Multilingual Similarity

Weston, Travis
twestone4@uncc.edu

Yermakovich, Rebekah
ryermako@uncc.edu

Bhuvana Sree Garikapati
bgarikap@uncc.edu

Shreyas Lokesha
slokesha@uncc.edu

Mohammed Sharik U Zama
mshariku@uncc.edu

September 2021

1 Abstract

Document similarity determines how similar a given set of documents are. In this project we are planning to perform a multilingual document level similarity analysis task for news from several domains. The data required is obtained from the contest main page. The training data is released in batches throughout the contest's preparation period. Our approach for this task is to use a pairwise rating on a 4-point scale. And, do a step-by-step process from extraction of data to analyzing each article. Libraries like gensim, NLTK, spaCy and Beautifulsoup are used in this project. The final aim of this project is to develop a system that can identify multilingual news articles that give similar information.

2 Introduction

The multilingual similarity project is a document level similarity analysis task which primarily aims at identifying news articles in different languages covering the same story. To compare different stories, we make use of a pairwise rating on a 4-point scale.

There are three primary steps involved in this process which are:

1. Extract text from news article webpages
2. Data Cleansing
3. Entity and Topic extraction

In the first step, we extract data from the websites using NLTK and beautiful soup. In the second step we try to remove as many stop words as possible which makes comparison easier. Stop words include words like a,the,an,and etc. After Cleaning and Tokenizing the data, we check for the correlation and the means.
Post this, we now analyze the articles and compare them with each other using the pairwise elimination. Currently we are trying to analyze articles and compare them.

3 Problem Statement

This SemEval task aims to develop systems that identify multilingual news articles that provide similar information.

4 Data Description

4.1 Input Data

The training data was provided in CSV format, containing the following columns:

- Pair-ID
- Link1
- Link2
- Ia_link1
- Ia_link2
- Geography
- Entities
- Time
- Narrative
- Overall
- Style
- Tone

The entries in the pair-id column consist of a sequence of numerals separated by a '_' character to denote an ID given to a pair of URL links given for comparison. The fields Ia_link1 and Ia_link2 consist of the Internet archive links corresponding to the URLs under columns link1 and link2 respectively.

The columns, Geography, Entities, Time, Narrative, Overall, Style and Tone are scoring parameters given as a part of the input data. These fields have been scored by individuals who were asked to go through the content in each of those links and compare the contents with each other. The scores have a range from 1-4 wherein a score of 4 would signify a high degree of similarity and 1 would denote the converse.

Geography - A score under this field would denote how close the events/content described in both the links were in terms of where they had occurred.

Entities - A score under this field would denote similarities between individuals or organizations that were involved in the news article.

Time - A value under this section would give us an estimate of how close or far the news was reported in a chronological sense. The scores under the overall section would denote how similar or different the pair of news articles are from each other.

4.2 Output Data

For a successful submission, the algorithm being developed must take into consideration the content in the links in the input csv file and generate an overall similarity score for a given pair-id. This data is later scripted into a .csv file with 2 columns, pair-id and overall similarity score.

4.3 Subsection motivating your approach

Our approach is devised of five phases: extraction, translation, tokenization, and comparison. This approach gives us optimal modularity, improving our ability to fine tune results.
Cosine similarity is a measure of how documents are closely related to one another and is expressed as the dot product of 2 document vectors divided by the magnitude of the normalized form of those document vectors.

4.4 Implementation

4.4.1 Extraction

The competition organizers provided a cli, which builds a directory with HTML and JSON files. Our application utilizes the glob library to locate the articles

associated files. The HTML files are parsed utilizing the Newspaper3k library to extract key text elements such as the article, title, and metadata tags. The JSON files are parsed via the python JSON library. The key JSON elements that we extract are in a list that we can modify to fine tune data elements.

4.4.2 Translation

All text that is not labeled as English "en" in the source training data file are passed to the translation method. Translations are per- formed by Google Translate python api.

4.4.3 Tokenization

The Spacy library is used as our NLP model and has a multiple pipeline approach that tokenizes and removes stop words of all text. Spacy also has the ability to extract entities, to be analyzed and compared for future consideration.

