Assessing multilingual news article similarity

Student Name: Venkata Sumanth Nagabhairu **Student ID:** 20012395

Introduction:

In the real world, there will be news articles written in multiple languages. At any instance, news agencies tend to author articles on similar topics to reach their targeted audiences. The set of audience for different news agencies can collide because of geography or audience knowing multiple languages. In this project, we calculate the similarity between the content of the news articles which are written in different languages. We score the similarity of the articles between 1 - 4.

The multilingual news article similarity model has various applications. We will review two of them. Firstly, Machine translation. The model can be used to translate news articles from one language to another while preserving the content of the original article. Second, sentiment analysis. The model can be used to analyze the sentiment of news articles in different languages and provide insights into public opinion on several topics.

It is difficult to model multilingual news article similarity because different languages have different syntactic and semantic structures. Language contact [1] can be a problem as languages lack words and phrases. Then the model must borrow words from other languages. There are a few other difficulties which will not go through.

Problem Formulation:

Given the train and test datasets, containing the links to news articles of various languages (English, Spanish, Arabic...). The dataset itself does not have the article text. Download the text using the **semeval_8_2022_ia_downloader** **.** After downloading, assign a percentage of train data for validation. Develop a model that predicts the similarity score in range 1 to 4. The model can use techniques such as optimizers, regularizer etc. Implement Evalution metrics and calculate MAE of the overall score on the test set.

In this project, we implement the task as regression model. We take a pair of articles along with their titles as inputs. Encoding is performed with sentence transformer. The similarity score is the output of the model. The target column of the dataset is “overall” column.

Background:

Assessing the similarity between multilingual news articles is a challenging task that requires a deep understanding of natural language processing techniques, machine learning algorithms, and cross-lingual information retrieval. Previous work in this field has focused on developing methods to bridge the language gap and enable effective comparison and clustering of news articles in different languages. Here are some key areas of background work related to assessing multilingual news article similarity:

Multilingual Text Representation: Researchers have explored various techniques for representing multilingual text, such as bag-of-words models, term frequency-inverse document frequency (TF-

IDF) weighting, and distributed word embeddings (e.g., Word2Vec, GloVe). These representations capture the semantic and contextual information of the text and can be used as input features for similarity assessment.

Cross-lingual Information Retrieval (CLIR): Cross-lingual information retrieval techniques aim to retrieve relevant information in a different language based on a user's query. These methods leverage multilingual resources, parallel corpora, machine translation, or cross-lingual word embeddings to bridge the language gap and enable effective retrieval of similar documents in different languages. CLIR techniques can provide valuable insights and approaches for assessing the similarity between multilingual news articles.

Similarity Measures: To quantify the similarity between news articles, various similarity measures have been employed, such as cosine similarity, Jaccard similarity, Euclidean distance, and probabilistic measures like KL divergence. These measures consider the feature representations of articles and provide a numerical measure of their similarity.

Supervised and Unsupervised Learning Approaches: Researchers have explored both supervised and unsupervised learning approaches for assessing the similarity between multilingual news articles. Supervised methods utilize labeled datasets where similarity scores are provided for pairs of articles. These methods typically involve training machine learning models, such as support vector machines, random forests, or neural networks, to predict similarity based on input features. Unsupervised methods, on the other hand, leverage clustering or dimensionality reduction techniques to group similar articles based on their feature representations without relying on labeled data.

Evaluation Metrics: To evaluate the performance of similarity assessment models, various evaluation metrics are used, including precision, recall, F1 score, mean average precision (MAP), and normalized discounted cumulative gain (NDCG). These metrics help measure the effectiveness and accuracy of the models in capturing the similarity between multilingual news articles.

By studying and building upon the existing literature in these areas, the project can leverage the advancements and insights gained to develop an effective algorithm for assessing multilingual news article similarity. It is crucial to review and understand the strengths and limitations of previous approaches to build upon their successes and address their shortcomings, ultimately providing a more accurate and reliable similarity assessment solution.

