Multilingual News Similarity

Chirag Ahuja
Gatech
cahuja8@gatech.edu,

Megha Sharma
Gatech
msharma98@gatech.edu,

Kiran Potnuru
Gatech
kpotnuru7@gatech.edu,

Chinmay Ajnadkar
Gatech
cajnadkar3@gatech.edu

Abstract

Nowadays news is distributed globally due to linguistic barriers. Significant events are reported by different sources and in different languages which leads to lack of unified view of global events. We try to address this problem with our system which assesses similarity between news articles in a multilingual setting. We present two parallel stream of experimentation - 1) adapting pre-trained language representation model BERT[2] to news domain and 2) multi-task learning which utilizes similarity scores of dimensions like geography, time and entity to determine overall similarity score. Our experiments can be reproduced from our Github repository [1].

1. Introduction

Following developing news stories is imperative for general awareness and making decisions based on important global events. Due to abundance of online media providers, evolution of internet and thus increased reached of media, the news is being reported in different languages. Multilingualism of news makes it very difficult to cluster articles reporting the same event. This hinders spread of knowledge and development of holistic understanding of the event. Most large organisations employ full-time personnel to search the media for relevant information. For example, political parties are interested in the media's opinions expressed about them or about certain subjects. Similarly organisations in medical or finance sector monitor reports about the outbreak of diseases and stock exchange-relevant transactions. Currently this job is done completely or partially manually by analysts who often encounter the problem of multilingualism of articles. The current approach is not robust and will not scale with continuous increase in amount of online data.

We contribute to this field by participating in ”Multilingual News Article Similarity” shared task at SemEval 2022 workshop[2]. This task is part of larger research project on Agenda setting in the European Union [3].

We propose a multilingual news similarity system which rates the similarity of a pair of articles on a scale of 1 to 4. We train it for 6 languages (English, French, Spanish, German, Polish, Turkish and Arabic) but its underlying architecture allows it to be extended to other languages even in low-resource settings. We conduct experiments along different dimensions which have been summarized below.

1. **BERT based classifiers and loss functions.** We use embeddings from pre-trained BERT and experiment with various Machine Learning and Deep Learning classification methods. We try to adapt the classifiers to our use-case of similarity prediction by experimenting with loss functions.

2. **Long-text classifiers.** We modify BERT based classifiers to handle long documents as BERT can only process inputs of length upto 512 tokens.

3. **Multi-task learning.** Besides overall similarity score, the dataset consists of sub-scores which evaluate article similarity based on geolocation, time, shared entities and shared narratives. We use these scores along with the overall similarity score to fine-tune pre-trained T5 [7] model.

We review related work from literature in Section 2 before explaining the details of our experiments and their results in Section 3 and 4. All experiments described in the paper can be reproduced using the source code available on GitHub[4]. We describe the challenges faced during the

[1]https://github.gatech.edu/cahuja8/CS7643_Multilinguals

[2]https://competitions.codalab.org/competitions/33835#learn_the_details-overview

[3]http://www.euagendas.org/

[4]https://github.gatech.edu/cahuja8/CS7643_Multilinguals

project in Section 7 and future directions worth exploring in 8.

2. Related Work

A lot of work has been done in the field of document similarity like mentioned in [11], which performs Semantic Textual Similarity (on resource rich and low resource language) by first learning a shared multilingual sentence encoder through translation task on large parallel corpus. Then textual semantic tasks are trained based on the pre-trained multilingual encoder with labeled data from both rich-resource language and low-resource language. This solution cannot be applied to our problem because we are interested in documents which talk about similar topics and are not semantic similarity. Another popular approach is Multilingual Document Clustering [6] which involves dividing a set of n documents, written in different languages, into specific k number of clusters, so that the document most similar to other document will be in the same cluster. This is not a preferred solution in our use case as we might not have a similar article at all in corpus. We want to compare similarity of 2 articles, instead of clustering multiple of them. A similar approach is described in [5]. Pre-trained Bidirectional Encoder Representations from Transformer (BERT) has also been used as mentioned in [10]. More recent and evolving approach is Multitask Learning in NLP. Several papers from recent years show that combining multiple NLP tasks can generate better and deeper representation of the text. For example, identifying entities in a sentence, such as names of locations or people, can help with finding mentions of them in subsequent sentences. A paper from 2018 [9] showed competitive performance in a multilingual classification task in a limited resource scenario.