4.4.4 Comparison

Currently we load the bag of words generated from the Spacy model into an LDA model for topic extraction. The top n topics identified in the LDA model for the tow articles being compared are vectorized and compared via cosine similarity. We manually adjust the threshold for the cosine similarity score to get the based results for the 1-4 overall similarity score for the two articles.

4.4.5 Scoring

After performing the step pertaining to comparison of the model, we set a score denoting the similarity of the contents within the two article links provided as an input. A final score is returned by the algorithm carrying out the comparison (cosine similarity in this case). We are then randomizing the score before making the first submission using some in-built python methods.

4.5 Subsection details of your experiments

4.5.1 Model Testing

Currently we are experimenting with different similarity methods and thresholds for scoring/scaling. This week we experimented with TFIDF models to get a similarity between the two documents. We take the mean of all of the words in the model and compare the mean score from the TFIDF model to the LDA Topic extraction cosine similarity score. Our results from the TFIDF mean similarity score have been unreliable and requires adjustments as document scores for similar articles was not consistent. Next steps with this experiment is to similarly extract the top n topics from the model and only get

the mean of the words associated with those topics.Figure 1 2 below show the results of comparing the topics extracted from two individual LDA modles compared to a single LDA model and randomly generated scores using cosine similarity. As you can see documents that had more text and more prominent topics had greater influence and skewed the comparison results when the documents were compared in a single model.

```
Comparing JSON:  1484084337  Files:  2  Lang:  en  TO  1484110209 Files:  2  Lang:  en
Randomly Generated Score:  2.896463240735085
Cosine Similarity - Documents in single model:  1.0000000000000004 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.7540368943703601 Rating:  3
**********
TF-IDF MEAN :  0.07
**********
Comparing JSON:  1484396422  Files:  2  Lang:  en  TO  1483924666 Files:  2  Lang:  en
Randomly Generated Score:  2.7790245685689245
Cosine Similarity - Documents in single model:  0.9999999989749992 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.7082555463545301 Rating:  2
**********
TF-IDF MEAN :  0.08
**********
Comparing JSON:  1484698254  Files:  2  Lang:  en  TO  1483758694 Files:  2  Lang:  en
Randomly Generated Score:  3.742648273143918
Cosine Similarity - Documents in single model:  1.0000000000000033 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.9507653410659087 Rating:  4
**********
TF-IDF MEAN :  0.07
**********
Comparing JSON:  1576314516  Files:  2  Lang:  en  TO  1576455088 Files:  2  Lang:  en
Randomly Generated Score:  3.769708098499486
Cosine Similarity - Documents in single model:  0.5237529067458019 Rating:  2
Cosine Similarity - Seperate LDA Models:  0.9191832852612968 Rating:  4
**********
TF-IDF MEAN :  0.07
**********
Comparing JSON:  1484036253  Files:  2  Lang:  en  TO  1483894099 Files:  2  Lang:  en
Randomly Generated Score:  2.7800022044760397
Cosine Similarity - Documents in single model:  1.0 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.7467880324494335 Rating:  2
**********
TF-IDF MEAN :  0.07
**********
Comparing JSON:  1484189120  Files:  2  Lang:  en  TO  1484113136 Files:  2  Lang:  en
Randomly Generated Score:  2.7982576567836155
Cosine Similarity - Documents in single model:  0.9999999990811832 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.6029088128813601 Rating:  2
**********
TF-IDF MEAN :  0.07
**********
Comparing JSON:  1484034982  Files:  2  Lang:  en  TO  1483785560 Files:  2  Lang:  en
Randomly Generated Score:  2.93004416402147
Cosine Similarity - Documents in single model:  0.9999999999999998 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.6375355591062043 Rating:  2
**********
TF-IDF MEAN :  0.06
**********
```

Figure 1: Experiment with cosine similarity scores, thresholds between single and multiple LDA models, and randomly generated scores.

