# Progress Report II - Peace Speech Project

Hongling Liu (hl3418), Haoyue Qi (hq2180),
Xuanhao Wu (xw2744), Yuxin Zhou (yz3904), Wenjie Zhu (wz2536)

November 27, 2021

## I. Progress Overview

In stage 1, we have developed some step-by-step methodologies to preprocess the data using a sample dataset, done some research on how BERT models work under the hood, and analyzed the scalability of BERT. In stage 2 of this project, we carried out these theoretical proposals on real data. As promised in our previous report, we will provide a more detailed exploratory data analysis (EDA) on the real data this time and briefly discuss the modifications we have made to the previous proposal based on the EDA outcome. Along with those updates, we will primarily focus on presenting the variety modeling approaches we have studied and the findings.

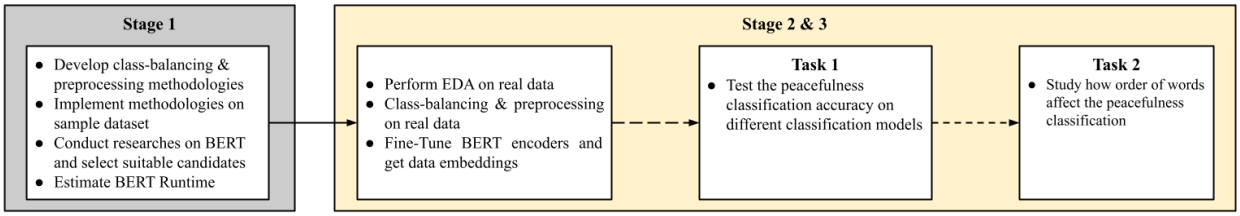


Figure 1-1 Progress Overview

## II. Exploratory Data Analysis (EDA) on Real Data

Recall from the previous report that the real data is stored on the AWS S3 bucket as many .json files in two separate directories: *highPeace* and *lowPeace*, with ~33M articles from top-10 high-peace countries and ~57M articles from top-10 low-peace countries under the *rule-of-5*[1] metric. Due to the limited computational resources, we proposed the Iterative EDA approach in Figure 2-1 in our previous report.

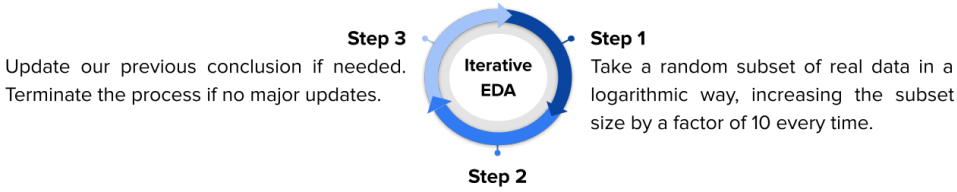


Figure 2-1: Iterative EDA Process

[1] For each country, scores for indices such as Global Peace Index, the Positive Peace Index, etc., are computed. Then countries are placed into low-, mid-, and high-peace categories for each index. If a country has at least 5 indices categorized as low-peace, and no index in high-peace, then the country is low-peace. Same logic follows for high-peace.

## II.I. Problems within the Data

All natural language samples in our dataset are published news articles and are collected and pre-cleaned by LexisNexis. Therefore, from a data-quality perspective, there are barely any typos and missing data in this dataset. From the data-engineering perspective, however, each data point is stored as an individual .json file with less than 20 KB per file on average. This storage format could cause a so-called *Small Files Problem* when loading the data, which means the milliseconds to open, read, and close each file could accumulate to a large runtime overhead if a program has to execute those steps in high frequency. As examined in the first report, loading 1,000 of these .json files into notebook memory takes 1 minute on average using a basic notebook instance (*ml.t2.medium*) on AWS SageMaker, which scales up to a total of 32 hours for loading 1M samples from each peace category. The existence of this problem adds difficulties for us to take a large enough subset as a representative of the entire data using the iterative EDA approach.

## II.II. The Solution to the Problems

To avoid encountering the *Small File Problem* repeatedly, we read in a large set of files from the real dataset at once and save them together as one large file for re-use in the rest of this project. To deal with the file fetching latency, we made several adjustments to our code: 1) use UltraJSON (*ujson*) package instead of the traditional json package for faster content loading; 2) parallelize the I/O bound loading task via multithreading; 3) switch to a larger notebook instance (*ml.m5.4xlarge*) with a wider network bandwidth, which unclogs the high-frequency file transferring traffic between S3 bucket and SageMaker, and a larger memory space, which allows a faster memory allocation for incoming data stream. The optimized program and the newly selected notebook instance took ~2.5 seconds to load 1K data points for each peace category, which is a ~24x speed up overall. In concern of the budget, we limited the amount of data we scanned here and studied in later steps to 1M per peace category (2M data points in total) and only kept the fields needed for lightweight file transfer and storage. But with a relaxed budget constraint, one can re-use the pipeline we present here to finish extracting and compressing the rest of the data files.