Method:

Download dataset:

To download the text of the news articles links given in the dataset, we use semeval_8_2022_ia_downloader. We follow the below steps to download the given dataset into google collab /context/ folder,

Install semeval_8_2022_ia_downloader:

```
!pip install semeval_8_2022_ia_downloader
```

Scrape train dataset:

```
!python -m semeval_8_2022_ia_downloader.cli --links_file='/content/semeval-2022_task8_train-data_batch.csv' --dump_dir='/content/my_folder'
```

Scrape test dataset:

```
!python -m semeval_8_2022_ia_downloader.cli --links_file='/content/final_evaluation_data.csv' --dump_dir='/content/downloads_test_data’
```

Data Preprocessing:

We start by walking through the dataset and collecting the Json files. Then we extract title and text from the Json files. We concatenate and separate them with the keyword [SEP]. This helps the model understand the relation between title and text and can generate better embeddings. To keep code simple, we choose separator over passing tile and text as separate inputs. Title and text, along with data in rest of the columns of the dataset are added to the new file. For train and test, we create new files with name train.csv and test.csv respectively.

We then clean the data by removing punctuation, URL and numbers in the concatenated text. After that, we assigned 20 percent of the train data for validation.

Embeddings:

This creates embeddings for a sentence, not individual words. It generates vector of size 768. When we pass a sentence to the sentence transformer, it tokenizes the sentence and passes it to a pre-trained BERT. To get the sentence embeddings, pooling operation is applied on output of BERT [3]. We use ‘paraphrase-multilingual-mpnet-base-v2' version, which is a multi-lingual variant of sentence transformers. It was trained on corpus with 50+ languages.

Model:

The model is partially inspired from the article [4]. We used Siamese neural architecture to find the similarities between multilingual news articles. Fun fact, Siamese means twin. This architecture has two branches. Two articles' embeddings are passed through each of the branches.

Each branch consists of a dense layer. The hidden unit's size is 256. These layers are enforced with ReLU activation, l2 kernel_regularizer.

The output of these two branches is passed to dot layer, where the dot product to two vectors is calculated. The parameter normalized=True, ensures the normalization of the product. The output of this layer is then passed to a dense layer with linear activation. This outputs a single scalar value.

To train the model, we use Adam optimizer. The parameters, that we set for training purposes are the following,

Number of epochs = 100

Learning rate = 0.001

Size of the batch = 32

In the experimentation and results section, we will observe the results of different optimizers and epoch settings. We will also observe the performance by adding additional layers, along with batch normalization and early stopping

Please find the model summary below,

```
Model: "model_15"
__________________________________________________________________________________________________
Layer (type)                Output Shape         Param #     Connected to
==================================================================================================
input_25 (InputLayer)       [(None, 768)]        0           []

input_26 (InputLayer)       [(None, 768)]        0           []

dense_101 (Dense)           (None, 256)          196864      ['input_25[0][0]']

dense_102 (Dense)           (None, 256)          196864      ['input_26[0][0]']

dot_15 (Dot)                (None, 1)            0           ['dense_101[0][0]',
                                                              'dense_102[0][0]']
```

dense_103 (Dense) (None, 1) 2 ['dot_15[0][0]']

==

Total params: 393,730

Trainable params: 393,730

Non-trainable params: 0

____

Loss Function:

We designed the assessing multilingual news article similarity as regression model. So, we can use Mean Squared Error, Mean Absolute Error, Huber loss. The architecture of the model is Siamese. So, we could use contrastive loss.

In this project, we have used Mean Squared Error as the primary loss function. In experimentation and results, we could find results for contrastive loss as well.

As evaluation metric, we used Mean absolute error.

Training:

The model has Siamese neural architecture. To build the model, we used TensorFlow framework. We have a data set that contains text from multiple languages. For training we use 4965 pairs of news articles. The data is preprocessed before moving to the next steps. We train the model by converting the strings of article title and text separated with [SEP] into sentence embeddings, using sentence transformer. This will create 768 sized vectors. We assigned 20 percent of train data for validation.