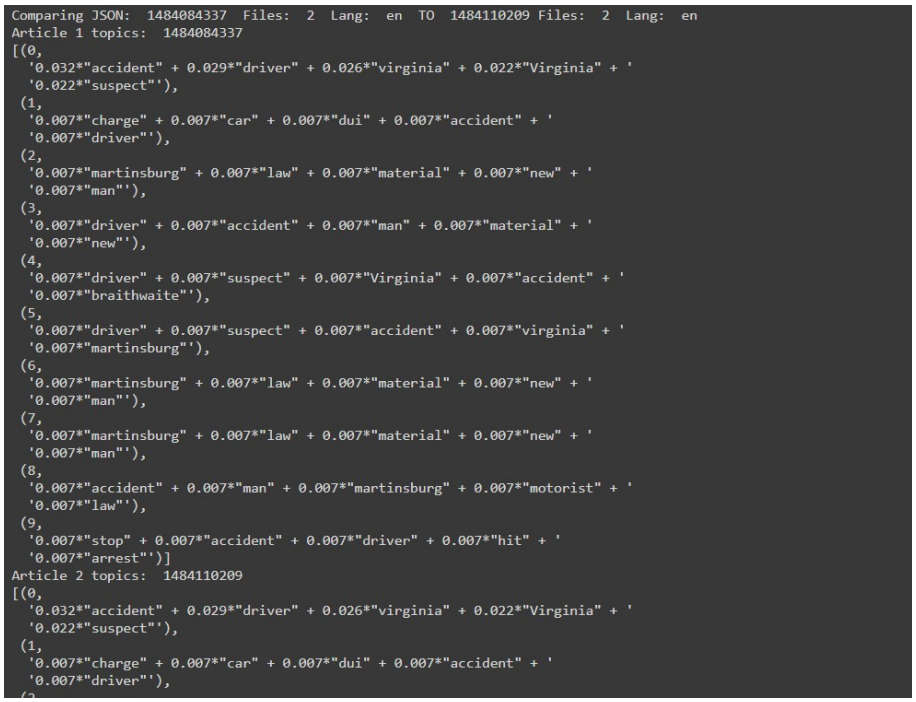

Figure 2: Topic extraction testing between LDA scores for the corpus

4.5.2 Bert Article Comparison

As defined by BERT's research team:

> BERT stands for Bidirectional Encoder Representations from Transformers. It is designed to pre-train deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of NLP tasks.

Our team used BERT to execute Cosine and Euclidean distance of a corpus composed of the main text extracted from two different news articles. What we we found is that BERT calculated results very similar to the LDA topic extraction results from our basline solution as shown in the figures below.

```
def bert_experiment(jfile1,jfile2):
  #Extract the main text from the Articles
  text1 = json_text(jfile1[0][0],jfile1[0][1])
  text2 = json_text(jfile2[0][0],jfile2[0][1])
  #Create the Corpus containing the text from both articles
  corpus = [text1,text2]
  #Convert the corpus to a data frame with two rows and single column 'documents'
  documents_df=pd.DataFrame(corpus,columns=['documents'])
  #Lowercase all text, remove empty space, and remove stop words. Place the clean text in the documents_cleaned Column
  documents_df['documents_cleaned']=documents_df.documents.apply(lambda x: " ".join(re.sub(r'[^a-zA-Z]',' ',w).lower() for w in x.split() if re.sub(r'[^a-zA-Z]',' ',w).lower() not in STOP_WORDS) )
  #Vectorize Documents using SKlearns TFIDF vectorizer
  tfidfvectoriser=TfidfVectorizer()
  tfidfvectoriser.fit(documents_df.documents_cleaned)
  tfidf_vectors=tfidfvectoriser.transform(documents_df.documents_cleaned)
  #Calculate Cosine Similarty and Euclidean Distance for the bert and TFIDF models
  #TFIF Model
  pairwise_similarities=np.dot(tfidf_vectors,tfidf_vectors.T).toarray()
  pairwise_differences=euclidean_distances(tfidf_vectors)
  tfidf_cos = most_similar(0,pairwise_similarities,'Cosine Similarity',documents_df,'TF-IDF')
  tfidf_euc = most_similar(0,pairwise_differences,'Euclidean Distance',documents_df,'TF-IDF')
  #Bert Model
  document_embeddings = bert_model.encode(documents_df['documents_cleaned'])
  bert_pairwise_similarities=cosine_similarity(document_embeddings)
  bert_pairwise_differences=euclidean_distances(document_embeddings)
  bert_cos = most_similar(0,bert_pairwise_similarities,'Cosine Similarity',documents_df,'BERT')
  bert_euc = most_similar(0,bert_pairwise_differences,'Euclidean Distance',documents_df,'BERT')
  #Return a dictionary of the experiment results
  return {'bert_cos':bert_cos,'bert_euc':bert_euc,'tf_cos':tfidf_cos,'tf_euc':tfidf_euc}
```