3. Methodology

We pose this problem as a multi-class classification with 4 classes - 1 to 4 (both inclusive). The labels indicate the similarity of articles with 4 being highly dissimilar. In this section we describe data, preprocessing techniques and approach taken for experimentation.

3.1. Data and Preprocessing

We use the dataset provided in the shared task [5]. It consists of 2939 pairs of news articles in 6 languages which are English(en), German(de), Spanish(es), Polish(pl), Turkish(tr), Arabic(ar), and French(fr). Each pair of article is given similarity scores based on overall similarity and similarity based on geolocation, time, shared entities and shared narratives. Scores are given in range of 1-4 where 1 being highly similar. Table 1 shows sample row from the dataset. These scores have been calculated by averaging several annotators' scores or by taking one annotators' score if only one is available. We round off these averaged scores to the

[5]https://competitions.codalab.org/competitions/33835#learn_the_details-timetable

Column name	Value
Article1	catholic academy of waterbury to reopen on tuesday, jan. 7 following water main break
Article2	osakis senior meals are served at the community center. call (320) 859-2325 for reservations. wednesday, january 1 – closed for new year's day. thursday, january 2 – closed. friday, january 3 – cook's choice. monday, january 6 – chicken strips, mashed potatoes and gravy, fruit salad. tuesday, january 7 – chicken alfredo, salad, brownie. wednesday, january 8 – pork roast, mashed potatoes and gravy, creamed peas, bread pudding.
Overall score	4.0
Geolocation score	3.67
Entity score	3.33
Time score	2.33
Narrative score	4.0
Style score	3.33

Table 1. Example from dataset

Language	Train	Valid	Test
en	1190	298	165
de	290	73	40
es	280	70	39
tr	34	8	5
pl	128	32	18
fr	18	5	2
ar	32	8	5

Table 2. Language wise distribution of dataset

Metric	Train	Valid	Test
min. no. of samples in a class	377	95	45
max. no. of samples in a class	661	177	90
avg. no. of samples in a class	493	123.5	68.5
min no. of tokens in article1	7	12	5
max no. of tokens in article1	10455	4711	2717
avg no. of tokens in article1	483.3	532.2	464.67
min no. of tokens in article2	7	16	9
max no. of tokens in article2	5073	4733	1618
avg no. of tokens in article2	373.18	417.59	350.78

Table 3. Dataset statistics

nearest smallest integers. Dataset consists of URLs of news articles. We use python script provided by the shared task organizers to download the earliest version of the article. We preprocess these articles by lowercasing the text and removing special characters.

We divide the data into train, test and validation set in 7.2:1.8:1 ratio. We use stratified sampling based on language of articles for splitting so that proportion of languages in each dataset remain uniform which can be observed in table 2. Table 3 describes the statistics of train, validation and test dataset. We observe length of articles is highly uneven and data consists of very long articles. This poses a challenge in using existing pre-trained models like BERT. We discuss these challenges in further sub-sections.

3.2. Single-task learning

3.2.1 Pre-trained BERT

BERT is a transformer model which learns text representations by jointly conditioning on both left and right context. It has been trained on Masked Language Modeling (MLM) and Next Sentence Prediction (NSP) tasks. We use *bert-base-multilingual-uncased* variant from Huggingface [6] which consists of 12 layers, generates word embeddings of size 768 and has 110M parameters.

3.2.2 Classification task

We concatenate both the articles using [SEP] token and use pre-trained BERT for generating joint embeddings of an article pair. Embedding of [CLS] token is extracted and considered as joint article embedding. Since BERT accepts maximum of 512 tokens as input, we limit our article length to x words each where x is a hyperparameter we tune and words are determined by splitting the article on space token. We use these embeddings for training the classification head. We experiment with 3 different classifiers:

1. **Cosine similarity.** It is a measure of similarity between two or more vectors quantified by the cosine of the angle between vectors. The vectors are typically non-zero and within an inner product space. For the classification task, we generate vectors using pre-trained BERT and convert cosine similarity into linear scale to get 4 classes.

2. **SVC.** Support vector machines is a supervised learning algorithm used for classification (SVC) and regression problems (SVR). We use SVC which finds a hyperplane separating features into different domains. We create vectors from the sentences in text using pre-trained BERT and then take difference between vectors in a pair, which are classified into 4 classes using SVC.