We train the network in such a way that it predicts the similarity score with minimal loss. Our target column in the data set is 'Overall'. The number of epochs used for this model is 100. For every step of epoch, we evaluate on validation data. This will help the model to generalize for unseen data. We use Adam optimizer to adjust weights during back propagation. The loss function we use is mean squared error, as it's a regression model.

The output of the mode will be a scalar value between 1-4. If we get a floating-point number, we will round it off to the nearest number. The evaluation metric used in this project meant absolute error. We will observe the loss and MAE value for various configurations in the experimentation and results section.

Inference:

For inference, we use 4903 pairs of news articles. This is preprocessed before encoded into sentences embedding. This is done by a paraphrase-multilingual-mpnet-base-v2, which is a version of sentence transformers. We then use the model that was saved in h5 format during the training phase, to predict on test examples. The result is evaluated by Mean Absolute Error. The MAE we obtained was 1.11

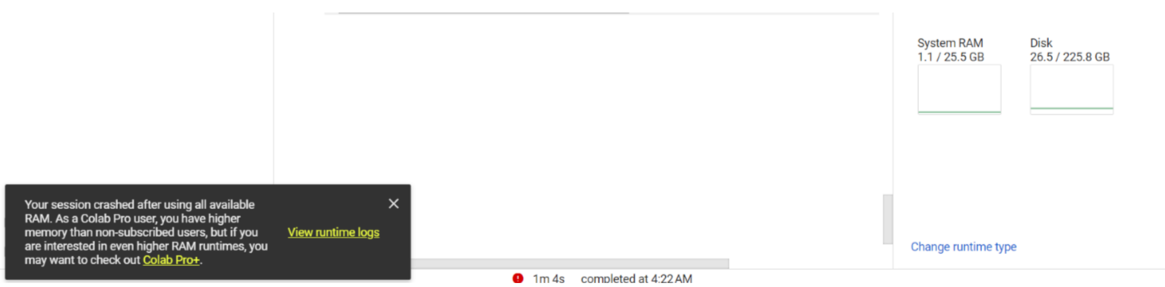
Experimentation and Results:

Embeddings:

To embed the tokenized data, we chose TFRobertaModel. This is XML-RoBERTa model written in TensorFlow. The embedding sizes that are produced by TFRobertaModel is 768.

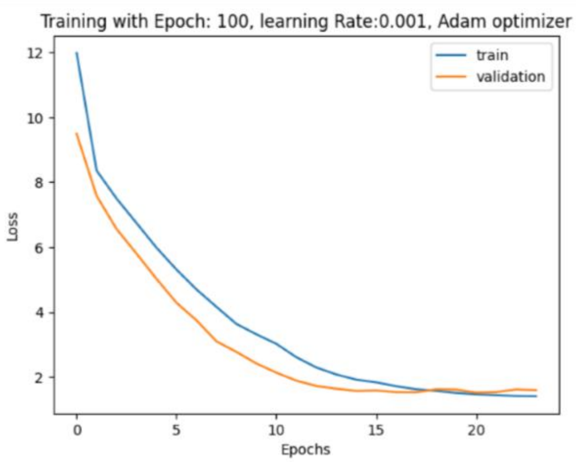
The reason we chose pre-trained models is because they give state-of-the-art performance as they were trained on large corpus. Also, they tend to capture better contextual information and meaning, while creating embedding vectors.

While this is a fantastic model, it requires a lot of computational resources. I have used 25 GB of ram, but still the Collab instance crashed.

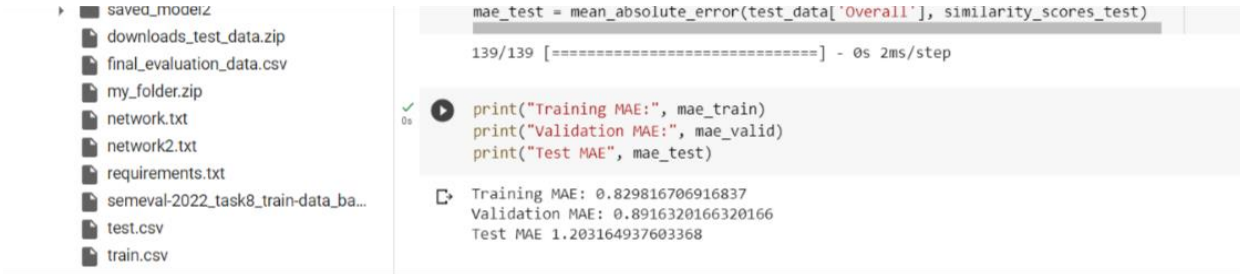


Additional Layers:

