ITNLP2022 at SemEval-2022 Task 8: Pre-trained Model with Data Augmentation and Voting for Multilingual News Similarity

Zhongan Chen[*], **Weiwei Chen**[*], **Yunlong Sun**[*], **Hongqing Xu**[*], **Shuzhe Zhou**[*], **Bohan Chen** [*],
Chengjie Sun, **Yuanchao Liu**
Intelligence Technology and Natural Language Processing Lab,
School of Computer Science and Technology,
Harbin Institute of Technology
{zachen, wwchen, ylsun, hqxu, szzhou, bhchen}@insun.hit.edu.cn
{cjsun, lyc}@insun.hit.edu.cn

Abstract

This article introduces a system to solve the SemEval 2022 Task 8: Multilingual News Article Similarity. The task focuses on the consistency of events reported in two news articles. The system consists of a pre-trained model(e.g., INFOXLM and XLM-RoBERTa) to extract multilingual news features, following fully-connected networks to measure the similarity. In addition, data augmentation and Ten Folds Voting are used to enhance the model. Our final submitted model is an ensemble of three base models, with a Pearson value of 0.784 on the test dataset.

1 Introduction

The task (Chen et al., 2022) aims to design a system that can find the similarities of multi-language news articles. The task focuses on the consistency of actual events reported in two news articles, but not the subjective factors such as writing style or political factors. Geographical location, time, common entity, and common narrative were used to judge similarity. Consistency is measured by a float value between [1,4]. The lower the value is, the more likely the two news are reporting the same thing. The metric is the Pearson Coefficient of the predicted and ground truth values.

Challenge of this task: 1. Need to understand every aspect of a news event: what happened, where and when, who was involved, and why and how it happened 2. Make the writing style and common phrases clear because some unnecessary content can be misleading. 3. Some information about the event is hard to get, such as the time, location, and description of the event.

We use the pre-trained language model XLM-Roberta and INFOXLM to fine-tune. Specifically, we splice the available information of two news articles (exact method will be described in Chapter 3), input it into the pre-trained model, and then transmit the output vector to the fully connected layer of downstream tasks. For the original fine-tuning task, we try to use various techniques to make the program run faster and work better. The techniques include 1. Freezing the lower layers of the pre-trained model, means not updating their parameters during training. 2. Data Augmentation. Use translation software to translate the original news text to expand the data. 3. Divide the training set into 10 parts. the ten folds voting was adopted to make full use of the data set.

Our code is available on github[1].

2 Related Work

2.1 Background

The input of the task is the content of two news, and the output is the Overall label of them. Overall label is a float value between 1 and 4, which is used to measure whether two news report a same thing.The lower the Overall score, the more likely the content of the two news to be the same. The datasets including nearly 5,000 pairs of news with Overall label given. In addition to the Overall label, Geography, Entities, Time, Narrative, Style, and Tone label are also noted in the datasets. The news is given in the form of links and contains seven languages(en, de, ar, es, fr, pl, tr), while the test set contains the other three languages(it, ru, zh). It should be noted that the dataset contains news pairs in different languages.

2.2 Pre-trained language model

Pre-trained language models such as BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) start to make a difference in the way of word representations rather than static word embedding methods, and Word2Vec (Mikolov et al., 2013) and FastText

[1]*These authors contributed equally.

[1]https://github.com/SemevalITNLP/Semeval8NewsCorrelation

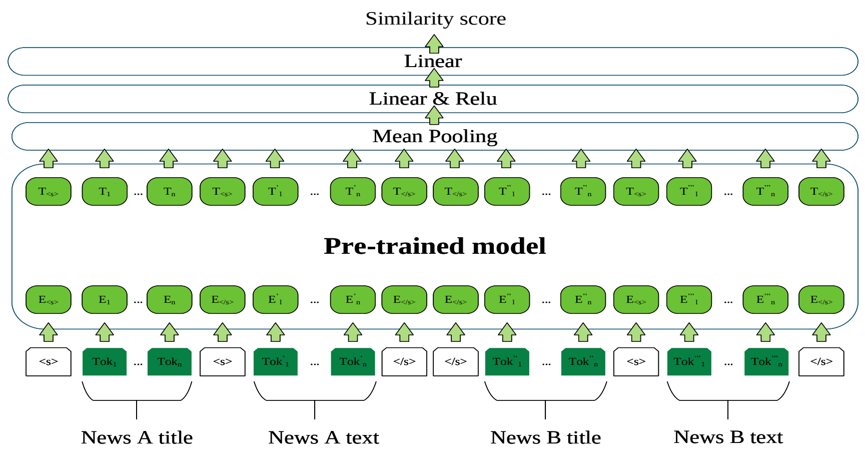

Figure 1: The architecture of the system.