Figure 3: Code displaying the usage of BERT model

```
Comparing JSON:  1484008894  Files:  2  Lang:  en  TO  1484328949 Files:  2  Lang:  en
Similar Documents:  BERT
Cosine Similarity : 0.7060989141464233
Similar Documents:  BERT
Euclidean Distance : 12.17457103729248
Similar Documents:  TF-IDF
Cosine Similarity : 0.03199781934412287
Similar Documents:  TF-IDF
Euclidean Distance : 1.3914037377094213
Cosine Similarity - Documents in single model:  1.0000000001342073 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.7887568771355752 Rating:  3
```

Figure 4: Comparison of BERT model output with other methods

id	bert_cos	bert_euc	tf_cos	tf_euc	lda_cos
1484084337_1484110209	0.719940066	11.62303066	0.058041964	1.372558222	0.754036894
1484396422_1483924666	0.768661916	10.65473461	0.077526118	1.358288543	0.708255546
1484698254_1483758694	0.890699387	7.303583145	0.30181321	1.181682521	0.950765341

Figure 5: BERT model experiment CSV output for analysis

4.5.3 Data Extraction

There are inconsistencies with the format and availability of news article elements. This requires a lot of experimentation with manual extraction utilizing beautiful soup, semi-manual extraction with Newspaper3k, and cli provided by the competition organizers. Currently we prioritize the organizers provided cli. Beyond extracting the article's text we must find relevant data elements such as title and meta-data tags. Articles are sourced from multiple different news sources covering multiple countries. This causes variation in format and article data elements. To experiment with different elements of the HTML and JSON files we created a list of elements that can be modified to include and exclude elements. Currently including title and metadata tags have provided the best results.

5 DISCUSSION AND RELATED WORK

Currently we are using cosine similarity to compare the top n topics extracted by the LDA method. Other methods would include Euclidean and Jaccard distance. Currently we are unable to compare our results to other teams, so we will implement these methods and compare the results to our cosine similarity score. Being that none of the articles are word for word, the Jaccard distance may not account for the differences in topics.

6 CONCLUSIONS AND OPEN PROBLEMS

6.0.1 Open Problems

A major issue right now is the availability of the evaluation data. The Organizers will release the evaluation data Dec 3, 2021. We may not be able to submit our results for evaluation and compare our results to other competitors.

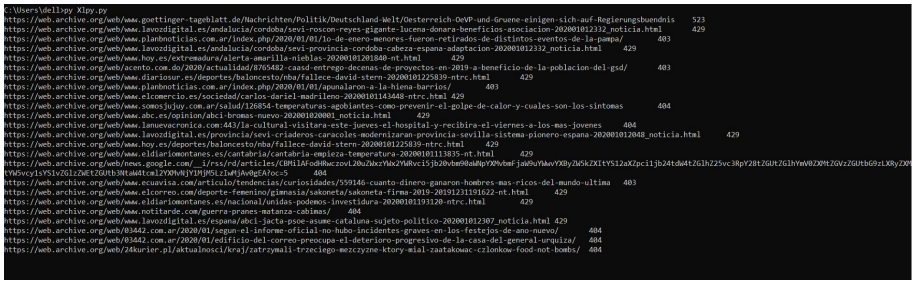

Figure 6: List of Unreachable links in training dataset

The current training data set contains articles that are no longer accessible. There is an open incident

The issue is still wish struggling to resolve the training data and looking to fix the websites that are unreachable. We are trying to eliminate the link pairs that are unavailable and going forward with the available links.

6.0.2 Conclusion

Combining topic extraction with cosine similarity to compare articles seems to be efficient. Currently we have a 80%-85% accuracy for finding similarity between article pairs. Once our results can be evaluated, will have the opportunity to tune our application.