We evaluate the model's performance by adding additional layers. To prevent overfitting, we use early stopping and dropout of 0.4. We enforce the layers with batch normalization. We set its parameters momentum 0.99 and epsilon 0.001. The model is saved as saved_model2.h5. The model summary is saved in network1.txt.




```
121/121 [==============================] - 0s 2ms/step
31/31 [==============================] - 0s 2ms/step
```



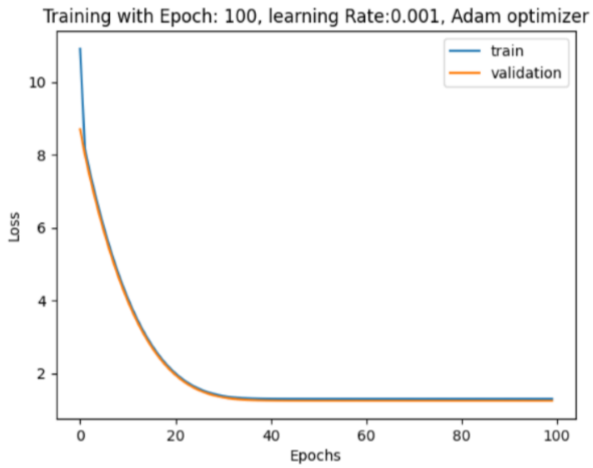
Different epoch sizes and optimizers:

The two optimizers we chose were Adam and SGD. These two optimizers have different approaches when updating parameters.

Adam is an adaptive learning optimizing algorithm. It maintains a learning rate for each parameter and adapts the learning rate based on the running average of the first and second moments of the gradients.

While SGD updates the weights of a neural network based on the gradient of the loss function with respect to the weights. SGD updates the weights after each batch of training data

Number of epochs – 100, optimizer Adam

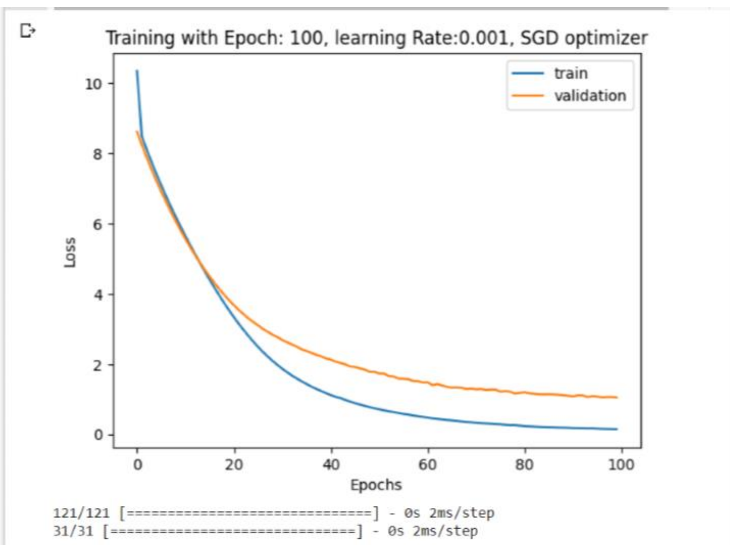



```
121/121 [==============================] - 0s 2ms/step
31/31 [==============================] - 0s 2ms/step
```

```python
print("Training MAE:", mae_train)
print("Validation MAE:", mae_valid)
print("Test MAE", mae_test)
```

```
Training MAE: 0.9803690630998824
Validation MAE: 0.9437456687456686
Test MAE 1.1108479927830401
```