(Joulin et al., 2016) are two examples. In particular, the XLM-RoBERTa (Conneau et al., 2019) model is a newly released large-scale cross-lingual language model based on RoBERTa and trained on 2.5TB filtered CommonCrawl data in 100 languages. Unlike other XLM models (Lample and Conneau, 2019), XLM-RoBERTa does not require language tokens to understand which language is used and can recognize the correct language from the input id.

INFOXLM (Chi et al., 2020) is a cross-lingual pre-trained model based on the XLM-RoBERTa structure, using monolingual and parallel corpora to train the model. Specifically, in addition to the masked language modeling(MLM) and the translating language modeling tasks(TLM), INFOXLM is jointly pre-trained with a newly introduced cross-lingual contrastive learning task. Through comparison, the cross-lingual uses bilingual pairs as the two views of the same meaning, making their encoded representations more similar than the negative examples. It uses the [CLS] tokens from the BERT encoder as sentence representations with linear projection heads. The momentum encoder is used to encode the query, while the online encoder is updated with InfoNCE (Van den Oord et al., 2018) loss.

2.3 Sentence similarity

There are usually two methods for comparing the similarity between two sentences. Cross-encoders perform full-attention over the input pair, and Bi-encoders map each input independently to a dense vector space.(Thakur et al., 2020)

3 System Description

3.1 Data processing

We use crawlers and the newspaper3 [1] library to download and parse news web pages to obtain various information about news, including title, text, pictures, keywords and abstracts. For the news data in the training and test set, some missing titles and body texts. At the same time, some web pages have errors during the crawling process, or the crawler crawls wrong news information. We revisit the news link and the alternate link to modify the news data. In addition, there are still problems such as advertising webpages and link failures, which are omitted in the training process.

3.2 Model

We designed cross-encoder model to solve the task. The structure of the cross-encoder model shows as Fig 1. It contains the pre-trained language model, pooling layer, and downstream layers, consisting of two fully connected layers and a relu activation function, to learn the Overall value of each news pair.

The inputs of cross-encoder model are composed of contents of two news which are named news A and news B. For each news, we use title, text and keywords. We use the symbol <s> to separate different parts of the news, the symbol </s> as the separator for two news articles, and add <s> and </s> at the beginning of each news pair. The input form finally can be illustrated as: <s> news A </s></s> news B </s>.

[1] https://github.com/codelucas/newspaper

For the pooling methods, we compared two methods of learning the representation of the whole news pair: 1. use the output of the model Pooler, which is similar to the token's representation of the [CLS] of BERT; 2.mean pooling: use the average vector of the whole hidden state. Experiments have shown that the latter works better.

The model eventually outputs a value between 1 and 4, which calculate MSE loss (Mean Squared Error Loss) with the standard Overall label of the data.

3.3 Methods

- **Ten-Fold Voting** Ten-Fold Voting method shuffles and divides the training set into ten parts equally, and separately chooses one of them as a validation set, while the remaining nine part are used as the training set. As a result, we end up with 10 models. These ten models use averaging method to vote, which increases model generalization. For example, we use 10 models to obtain 10 Overall predictions for a given news pair, and the average of these 10 results is taken as the final prediction for that news pair.

- **Data Augmentation** Due to the addition of three new languages (zh, ru, it) in the test set, we added translation corpus to the training data so that the model can better deal with the new languages. Specifically, we use Baidu translation API [1] to obtain the translated text, and the news in each language is translated into the other 9 languages. Different epochs are trained alternately with the original and translated data. Each news is translated into one of the nine other languages with the same probability at the epoch using translated data.

- **Frozen Layers** Generally, lower layers of a language model encode more local syntax while higher layers capture more complex semantics (Tenney et al., 2019). Therefore, during training, we freeze the parameters of the embedding layer and some lower layers of the pre-trained model. The parameters of frozen layers cannot participate in back propagation thus keeping the original parameters. Therefore, we aim to choose the most memory-efficient hyperparameters.

