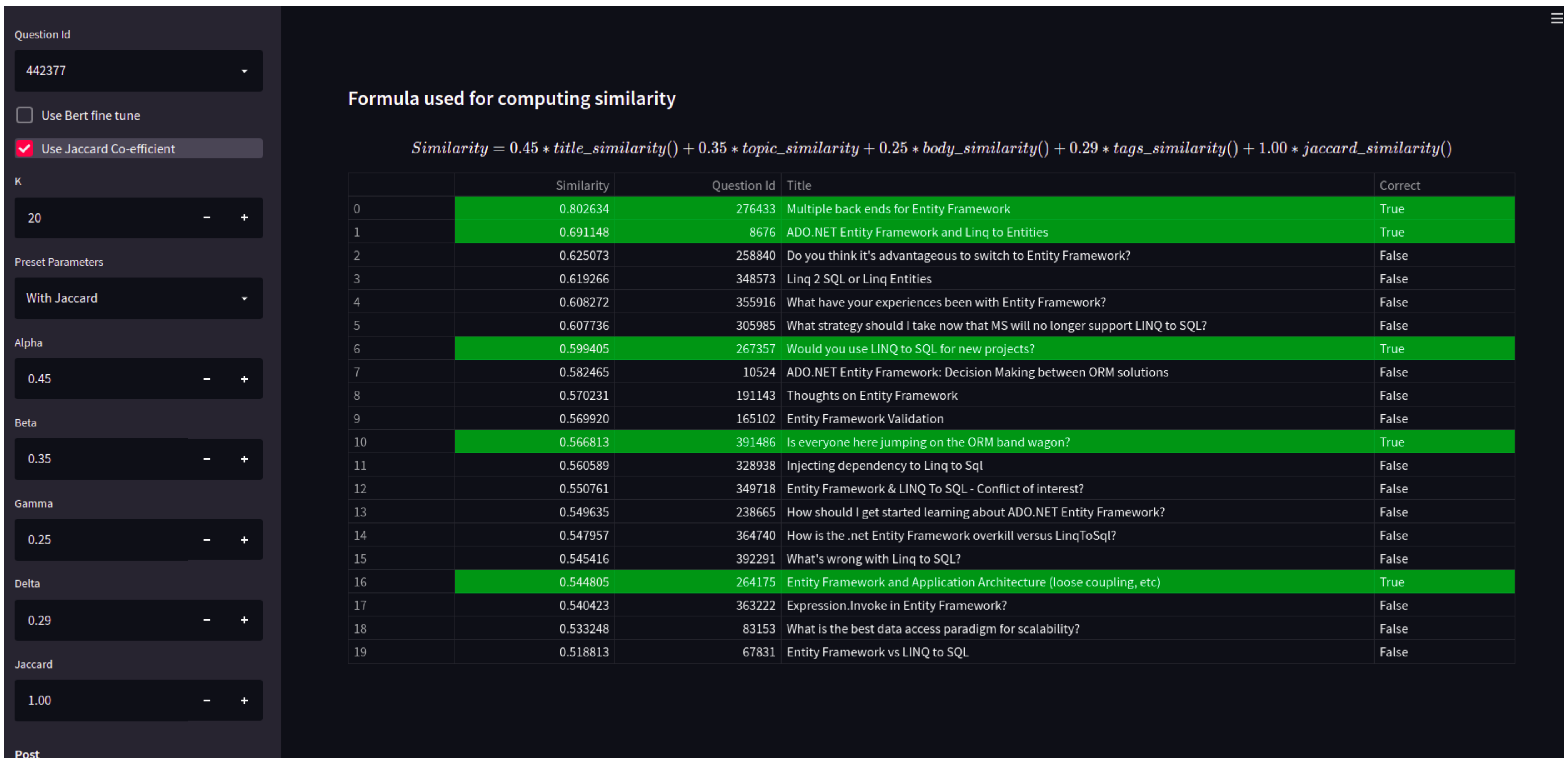
Post

```
{
  "Title" : "L2SQL or EF?"
  "Body" :
  "<p>I am wondering what's your opi
```