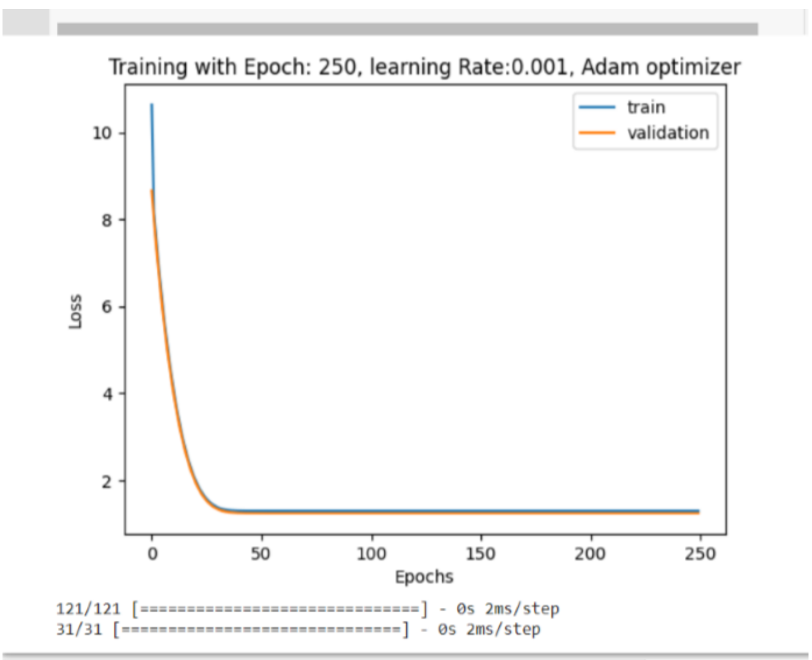
Number of epochs – 100, optimizer SGD




```
[34] print("Training MAE:", mae_train)
     print("Validation MAE:", mae_valid)
     print("Test MAE", mae_test)

     Training MAE: 0.08038330546783082
     Validation MAE: 0.7938496188496189
     Test MAE 0.8607352277852954
```

Number of epochs – 250, optimizer Adam

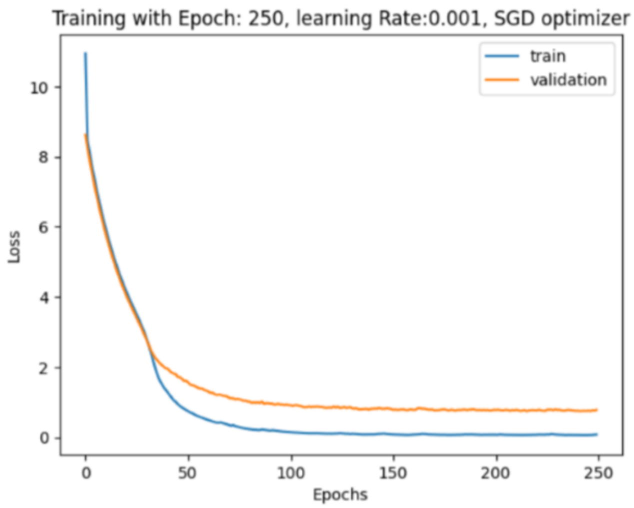


```
139/139 [==============================] - 0s
```

```
[41] print("Training MAE:", mae_train)
     print("Validation MAE:", mae_valid)
     print("Test MAE", mae_test)
```

```
Training MAE: 0.9803690630998824
Validation MAE: 0.9437456687456686
Test MAE 1.1108479927830401
```