## II.III. EDA Result

Among multiple fields per data record, only the *title*, *content*, *wordCount*, and *country* are the most relevant to our topic. Having noticed that the article distribution across countries has been collected and analyzed before, we will focus more on the word count distribution in our EDA. Based on the sampled 2M data we obtained above, we summarised the sample article count distribution, sample average word count and the 95% confidence interval (C.I.) for the average word count per country in *Table 2-1*. This table further validates the data imbalance condition we concluded in the last report. In the non-peace country group, non-Indian data only makes up less than 10% of the category total, and we only have 1 data point from Guinea among the 1M non-peace samples. The peace country group is slightly more balanced, but the dominance of Australian data is still obvious. Luckily, we still have enough samples from the majority of countries to narrow down the width of the word count C.I. to ~30 words.

| country | sample_perc | avg_wordCount | wordCount_CI_95 |
|---|---|---|---|
| Afghanistan | 0.2328 | 221.809708 | (213.6184, 230.001) |
| Congo | 0.4985 | 489.747041 | (477.5319, 501.9622) |
| Guinea | 0.0001 | 3121.000000 | (nan, nan) |
| India | 91.7565 | 422.663411 | (421.0335, 424.2933) |
| Iran | 2.3680 | 484.964443 | (479.5853, 490.3436) |
| Kenya | 0.5157 | 607.522397 | (594.3597, 620.6851) |
| Nigeria | 2.7022 | 579.226482 | (571.1824, 587.2705) |
| Sri Lanka | 1.3659 | 831.227542 | (810.2969, 852.1582) |
| Uganda | 0.2902 | 575.023777 | (560.7757, 589.2719) |
| Zimbabwe | 0.2701 | 566.279156 | (551.8202, 580.7381) |

| country | sample_perc | avg_wordCount | wordCount_CI_95 |
|---|---|---|---|
| Australia | 76.0521 | 884.916874 | (882.2838, 887.55) |
| Austria | 0.4210 | 1240.742993 | (1200.2169, 1281.2691) |
| Belgium | 5.3665 | 703.747545 | (683.1177, 724.3774) |
| Czech Republic | 0.5935 | 415.039596 | (402.2787, 427.8005) |
| Denmark | 1.3845 | 1281.044854 | (1252.5792, 1309.5105) |
| Finland | 1.0927 | 1807.282694 | (1767.3396, 1847.2258) |
| Netherlands | 0.8204 | 1566.497075 | (1530.0766, 1602.9176) |
| New Zealand | 9.0610 | 526.680709 | (522.3605, 531.0009) |
| Norway | 1.2701 | 1851.772459 | (1815.7198, 1887.8251) |
| Sweden | 3.9382 | 1688.450307 | (1673.9758, 1702.9249) |

Table 2-1: Data Distribution Overview
(Left: Non-peace Countries, Right: Peace Countries)

We performed the iterative EDA step on the word count and plotted the boxplot for each peace group per iterative step in *Figure 2-2*. As we include more and more samples in our dataset, more large-value samples appears for both groups, but the quartile lines for the box are always stabilized at ~ (130, 260, 420) for the non-peace group, and ~ (240, 495, 1150) for the other group. The long and growing tail points out a possibility that the true distribution of word counts has an infinite variance, which makes sense in reality since there is no upper limit posed on article length in general.