3. **Fully connected feedforward network(FFN)** We create a feedforward network of 1 layer which takes embedding of [CLS] token as input and generates logits of 4 classes. Final probabilities are obtained by adding softmax layer on top of the logits. Cross-entropy and Supervised Contrastive Learning is used as loss functions for training the network.

3.2.3 Loss function

In FFN described in section 3.2.2 cross entropy is used as the loss function. In this experiment we use weighted average of Supervised Contrastive Loss (SCL) and CE for training FFN as shown in 1 where λ is scalar weighting hyperparameter. This approach is inspired by [3, 4, 8]. Using this loss we try to achieve better classification results by keeping embedding vectors of news article pairs with same label tightly close while distinctly pushing those far off with different labels.

$$L = (1 - \lambda)L_{CE} + \lambda L_{SCL} \qquad (1)$$

3.2.4 Long-text classifiers

As seen in 3.2.2 sub-section we truncate articles to x words due to BERT's maximum input length of 512. In order to alleviate this limitation, we build 2 classifiers from scratch which use BERT for generating embeddings of segments of articles instead of complete ones. Segments of article is created by bucketizing article into groups of x words (where x is a hyperparameter). We maintain an overlap window between segments in order to avoid loss of context. In both classifiers we implement Siamese architecture where we create 2 identical models and share the weights between them. Both the articles are passed through separate models.

1. **Siamese FFN.** We generate BERT embeddings of segments of an article, extract [CLS] embedding of each segment and take their mean. The mean embedding of both articles is concatenated and passed through a dropout layer and then a trainable linear layer. Relu is applied on its output and the resulting tensor is passed through another linear layer which generates logits as output.

2. **Siamese LSTM network.** We generate embeddings of segment of article like in Siamese feed-forward network. Instead of taking mean of segment embeddings, we pass it through series of unidirectional LSTM layers. Hidden state of the last cell of the last LSTM layer is taken as the article embedding. The concatenated hidden states are passed through the same architecture as described in feed-forward network.

[6]https://huggingface.co/

In both the classifiers probabilities are generated by applying softmax on the logits. Cross entropy has been used as a loss function.

3.3. Multi-task learning

3.3.1 Background

In Machine Learning, we typically care about optimizing for a particular metric, whether it is a score on a certain benchmark or a business KPI. In order to do this we generally train a single model or an ensemble of models to perform our desired task. We then fine-tune and tweak these models until their performance no longer increases. While we can generally achieve acceptable performance this way, by being laser-focused on our single task, we ignore information that might help us do even better on the metric we care about. Specifically, this information comes from the training signals of related tasks. By sharing representations between related tasks, we can enable our model to generalize better on our original task.

3.3.2 Pre-trained T5 and MT5

T5 is a Transformer based architecture that poses every problem as a text-to-text task. Every task including translation, question answering, and classification is cast as feeding the model text as input and training it to generate some target text. This allows us to use the same model, loss function and hyper parameters across diverse set of tasks. To help the model distinguish among tasks, a task name is appended at the beginning of the input. T5 model is trained only for English whereas MT5 is its multilingual variant. We use t5-base (for English articles) and google/mt5-small (for multilingual setting) in our experiments.

3.3.3 Classification task

Our dataset consists of 6 scores: geo-location similarity score, time similarity score, similar entities score, style similarity score, narratives similarity score and overall similarity score for each pair of article. Each score is in range of 1-4, hence we formulate our problem as multi-class classification task. We train our T5 using all the 6 scores. We prefix each article with [SentenceX:] token and concatenate them. For each task we add task name prefix to concatenated sentence pair and train it using score of that particular task. For example, If we want to train for geolocation similarity score, we formulate our input as ”Geo: Sentence1:Article1 Sentence2: Article2 ” and output label as geolocation similarity score. To limit the training time we use only used 220 words from each article.

Metric	SVM	Cosine Similarity	FFN + CE	FFN + SCL + CE
Val. Precision	0.30	0.34	0.58	**0.59**
Val. Recall	0.29	0.19	0.58	**0.61**
Val. F1-score	0.29	0.09	0.58	**0.60**
Test Precision	0.34	0.27	0.63	**0.64**
Test Recall	0.30	0.20	0.65	**0.66**
Test F1-score	0.31	0.12	0.63	**0.64**

Table 4. Results of BERT-based classifiers

4. Experimentation Setup and Results

In all experiments, we use weighted averaged Precision, weighted averaged Recall and weighted averaged F-Score as evaluation criteria. In this section, we describe the experimentation setup and results of all experiments.