Our data is available at:
`https://drive.google.com/drive/folders/1MqcXatfZ8rXSLR9IeEGIahYCNfAXKig6?usp=sharing` Our code is available at:
`https://colab.research.google.com/drive/16oQTKU8Snap\textbackslash_SbI1YSZRHOWZ4lQh19o0?usp=sharing`

7 CONTRIBUTIONS AND CONFLICT OF INTEREST DECLARATIONS.

This you'll fill only for the final report (based on the tables of contributions)

References

[1] Luling Huang. *Measuring Similarity Between Texts in Python*. URL: `https://sites.temple.edu/tudsc/2017/03/30/measuring-similarity-between-texts-in-python`. (09.22.2021).

[2] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 1994.

[3] Peipei Xia, Li Zhang, and Fanzhang Li. *Learning similarity with cosine similarity ensemble*. URL: `https://www.sciencedirect.com/science/article/pii/S0020025515001243?casa%5C_token=roHZ2606QUsAAAAA:YtUpL8gKQj-Ss%5C_rFqgDGx-shL0cvvsqFM4LGzbWNjqF4iAI1TAHA8flj1J9o%5C_P6u9n3RTZnPwuI`. (accessed: 09.22.2021).

8 APPENDIX 1

Where you put details allowing 'anyone skilled in the art' to reproduce your results – you can used as many appendices for it as you wish.

9 Appendix 2

10 Appendix 3

....
e.g. you can post the printout of your notebooks here – no limit on the number of pages in the appendices.

Codalab Multilingual Similarity Progress Summary: September 27, 2021—October 5, 2021

The overall objective of this project is:
Finding the best way to withdraw and extract the topics from articles of all languages. Includes taking the stop words from being categorized. We needed to address the articles in different languages by translating the language of the article and categorizing the topics.

The objective for the last week was:
Build an algorithm to generate a more accurate threshold for cosine similarity scores

The objective for the next week is:
Build an algorithm to generate a more accurate threshold for cosine similarity scores the problems: Some articles in the training data are not available and the links are broken. The competition organizers are working on resolving an issue with submitting project reports.

The problems we are facing are:
Implementing an neural network algorithm

We plan to address them as follows:
Learn more about Neural Network Implementation and concepts behind it.
We need help with:

Suggestions on improving our application by implementing an neural network algorithm

Codalab Multilingual Similarity Progress Summary: October 12, 2021—October 19, 2021

The overall objective of this project is:
Document similarity determines how similar a given set of documents are. In this project we are planning to do a multilingual document level similarity task to news from several domains. The data required is obtained from the contest main page. The training data is released in batches throughout the contest's preparation period. Our approach for this task is to use a pairwise rating on a 4-point scale. And, do a step-by-step process from extraction of data to analyzing each article. Libraries like gensim, NLTK, spaCy and Beautifulsoup are used in this project. The final aim of this project is to develop a system that can identify multilingual news articles that give similar information.

The objective for the last week was:
Build an algorithm to generate a more accurate threshold for cosine similarity scores the problems: Some articles in the training data are not available and the links are broken. The competition organizers are working on resolving an issue with submitting project reports.

The objective for the next week is:
Use the perplexity measure to fine tune the number of topics in the LDA models. Compare a sample of randomly generated scores to the predicted similarity score from cosine similarity.

The problems we are facing are:
Currently we have been able to produce a solution but we are unable to compare our results to other competitors. There are several

We plan to address them as follows:
Continue to work along side the organizers to find ways to evaluate our results and manually testing article pairs.

We need help with:
Finding the appropriate way to evaluate our results.

Date	Measure you use	Val of the measure	Method	Num of Exp	Comments
Current date 10-19-2021			Cosine-Similarity	Number of Documents processed	Very reliable results identifying similar documents, and testing scoring scale (1-4)
Previous report date 10-12-2021			TF-IDF	Number of Documents processed	TF-IDF provided unreliable similarity results due to noise.
Your Baseline date 09-13-2021			tokenization,POS-tagging		
External report date					

Table 1: Progress Summary

Codalab Multilingual Similarity Progress Summary: October 19, 2021—October 26, 2021

The overall objective of this project is:
Document similarity determines how similar a given set of documents are. In this project we are planning to do a multilingual document level similarity task to news from several domains. The data required is obtained from the contest main page. The training data is released in batches throughout the contest's preparation period. Our approach for this task is to use a pairwise rating on a 4-point scale. And, do a step-by-step process from extraction of data to analyzing each article. Libraries like gensim, NLTK, Spacy and Beautifulsoup are used in this project. The final aim of this project is to develop a system that can identify multilingual news articles that give similar information.