- **Multi-task Learning** We noticed that the training data not only has the Overall label but also has other labels. Considering that Narrative and Entity are consistent with the target task, we let the model fit the Overall label and the Narrative label and Entity label. Specifically, after the pre-trained model, we separately use a dense layer to obtain different predicted values for Entity, Narrative, and Overall and calculate the MSE loss of the three labels: $loss_{entity}$, $loss_{narrative}$ and $loss_{Overall}$. Then the multi-task loss of the model, $loss_{multi-task}$, take the weighted sum of the three.

- **Auxiliary Loss** We add a loss function to help the model distinguish those news pairs. In addition to calculating the similarity of each news pair, we also calculate the MSE loss between the predicted Overall labels of the two news pairs and their true Overall labels, which is as following.

$$loss_{AL} = mse(\hat{y_1} - \hat{y_2}, y_1 - y_2) \qquad (1)$$

  where y_1 is the standard Overall value of news pair 1 and $\hat{y_1}$ is the predicted Overall value of news pair 1, so as y_2 and $\hat{y_2}$. Two news pairs are randomly selected. In practice, we randomly draw several news pairs from a batch and combine them. The training data was shuffled before the start of each training epoch so that the data within the batch are not precisely the same in different epochs.

4 Experiment

All experiments were run on two GPUs: NVIDIA GeForce RTX 3080 Ti and NVIDIA GeForce RTX 3090. For optimizer selection, we use Adamw optimizer with weight decay taken as 0.01 and set 1e-5 as learning rate for the pre-trained layer and 1e-4 for downstream layers with the batchsize 8.If not specified, we use **INFOXLM-large** as the pre-trained model from Hugging Face[2].The weight of the $loss_{entity}$, $loss_{narrative}$ and $loss_{Overall}$ in the $loss_{multitask}$ are set to 0.3, 0.7 and 1.0 respectively.The weight of the $loss_{AL}$ is set to 0.1.

4.1 Information Selection

This section presents experiments to explore which news information should be used. From Table 1

[1] http://api.fanyi.baidu.com/

[2] https://huggingface.co/models

,it can be seen that the result of using only title and only text is not as good as using title and text at the same time. The model shows better performance when facing more text types, which may be because different texts contain different information. The title focuses on summarizing the news and represents the article's central idea; The text focuses on describing the event and conveying more detailed information. The performance of using title, keywords and text is not improved. Most of the keywords information is likely contained in the first two.

Model	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$
Title60	0.798	0.684
Text100	0.813	0.702
Title60+Text100	0.852	0.759
Title60+Text100+Keywords50	0.855	0.760
Title60+Text150	0.859	**0.772**
Title60+Text200	**0.859**	0.769

Table 1: The impact of using different information and different content lengths on the model. Pearson coefficients in the table are the maximum values for the validation set, notated $\mathcal{D}_{val}$ in 15 epochs, while $\mathcal{D}_{test}$ is the corresponding pearson coefficients in test set at this time.The number after the title and text in the table indicates how many text words are used. For example, Title60 indicates using the first 60 words of the title; The ratio of validation set to training set is 1:9.

The text length of each news article is related to the final prediction of the system. The table shows the Overall scores of different text lengths. Due to the input length limitations, our max length of the title is 60, while the max length of the text is between 100 and 200. The model works better with longer text lengths in the validation set, probably because longer texts contain richer information, while it achieves the best result in the test set when text length is 150. The reason may be that the behavior of the test set is inconsistent with the validation set.

4.2 Frozen layers

We tested the performance of unfrozen, frozen embedding layers and partial transformer layers respectively. The results are shown in Table 2. As we can see, when freezing embedding layers and the layers from 0 to 11, the result is similar to not freezing all layers, but it saves memory and reduces training time. So we take these kinds of parameters in the following experiments.

Frozen layers	$\mathcal{D}_{test}$
no freezing	0.760
freeze embedding + layers 0∼5	0.759
freeze embedding + layers 0∼11	0.759
freeze embedding + layers 0∼14	0.750

Table 2: Results of freezing different pre-trained model layers on the test set.