Formula used for computing similarity

$$Similarity = 0.40 * title\_similarity() + 0.50 * topic\_similarity + 0.14 * body\_similarity() + 0.20 * tags\_similarity()$$

	Similarity	Question Id	Title	Correct
0	0.453498	276433	Do you think it's advantageous to switch to Entity Framework?	True
1	0.418993	8676	Thoughts on Entity Framework	True
2	0.389377	267357	What is the best way to build a data layer across multiple databases?	True
3	0.380160	348573	Would you use LINQ to SQL for new projects?	False
4	0.366709	391486	Multiple back ends for Entity Framework	True
5	0.348964	305985	What have your experiences been with Entity Framework?	False
6	0.343149	191143	How is the .net Entity Framework overkill versus LinqToSql?	False
7	0.342283	165102	EF entites via WCF problem	False
8	0.339973	264175	What strategy should I take now that MS will no longer support LINQ to SQL?	True
9	0.332620	258840	Injecting dependency to Linq to Sql	False
10	0.325736	10524	Is LINQ to SQL Dead or Alive?	False
11	0.320857	328938	Entity Framework Validation	False
12	0.318595	252683	ADO.NET Entity Framework and Linq to Entities	False
13	0.312223	392291	Entity Framework & LINQ To SQL - Conflict of interest?	False
14	0.307358	83153	How should I get started learning about ADO.NET Entity Framework?	False
15	0.305912	67831	Entity Framework and Application Architecture (loose coupling, etc)	False
16	0.304622	238665	What is the best data access paradigm for scalability?	False
17	0.304261	417443	What's wrong with Linq to SQL?	False
18	0.301068	355916	Expression.Invoke in Entity Framework?	False
19	0.300972	411459	Entity Framework vs LINQ to SQL	False



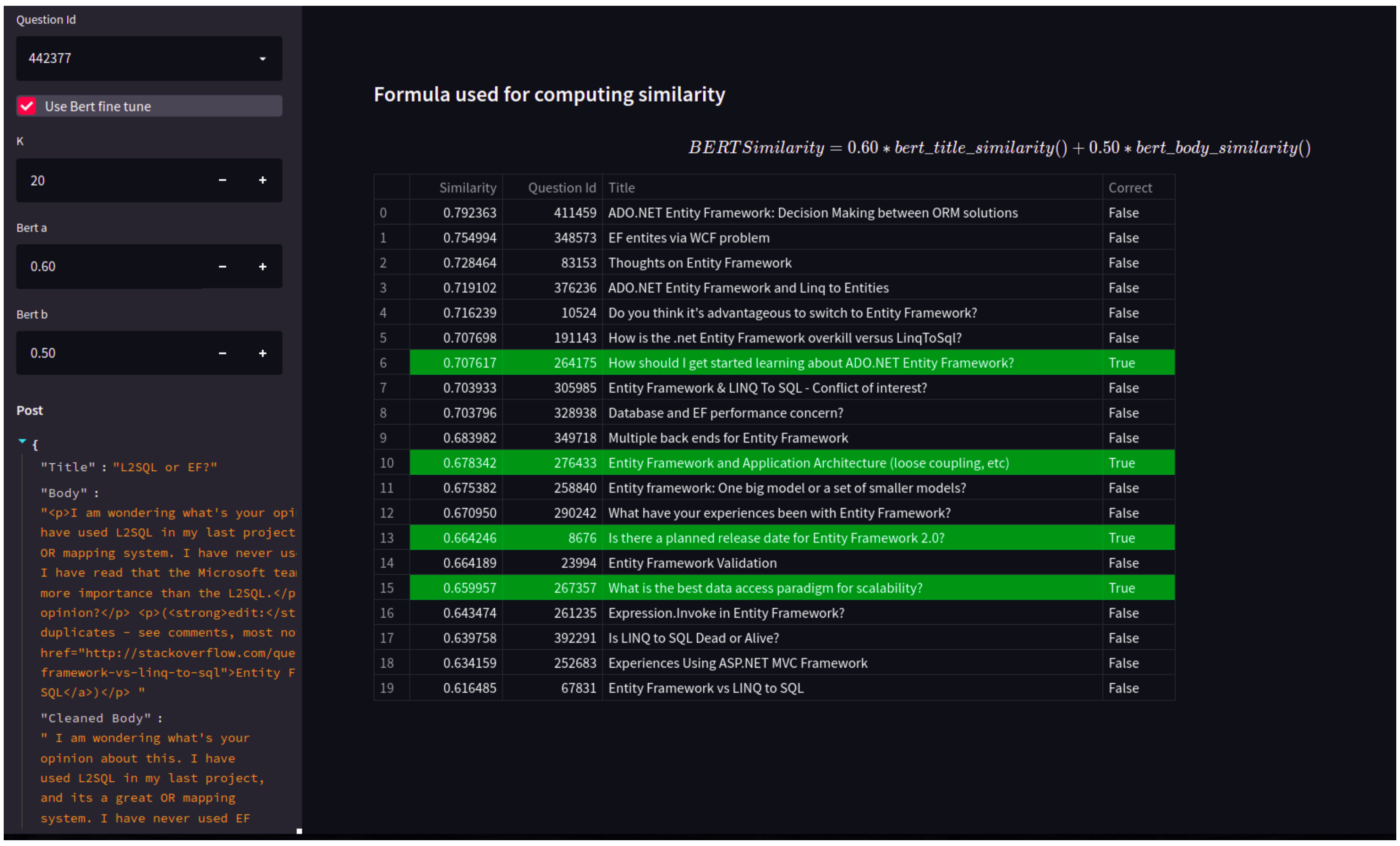
Question Id: 442377
Use Bert fine tune: ☐
Use Jaccard Co-efficient: ✓
K: 20
Preset Parameters: With Jaccard
Alpha: 0.45
Beta: 0.35
Gamma: 0.25
Delta: 0.29
Jaccard: 1.00