The objective for the last week was:
Validate the results that we produced utilizing a manual QA testing method and comparison to additional models and methods.

```
def bert_experiment(jfile1,jfile2):
  #Extract the main text from the Articles
  text1 = json_text(jfile1[0][0],jfile1[0][1])
  text2 = json_text(jfile2[0][0],jfile2[0][1])
  #Create the Corpus containing the text from both articles
  corpus = [text1,text2]
  #Convert the corpus to a data frame with two rows and single column 'documents'
  documents_df=pd.DataFrame(corpus,columns=['documents'])
  #Lowercase all text, remove empty space, and remove stop words. Place the clean text in the documents_cleaned Column
  documents_df['documents_cleaned']=documents_df.documents.apply(lambda x: " ".join(re.sub(r'[^a-zA-Z]',' ',w).lower() for w in x.split() if re.sub(r'[^a-zA-Z]',' ',w).lower() not in STOP_WORDS) )
  #vectorize Documents using SKLearns TFIDF vectorizer
  tfidfvectoriser=TfidfVectorizer()
  tfidfvectoriser.fit(documents_df.documents_cleaned)
  tfidf_vectors=tfidfvectoriser.transform(documents_df.documents_cleaned)
  #Calculate Cosine Similarty and Euclidean Distance for the bert and TFIDF models
  #TFIF Model
  pairwise_similarities=np.dot(tfidf_vectors,tfidf_vectors.T).toarray()
  pairwise_differences=euclidean_distances(tfidf_vectors)
  tfidf_cos = most_similar(0,pairwise_similarities,'Cosine Similarity',documents_df,'TF-IDF')
  tfidf_euc = most_similar(0,pairwise_differences,'Euclidean Distance',documents_df,'TF-IDF')
  #Bert Model
  document_embeddings = bert_model.encode(documents_df['documents_cleaned'])
  bert_pairwise_similarities=cosine_similarity(document_embeddings)
  bert_pairwise_differences=euclidean_distances(document_embeddings)
  bert_cos = most_similar(0,bert_pairwise_similarities,'Cosine Similarity',documents_df,'BERT')
  bert_euc = most_similar(0,bert_pairwise_differences,'Euclidean Distance',documents_df,'BERT')
  #Return a dictionary of the experiment results
  return {'bert_cos':bert_cos,'bert_euc':bert_euc,'tf_cos':tfidf_cos,'tf_euc':tfidf_euc}
```

Figure 7: Code displaying the usage of BERT model

```
Comparing JSON:  1484008894  Files:  2  Lang:  en  TO  1484328949 Files:  2  Lang:  en
Similar Documents:  BERT
Cosine Similarity : 0.7060989141464233
Similar Documents:  BERT
Euclidean Distance : 12.17457103729248
Similar Documents:  TF-IDF
Cosine Similarity : 0.03199781934412287
Similar Documents:  TF-IDF
Euclidean Distance : 1.3914037377094213
Cosine Similarity - Documents in single model:  1.0000000001342073 Rating:  4
Cosine Similarity - Seperate LDA Models:  0.7887568771355752 Rating:  3
.........................
```

Figure 8: Comparison of BERT model output with other methods

The objective for the next week is:

id	bert_cos	bert_euc	tf_cos	tf_euc	lda_cos
1484084337_1484110209	0.719940066	11.62303066	0.058041964	1.372558222	0.754036894
1484396422_1483924666	0.768661916	10.65473461	0.077526118	1.358288543	0.708255546
1484698254_1483758694	0.890699387	7.303583145	0.30181321	1.181682521	0.950765341