4.1. Single-task learning

4.1.1 Classification

We train 3 classification heads which use embeddings generated from BERT as joint representation of the articles. We tune the hyperparameters of SVM and cosine-based classifier using grid search. We get best results on validation set using hard margin SVM for SVC. The value of c = 0.001 produce the best results compared to higher values of c. For cosine similarity, no hyperparameter tuning is needed as we just take the inner product to find the angle between 2 vectors. Since we use single layer in feed-forward network, we didn’t perform any tuning on that as well.

Table 4 describes the metrics obtained using best configuration of these models on validation and test set. The results are inline with the representational power of the model. Given the same input FFN performs the best.

4.1.2 Loss Function

We use dataset with training, validation and test splits as mentioned in section 3.1

To experiment with loss functions, we use open source SBERT[7] sentence pair classification cross-encoder model[8]. We initialize the cross-encoder model with bert-base-multilingual-uncased variant from Huggingface as mentioned in 3.2.1. We modify SBERT library to include enhanced model performance reports and train with supervised contrastive loss function.

We experiment with CE loss and SCL+CE loss with 2 different truncated article lengths = $\{35, 220\}$. We fine tune with batch size = 4, Adam optimizer with learning rate =

[7]https://www.sbert.net

[8]https://www.sbert.net/examples/applications/cross-encoder/README.html

$2e-5$ and WarmupLinear scheduler. We keep 10% of training data for warm up and perform early stopping at epochs = 3.

For SCL term, we use and extend SupCL-Seq method[9] to generate positive pairs by data augmentation. We use dropout masks with dropouts = $[0.0, 0.05, 0.2]$. This generates 3 distinct sentence embeddings for each input anchor for positive pairing. Negative pairs are generated with other labels in the batch. We perform grid-search based hyperparameter tuning with $\lambda = \{0.8, 0.9\}$ and temperature $\tau = \{0.1, 0.3\}$.

We find the SCL+CE loss is very effective in overall training for hyperparameters with words per article = 220,$\lambda = 0.9$ and temperature $\tau = 0.1$.

Loss function results are shown in Table 4 as FFN+CE column for cross-entropy loss and FFN+SCL+CE column for SCL+CE loss.

4.1.3 Long-text classifiers

We train Siamese FFN and Siamese LSTM network from scratch in order to handle long articles. We segment the articles into segments of 220 tokens each. We perform hyperparameter tuning using Grid Search algorithm whose details are given in Appendix 1. Best results were obtained for Siamese FFN using learning rate=0.001, batch size=64, hidden size=512 and dropout=0.3 and that for Siamese LSTM were obtained using learning rate=0.001, batch size=4, dropout=0.4, number of LSTM layers=2, size of LSTM hidden state=128 and size of linear hidden state=64. Both models are trained for 300 epochs and early stopping is enforced if difference between average validation and average training loss is more than 0.1. It prevents overfitting of data on small dataset as we can see in Figure 4.1.3.

Table 6 summarizes the results of both the models. We observe that Siamese LSTM out performs Siamese FFN in all metrics but lacks behind the BERT-based classifier with SCL described in section 4.1.2. In these models both articles are passed separately whereas BERT takes both articles simultaneously and applies cross-attention on them. We speculate the absence of cross-attention between articles as major reason of lacking performance compared to BERT-based classifiers.

4.2. Multi task training

In multi task learning we use 2 pre-trained models t5-base (for English language) and google/mt5-small (for multilingual setting)

We train T5 only on English articles and use Overall Similarity score. After that we extend the same model by considering all 6 scores for training as explained in Section 3.3.3. At last we train MT5 on multiple languages and all 6 scores. Table 6 describes the results obtained by these experiments. We observe that it performs worst than BERT based classifiers even after given more information possibly due to difference in pre-training regimes of these architectures. Due to time constraints we weren't able to do further analysis on it.