Post

Formula used for computing similarity

$$Similarity = 0.45 * title\_similarity() + 0.35 * topic\_similarity + 0.25 * body\_similarity() + 0.29 * tags\_similarity() + 1.00 * jaccard\_similarity()$$

	Similarity	Question Id	Title	Correct
0	0.802634	276433	Multiple back ends for Entity Framework	True
1	0.691148	8676	ADO.NET Entity Framework and Linq to Entities	True
2	0.625073	258840	Do you think it's advantageous to switch to Entity Framework?	False
3	0.619266	348573	Linq 2 SQL or Linq Entities	False
4	0.608272	355916	What have your experiences been with Entity Framework?	False
5	0.607736	305985	What strategy should I take now that MS will no longer support LINQ to SQL?	False
6	0.599405	267357	Would you use LINQ to SQL for new projects?	True
7	0.582465	10524	ADO.NET Entity Framework: Decision Making between ORM solutions	False
8	0.570231	191143	Thoughts on Entity Framework	False
9	0.569920	165102	Entity Framework Validation	False
10	0.566813	391486	Is everyone here jumping on the ORM band wagon?	True
11	0.560589	328938	Injecting dependency to Linq to Sql	False
12	0.550761	349718	Entity Framework & LINQ To SQL - Conflict of interest?	False
13	0.549635	238665	How should I get started learning about ADO.NET Entity Framework?	False
14	0.547957	364740	How is the .net Entity Framework overkill versus LinqToSql?	False
15	0.545416	392291	What's wrong with Linq to SQL?	False
16	0.544805	264175	Entity Framework and Application Architecture (loose coupling, etc)	True
17	0.540423	363222	Expression.Invoke in Entity Framework?	False
18	0.533248	83153	What is the best data access paradigm for scalability?	False
19	0.518813	67831	Entity Framework vs LINQ to SQL	False


Question Id

442377

✓ Use Bert fine tune

K

20

Bert a

0.60

Bert b

0.50

Post

```
{
  "Title" : "L2SQL or EF?"
  "Body" :
  "<p>I am wondering what's your opi
  have used L2SQL in my last project
  OR mapping system. I have never us
  I have read that the Microsoft tea
  more importance than the L2SQL.</p
  opinion?</p> <p>(<strong>edit:</st
  duplicates - see comments, most no
  href="http://stackoverflow.com/que
  framework-vs-linq-to-sql">Entity F
  SQL</a>)</p> "
  "Cleaned Body" :
  " I am wondering what's your
  opinion about this. I have
  used L2SQL in my last project,
  and its a great OR mapping
  system. I have never used EF
```