Figure 9: BERT model experiment CSV output for analysis

Complete manual QA testing. Analyse testing results and use the results to modify comparison approach. Improve the readability of our Colab notebook to aid in the hand-off.

```
[ ]  1 tester_index = 1
     2 num_testers = 4
     3 num_bins = sample_size / num_testers
     4 testers = []
     5 for i in range(1,sample_size+1):
     6   testers.append(tester_index)
     7   if(i%num_bins) == 0:
     8     tester_index += 1
     9
    10 sample_data = sample_data.assign(tester_id=testers)
    11 sample_data.head()
    12 sample_data.tail()
    13
```

	url1_lang	url2_lang	lang1	lang2	pair_id	link1	
311	en	en	en	en	1484765381_1543550230	https://news.google.com/__i/rss/rd/articles/CB...	https://www.capitaleconomics.com/pu
1601	en	en	en	en	1484189098_1483804274	https://www.haaretz.com/israel-news/but-ima-we...	https://sputniknews.com/analysis/20200
449	en	en	en	en	1484189457_1483848158	https://www.seacoastonline.com/news/20191230/y...	https://minnesota.cbslocal.com/2019/1
1592	en	en	en	en	1484032698_1483907371	http://www.iran-daily.com/News/263803.html	https://thefrontierpost.com/iran-chi
912	en	en	en	en	1484454170_1484311455	https://timesofindia.indiatimes.com/city/thiru...	https://www.siasat.com/govt-brings-o

Figure 10: Randomly samples 200 records and assigns them to QA tester

The problems we are facing are:
Currently we have been able to produce a solution but we are unable to compare our results to other competitors.

We plan to address them as follows:
Currently we are working along side the organizers to find ways to evaluate our results and manually testing article pairs. In t he interim we are comparing our Gensim LDA cosine comparison solution against Bert and SKLearn using Cosine and Euclidean distance algorithms. We are also, manually QA testing by taking a random sample of the data and splitting the testing duties amongst the team.

We need help with:
Finding the appropriate way to evaluate our results.

Date	Measure you use	Val of the measure	Method	Num of Exp	Comments
Current date 10-26-2021			BERT Model for computing similarity	Number of documents processed	Results are as reliable as Cosine Similarity method.
Previous report date 10-19-2021			Cosine-Similarity	Number of Documents processed	Very reliable results identifying similar documents, and testing scoring scale (1-4).
Your Baseline date 09-13-2021			tokenization,POS-tagging		
External report date					

Table 2: Progress Summary

Project Summary of Individual contributions.

https://www.overleaf.com/project/61586c8fc353a408cb9334e9

	Shreyas Lokesha	Mohammed Sharik U Zama	Travis Weston	Bhuvana Sree Garikapati	Rebekah Yermakovich
Week1(Sep 13th - Sep.20th)	Analysis of Problem Task	Went through thorough documentation of problem statement and related papers	Worked on article extraction and cli configuration	Research on document comparison methods	Research on document comparison methods
Week2(Sep 21st - Sep.27th)	Developed program for pair Link elimination on trial data Input data set	Experimented with trial input data. Performed Data extraction and analysis. Reviewed the shared Google Colab code notebook	Built Spacy nlp model	Data extraction and analysis	Data extraction and analysis
Week3(Sep 28th - Oct 4th)	Developed code for randomized score during first submission	Helped with presentation and documentation of the project. Looked into the testing and validation of built models	Worked on translating documents to english and scoring threshold	Similarity results testing and validation	Similarity results testing and validation
Week4(Oct 5th - Oct 12th)	Testing data extraction elements	LaTex Report Formatting	TF-IDF experiment and comparison	TF-IDF experiment and comparison, Article translation testing	CLI testing and integration
Week5(Oct 12th - Oct 19th)	Pair Link elimination clean up on training data, preparing weekly report	Preparing LaTex report and formatting	Experimenting with perplexity to tune the number of topics extracted for the LDA model, preparing LaTex report	Analyzing Sentence Transformer model to apply BERT architecture	Analyzing Sentence Transformer model to apply BERT architecture, preparing weekly report
Week6(Oct 19th - Oct 26th)	Analysis of the BERT model and Project report formatting in Latex	Validate the results that were produced utilizing a manual QA testing method	Develop Bert Model to compare two articles, and export the data to a dataframe and csv for analysis	Comparison to additional models and methods	

Table 3: Team Contributions