4.3 Strategies

We also conducted additional experiments, including data augmentation through translation, a multi-task learning approach, and Auxiliary Loss mentioned before. These methods are effective in the validation set, but some did not improve much in the final test set.

Table 3 shows the comparison of different methods used in the model. The parameters used by the base model are INFOXLM$_{\text{large}}$, Title60 and Text100. The split ratio between the training set and the validation set is 9:1. According to the table, we have the following conclusions:

In the validation set, when the model cumulatively uses Multi-task Learning, Auxiliary Loss and Data Augmentation, the performance is continuously improved. When the three methods are used together, the model performance is best to reach 0.8627 in the validation set. Multi-task Learning and Auxiliary Loss increase the learning and representation ability of the model by modifying the task and loss. Data Augmentation improves the generalization of the model by introducing the translation corpus.

In the test set, when the model uses Multi-task Learning and Auxiliary Loss, the performance is not much different from that of the base model. This may be that Multi-task Learning and Auxiliary Loss can improve the representation ability of the model in the validation set, but they are lack of generalization and generally perform in the face of a large amount of data and new data. However, when the model uses Data Augmentation, the performance reaches 0.7667 in the test set, which is greatly improved compared with the base model and the model using Multi-task Learning and Auxiliary Loss. This may be because the translation corpus introduces languages (Chinese, Russian and Italian) that the model has not encountered before. The newly introduced language enhances the model's generalization ability in the test set and enables the model to understand the news in the test set better.

Model	$\mathcal{D}_{val}$	$\mathcal{D}_{test}$
Base	0.852	0.759
Base + MT	0.854	0.758
Base + MT + AL	0.859	0.759
Base + MT + AL + DA	**0.862**	0.766
Base + MT + AL + DA + TFV	/	**0.781**

Table 3: Results of system under different methods. Base: $\text{INFOXLM}_{\text{large}}$ +Title60+Text100; MT:Multi-task Learning; AL:Auxiliary Loss; DA:Data Augmentation; TFV:Ten-fold voting

Finally, we can see that when the model uses Ten-Fold voting, the performance is improved from 0.7667 to 0.7810. There may be two reasons for this. On the one hand, the Ten-Fold voting essentially uses all the training data, and the expansion of the amount of data may increase the performance of the model. On the other hand, the Ten-Fold voting integrates the model results under ten different training data sets, which greatly improves the robustness of the results and strengthens its generalization ability.

4.4 Ensemble

The ensemble technique is a widely used strategy. Ensemble methods work by aggregating the predictions of multiple single models. The strategy we use in the competition is simple averaging. The final prediction value is obtained by averaging the prediction results of different models, which will improve the robustness of the prediction results. The results are shown in Table 4.

Model	$\mathcal{D}_{test}$
INFOXLM	0.776
INFOXLM + DA	0.781
XLM-RoBERTa + DA	0.779
Ensemble	**0.784**

Table 4: Results of base models and ensemble model.

All three base models use Title60, Text100, Multi-Task Learning and Auxiliary Loss. The latter two models additionally use the Data Augmentation strategy. The result of the ensemble model achieves the best performance in our competition.

5 Conclusion

In this paper, we summarize our work in Multilingual News Article Similarity. We utilize INFOXLM and XLM-RoBERTa pre-trained models to handle multilingual news. In addition, many methods are used such as Data Augmentation and Ten-Fold Voting. Our final submitted model is an ensemble of three base models, and we achieve Pearson value of 0.784 on the test dataset.

Acknowledgements

Thank the organizer for holding a great competition and providing the large multilingual news article similarity dataset. This research was supported in part by the National Natural Science Foundation of China under contract 62176074.

References

Xi Chen, Ali Zeynali, Chico Q. Camargo, Fabian Flöck, Devin Gaffney, Przemyslaw A. Grabowicz, Scott A. Hale, David Jurgens, and Mattia Samory. 2022. SemEval-2022 Task 8: Multilingual news article similarity. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*. Association for Computational Linguistics.

Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. 2020. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. *arXiv preprint arXiv:2007.07834*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *north american chapter of the association for computational linguistics*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical nlp pipeline. *arXiv preprint arXiv:1905.05950*.

Nandan Thakur, Nils Reimers, Johannes Daxenberger, and Iryna Gurevych. 2020. Augmented sbert: Data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. *arXiv preprint arXiv:2010.08240*.

Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807.