Number of epochs – 250, optimizer SGD

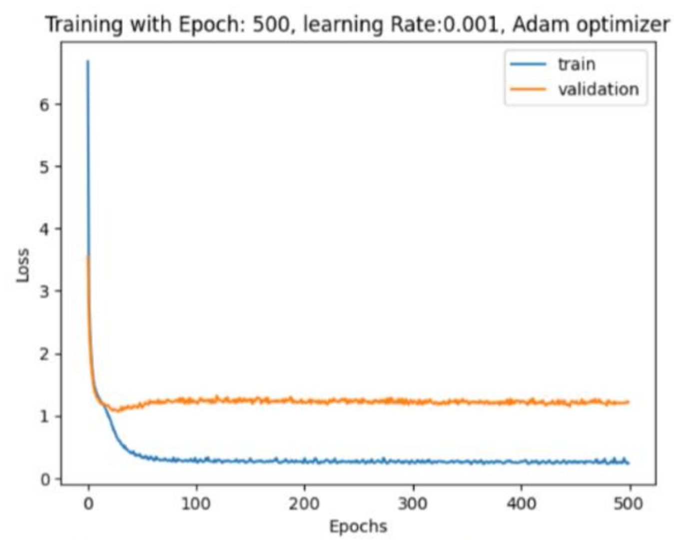



```
121/121 [==============================] - 0s 2ms/step
31/31 [==============================] - 0s 2ms/step
```

```python
print("Training MAE:", mae_train)
print("Validation MAE:", mae_valid)
print("Test MAE", mae_test)
```

```
Training MAE: 0.09446405350176482
Validation MAE: 0.6618329868329869
Test MAE 0.8421665914900016
```

Number of epochs – 500, optimizer Adam

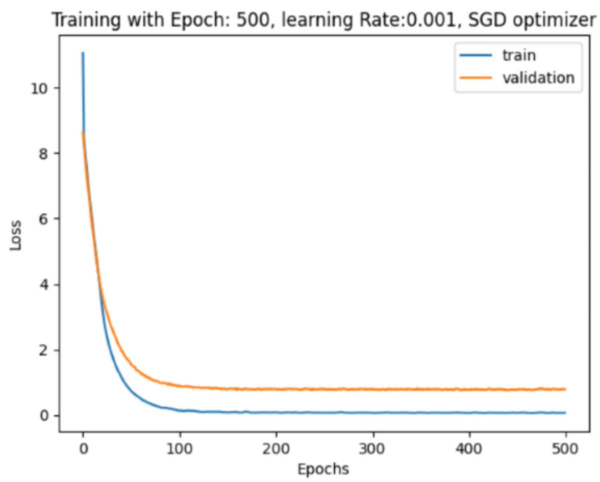



```
121/121 [==============================] - 0s 2ms/step
31/31 [==============================] - 0s 2ms/step
```

```
[63] print("Training MAE:", mae_train)
     print("Validation MAE:", mae_valid)
     print("Test MAE", mae_test)
```

```
Training MAE: 0.16049105207752803
Validation MAE: 0.813981288981289
Test MAE 0.8964441437377837
```

Number of epochs – 500, optimizer SGD




```
121/121 [==============================] - 0s 2ms/step
31/31 [==============================] - 0s 2ms/step
```

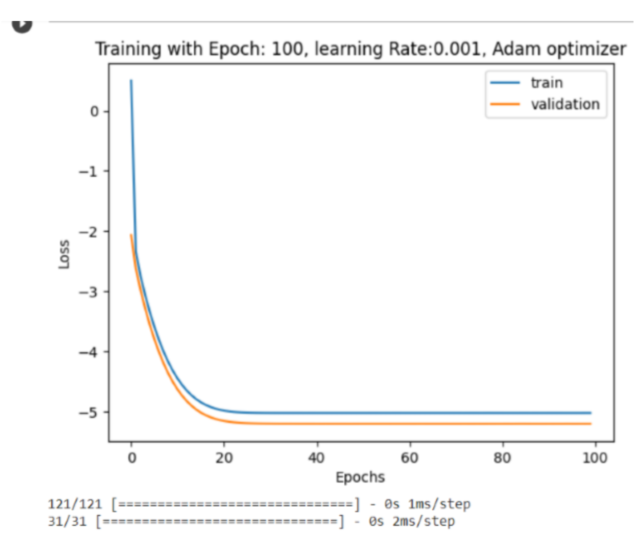
```python
print("Training MAE:", mae_train)
print("Validation MAE:", mae_valid)
print("Test MAE", mae_test)
```

```
Training MAE: 0.08140194439284165
Validation MAE: 0.6623873873873874
Test MAE 0.8455495414223426
```

Different loss functions:

Contrastive loss:

Constrastive loss is used for learning similarity between pairs.



```
[90] print("Training MAE:", mae_train)
     print("Validation MAE:", mae_valid)
     print("Test MAE", mae_test)

     Training MAE: 1.7962604495634404
     Validation MAE: 1.8352564102564102
     Test MAE 1.4198240866035183
```

Future Work:

Enhanced Multilingual Embeddings: Improve the representation of multilingual text by exploring advanced embedding techniques such as cross-lingual word embeddings, transformer-based models (e.g., multilingual BERT), or language-agnostic embeddings. These models capture semantic similarities across languages, enabling more accurate comparisons between news articles in different languages.

Cross-Lingual Information Retrieval Techniques: Investigate and incorporate state-of-the-art cross-lingual information retrieval techniques to enhance the retrieval and matching process. These techniques leverage parallel corpora, machine translation, or cross-lingual word alignment to bridge the language gap and improve the cross-lingual similarity assessment.

Supervised Learning for Similarity Assessment: Explore supervised learning approaches to train a similarity model using labeled data. Develop a dataset with pairs of news articles in multiple languages and their similarity labels, allowing the model to learn from explicit similarity judgments. Techniques such as siamese networks or triplet loss can be employed to train a neural network-based similarity model.

Unsupervised Learning for Similarity Assessment: Investigate unsupervised learning methods, such as clustering or dimensionality reduction techniques, to group similar news articles together without relying on explicit similarity labels. These approaches can identify hidden patterns and similarities in the data and facilitate the organization and retrieval of multilingual news articles.

Evaluation Metrics: Develop appropriate evaluation metrics for assessing the performance of the multilingual news article similarity model. These metrics should consider the unique challenges posed by multilingual data, such as language differences, varying article lengths, and cultural nuances. Evaluation can be performed using existing multilingual news datasets or by creating a new benchmark dataset specifically designed for assessing cross-lingual similarity.

Example Scenario: To illustrate the future work, let's consider a scenario where we aim to build a multilingual news article similarity model for English, French, and Spanish news articles. The model will be trained on a large corpus of news articles from different sources and domains. We will explore advanced multilingual embedding techniques, such as multilingual BERT, to capture the semantic similarities across languages. Additionally, we will incorporate cross-lingual information retrieval techniques, leveraging parallel corpora and machine translation, to improve the cross-lingual matching process.

To assess the model's performance, we will collect a labeled dataset of news article pairs in English, French, and Spanish, along with their similarity labels. This dataset will serve as the training data for supervised learning approaches, such as Siamese networks, to train a similarity model. We will also explore unsupervised learning techniques, such as clustering, to group similar articles together without using explicit similarity labels.

Finally, we will evaluate the model using appropriate evaluation metrics, considering the challenges posed by multilingual data. This evaluation will provide insights into the model's effectiveness in capturing cross-lingual news article similarity. The future work aims to advance the state-of-the-art in multilingual news article similarity assessment and contribute to various applications such as multilingual news recommendation, cross-lingual information retrieval, and content analysis across languages.

Conclusion:

Assessing multilingual news similarity can be modelled with simple Siamese neural architecture. This is because sentence transformers are trained on huge amount of corpus. They can encode context and meaning of text. Observing the results of experimentation, the model performs better with 100 epochs, Adam optimizer and Mean Squared Error (MSE) loss. There is room to experiment with different architectures and improve performance. The performance can also be improved with hyper parameter tuning methods, different loss functions and optimizers. NLP is an experimental field, there is no end for options. The model we developed can help the users to cut down the duplication of news in their daily life.

References:

1. Yin, W., Schütze, H., Xiang, B., & Zhou, B. (2016). ABCNN: Attention-based convolutional neural network for modeling sentence pairs. arXiv preprint arXiv:1512.05193. Link
2. Reimers, N., & Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) (pp. 3973-3983). Link
3. Cer, D., Yang, Y., Kong, S., Hua, N., Limtiaco, N., John, R. S., ... & Kurzweil, R. (2018). Universal Sentence Encoder. arXiv preprint arXiv:1803.11175. Link