[9]https://github.com/hooman650/SupCL-Seq

Metric	Siamese FFN	Siamese LSTM
Val. Precision	0.38	0.41
Val. Recall	0.41	0.39
Val. F1-score	0.37	0.4
Test Precision	0.34	0.43
Test Recall	0.39	0.42
Test F1-score	0.36	0.42

Table 5. Results of long-text classifiers

Metric	T5	T5 Multi Task	MT5 Multi Task
Test Precision	0.43	0.22	0.17
Test Recall	0.50	0.35	0.26
Test F1-score	0.44	0.26	0.21

Table 6. Results of Multi Task Learning

5. Results Discussion

We compare the models quantitatively using weighted average Precision, Recall and F1-score. In single task training we get best F1 score of 0.64 on test using BERT-based classifier with CE and SCL loss function. We observe that classifiers trained using classical Machine Learning techniques perform the worst due to their low representational power. Even though Siamese LSTM and Siamese FFN receive non-truncated text as input, they don't perform better than BERT-based classifiers. This is because BERT acts as a cross-encoder when pair of sentences are passed as input. In multi-task learning, we fine-tune T5 and MT5 on our dataset but weren't able to do further experimentation and analysis due to time constraints.

Further we analyse the performance of our best model (BERT-based classifier with CE and SCL loss function) qualitatively. Details of its qualitative analysis have been added in Appendix 3.

6. Implementation Details

We implement our experiments with Pytorch[10] and HuggingFace[11]. We extend open source transformer libraries like SimpleTransformer[12] and SentenceTransformers[13]. We refer to these Git repositories for our experiments:

[10]https://pytorch.org/
[11]https://huggingface.co/
[12]https://simpletransformers.ai/
[13]https://www.sbert.net/

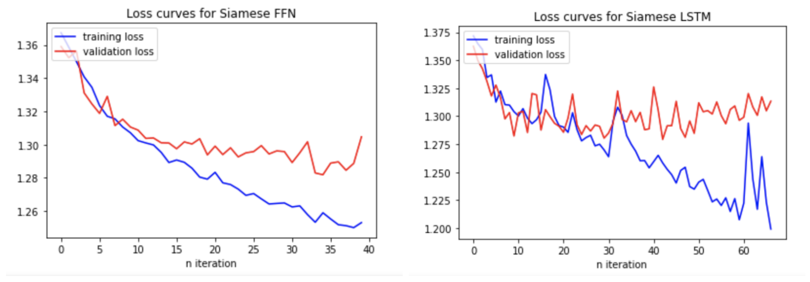

Figure 1. Training and Validation loss curves of long-text classifiers

1. Extended Cross Encoder and Cross-entropy Evaluator classes and created custom Evaluator for SCL loss [14]

2. Extended SupCL-Seq Method[15] for data augmentation using dropout mask

3. Extended SCL to compute loss per batch [16]

4. Took reference for implementing Siamese networks [17]

7. Challenges and Learning

7.1. Initial plan

Our initial idea was to pre-train long-text model like Longformers[1] on multilingual data as it's currently trained only on English data. Post pre-training, we would've trained multiple classifier heads using the article embedding from Longformers. Each head would've trained on different similarity score like overall, geolocation, entity and time similarity score. Collectively these train would've fine-tuned last layers of pre-trained Longformers. At inference we would've used classification head trained on overall score.

7.2. Challenges

We had to diverge from our plan as we had access to a 16GB GPU and pre-training required more compute. Another challenge was less volume of task specific fine-tuning data. Only 2939 samples were provided in the dataset. This often posed the problem of overfitting in large pre-trained models and underfitting in models trained from scratch. We use mechanisms like dropout and early stopping to prevent overfitting. Besides multilingualism of data hampered us from doing thorough manual analysis of the trained models.

[14]https://github.com/UKPLab/sentence-transformers
[15]https://github.com/hooman650/SupCL-Seq/blob/main/src/SupCsTrainer.py
[16]https://github.com/HobbitLong/SupContrast/blob/master/losses.py
[17]https://github.com/AndriyMulyar/bert_document_classification

7.3. Learning

Irrespective of all the challenges and constraints this project proved to be a good learning experience as we learnt about the lifecyle of model development which included infrastructure setup, project scoping, literature survey, data preprocessing, model training and analysis. We learnt about concepts like Siamese architectures, multi-task learning and contrastive losses. While experimenting with Supervised Contrastive loss, we went through the code-base of SBERT library and extended it for our use-case. Though our Siamese models didn't outperform BERT-based classifiers but learnt a lot while implementing and tuning these models from scratch. Overall we feel we've succeeded in this project.