Formula used for computing similarity

$$BERTSimilarity = 0.60 * bert\_title\_similarity() + 0.50 * bert\_body\_similarity()$$

	Similarity	Question Id	Title	Correct
0	0.792363	411459	ADO.NET Entity Framework: Decision Making between ORM solutions	False
1	0.754994	348573	EF entites via WCF problem	False
2	0.728464	83153	Thoughts on Entity Framework	False
3	0.719102	376236	ADO.NET Entity Framework and Linq to Entities	False
4	0.716239	10524	Do you think it's advantageous to switch to Entity Framework?	False
5	0.707698	191143	How is the .net Entity Framework overkill versus LinqToSql?	False
6	0.707617	264175	How should I get started learning about ADO.NET Entity Framework?	True
7	0.703933	305985	Entity Framework & LINQ To SQL - Conflict of interest?	False
8	0.703796	328938	Database and EF performance concern?	False
9	0.683982	349718	Multiple back ends for Entity Framework	False
10	0.678342	276433	Entity Framework and Application Architecture (loose coupling, etc)	True
11	0.675382	258840	Entity framework: One big model or a set of smaller models?	False
12	0.670950	290242	What have your experiences been with Entity Framework?	False
13	0.664246	8676	Is there a planned release date for Entity Framework 2.0?	True
14	0.664189	23994	Entity Framework Validation	False
15	0.659957	267357	What is the best data access paradigm for scalability?	True
16	0.643474	261235	Expression.Invoke in Entity Framework?	False
17	0.639758	392291	Is LINQ to SQL Dead or Alive?	False
18	0.634159	252683	Experiences Using ASP.NET MVC Framework	False
19	0.616485	67831	Entity Framework vs LINQ to SQL	False

Work Distribution

Member	Work
Anmol	Data Cleaning, Error Analysis on ablation results, Search Space Pruning, insights on fitting BERT in the pipeline
Gurkirat	Data Preprocessing, Database Handling and Sorting, Paper Experiments and Website Creation
Shrey	Data Preprocessing, Paper Experiments and Model, Ablation, Jaccard and BERT explorations
Pratyush	Data Preprocessing, Paper Experiments and Model, Ablation, Jaccard and BERT explorations

Limitations

All the limitations and the challenges we faced in the course of this project, mainly boil down to the computational limits of the process.

Size of training and testing set

The training and testing set size was extremely large.

Since we have to compute for every duplicate question the pairwise scores for all the questions that came before it, the number of pairwise scores to compute increases exponentially as we try to increase the number of duplicate questions. Currently, we have around 60,000 candidate questions per duplicate question in our training set (300 duplicate questions) and 80,000 candidate questions per duplicate question in the test set (100 duplicate questions).

The time taken to compute the scores for the training set **(300 questions)** came around **8 hours**. And the time taken to compute scores for the test set (100 questions) was **5 hours** since the test set had more even a more number of candidate questions per duplicate. From this trend, we can see that the compute time increases exponentially and using more questions becomes incredibly difficult.

Estimation of parameters

Even though we managed to estimate the majority of the parameters, since the procedure for estimation is greedy, it takes around 4 hours for each training. Therefore, we were unable to compute the parameters for recall@5.

Bert Training

Creating embeddings for each post in the training set using either Word2Vec or BERT was exceedingly compute-heavy as well. The estimated time for computing this was approximately 20 hours and consumed approximately 14GB of RAM making it impossible to run our laptops. **Even on ADA, we weren't able to allocate more than 4GB of RAM to a batch job** and the process kept on failing. Due to this, we weren't able to train using BERT and thus resorted to a re-ranking approach to reduce the question search space.

Due to these limitations, only one laptop was able to run the training steps making the entire process extremely sequential.