8. Future Work and Conclusion

In this paper we present our experiments for multilingual news similarity along 4 dimensions - classifiers based on pre-trained BERT, classifier with custom loss function, classifier handling long-text and multi-task classification. We get best F1 score of 0.64 using classifier using embeddings from pre-trained BERT and SCL loss function. In future work we will explore multi-task learning on top of embeddings from long-text models like Longformers. We plan to pre-train Longformers on news article in a semi-supervised setting. We will also explore data augmentation techniques to further train our Siamese models using pseudo labels from strong models like BERT.

References

[1] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020. 6

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[3] Beliz Gunel, Jingfei Du, Alexis Conneau, and Ves Stoyanov. Supervised contrastive learning for pre-trained language model fine-tuning, 2021. 3

[4] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning, 2021. 3

[5] Sebastião Miranda, Artūrs Znotiņš, Shay B. Cohen, and Guntis Barzdins. Multilingual clustering of streaming news. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4535–4544, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. 2

[6] Soto Montalvo, Raquel Martínez, Arantza Casillas, and Víctor Fresno. Bilingual news clustering using named entities and fuzzy similarity. In Václav Matousek and Pavel Mautner, editors, *Text, Speech and Dialogue, 10th International Conference, TSD 2007, Pilsen, Czech Republic, September 3-7, 2007, Proceedings*, volume 4629 of *Lecture Notes in Computer Science*, pages 107–114. Springer, 2007. 2

[7] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer, 2020. 1

[8] Hooman Sedghamiz, Shivam Raval, Enrico Santus, Tuka Alhanai, and Mohammad Ghassemi. Supcl-seq: Supervised contrastive learning for downstream optimized sequence representations, 2021. 3

[9] Karan Singla, Dogan Can, and Shrikanth Narayanan. A multi-task approach to learning multilingual representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 214–220, Melbourne, Australia, July 2018. Association for Computational Linguistics. 2

[10] Shruthi Srinarasi, Reshma Ram, Sanjay Raghavendra, Annapurna P Patil, S Rajarajeswari, Manjunath Belgod, Rituraj Kabra, and Abhishek Singh. A combination of enhanced wordnet and bert for semantic textual similarity. In *2021 2nd International Conference on Control, Robotics and Intelligent System*, CCRIS'21, page 191–198, New York, NY, USA, 2021. Association for Computing Machinery. 2

[11] Xin Tang, Shanbo Cheng, Loc Do, Zhiyu Min, Feng Ji, Heng Yu, Ji Zhang, and Haiqin Chen. Improving multilingual semantic textual similarity with shared sentence encoder for low-resource languages, 2018. 2

Appendix 1: Work Division

Table 8 contains the detailed task breakdown for the project. Our team consists of 4 members, with tasks divided more or less equally among members.

Appendix 2: Hyper-parameter tuning of Long-text classifiers

Table 7 contains hyper-parameter ranges for Siamese FFN and LSTM used in tuning using Grid Search algorithm.

Appendix 3: Qualitative analysis of BERT-based classifier + CE + SCL

We take our classifier trained using CE and SCL and pass different types of articles through it in order to assess its predictive power. We found that we get high similarity if an article is compared with itself, its translation and its title. However, similarity score is low if an article is compared with another article having similarity element towards the end. This is expected as text is truncated before passing to BERT due to constraint on maximum length. Details of these experiments have been detailed in Table 9.

Hyperparameters	Range of values
Learning rate	[1e-4, 1e-3, 1e-2]
Batch size	[4, 8, 16]
Dropout	[0.2, 0.3, 0.4]
Hidden size of linear layer	[512, 256, 128, 64, 32]
LSTM layers	[1,2,3,4]
LSTM hidden-size	[512, 256, 128, 64]

Table 7. Hyperparameters for long-text classifiers

Task	Owner
Project idea and scoping	Megha
Literature Survey	Chirag
Project proposal	Megha
Dataset Download and Tokenization	Chinmay
Dataset Preprocessing	Megha
NER for multi-task learning	Megha
BERT-based classifiers	Chinmay
Comparison of CE vs SCL	Kiran
Multi-task learning using T5	Chirag
Handling of long text using Siamese LSTM and FFN	Megha
Model analysis	All
Final paper [Intro, Related Work]	Kiran
Final Paper [Abstract, Dataset]	Chirag
Final paper [Methodology, Experiments]	Megha
Final Paper [Results, Conclusion/Future Work]	Chinmay

Table 8. Work division

Experiment input	Predicted label
Identical articles	1
English article and its French translation	1
Title and its complete article	1
Articles having similarity element towards the end	3

Table 9. Sanity checks on best model