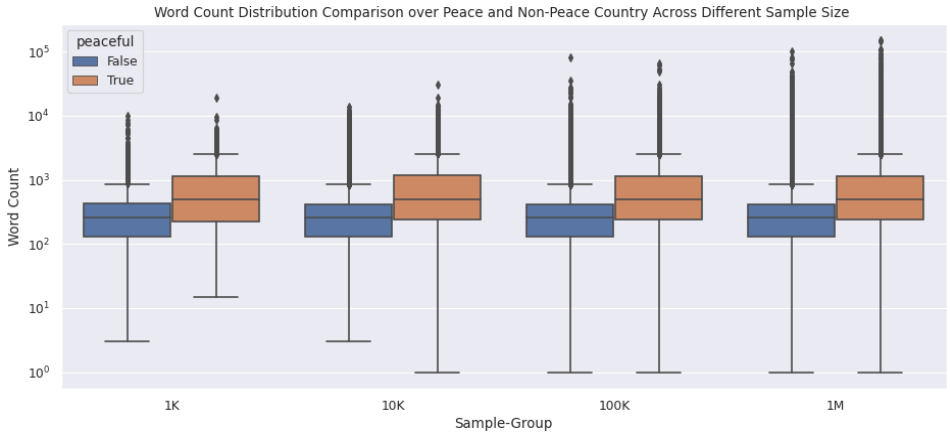


Figure 2-2: Iterative Boxplot per Peace Group

The high peace countries seem to have more words than the low peace countries according to the C.I in the summary table and the boxplot. To further validate this hypothesis, we performed a one-sided Wilcoxon signed-rank test which is suitable for our situation, since it doesn't put any assumption on the original data distribution. The result is shown in *Figure 2-3*. The p-value is so small that it's rounded to 0 in the output. Thus we have strong evidence to accept this hypothesis.

```
In [68]: scipy.stats.wilcoxon(hp_df.wordCount, y=lp_df.wordCount,
                              zero_method='wilcox', correction=False, alternative='greater', mode='auto')
Out[68]: WilcoxonResult(statistic=374641694823.0, pvalue=0.0)
```

Figure 2-3: Wilcoxon Signed-rank Test Result

## II.IV. Finalize details to the Proposed Pipeline

Based on the above EDA results, we finalized two details in our data balancing approach and BERT training plan presented in report 1. First, the theoretical approach to balance the data should be by taking the same amount of articles from each country. Since we have 10 countries in each peace group, the outcome dataset can have the balance between countries within each peace group as well as across peace groups. However, the data storage structure is not quite ideal. Under the computation resource constraints, there is a limited number of files we can scan which impedes us from finding enough data points for underrepresented countries. So we decided to focus on balancing the dataset across peace groups.

Next, most BERT encoders take a unified input length capped by 512 tokens. Unifying the length to maximum token length allows data information to be preserved to the maximum. However, if non-peace countries tend to have shorter articles, then the [PAD] tokens for filling articles length will be more frequently appearing in inputs from non-peace countries than in those from the other group. Though BERT's attention mask is applied to ignore the [PAD] tokens, McCormick, the founder of McCormick.AI, noted in his tutorial blog post [1] that it is still an unknown question of whether the [PAD] tokens have any effects in BERT's prediction result due to the difficulty in explaining deep learning models. So we shrunk the input length to 128 to confound these input length differences in two peace groups as well as to speed up the BERT training time on our large dataset.

# III. Fine-Tuning RoBERTa-base

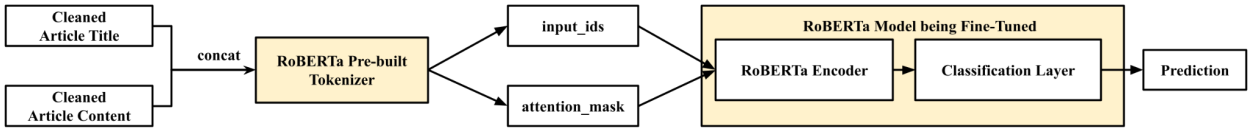


Figure 3-1: Experiment Pipeline Setting

## III.I. Experiment Setup - Train-Validation Split

The *Figure 3-1* above reiterates the experiment pipeline setting that we mentioned in the last report. During our exploration on model fine-tuning, SpaCy's default named entity recognizer appeared to fail recognizing all of the named entities. Some edge cases include: 1) all capitalized country names, and 2) the adjective form of a country or a location name. In order to alleviate the effect of these named entity residuals, we employed two approaches to split the training and validation dataset: split randomly and split by country. Split by country means to train the model on samples from a subset of countries and validate on samples from the rest. By doing so, any named entities that have high correlation with a country in the validation set would not be learned during the training time, and hence would not be key features in the model's decision function. Residual named entities most frequently occurred in two countries with the most samples in each class: India and Australia, which we refer to as the "majorities" in the following report. Thus articles from the majorities constitute our validation set, while the samples from the rest of the countries, which we refer to as "minorities", will be in our training set. If all countries in the same peace group all share some common language structures, then this approach should not weaken the strength of the model's performance.

## III.II. Experiment Setup - Training Size vs. Epochs

When trading off the training size with epochs for the sake of equally distributing the limited computational resources, we noticed the quick overfitting tendency of the RoBERTa model as shown in *Figure 3-2*. In a trial model fine-tune with 10 training epochs, default batch size of 32, and learning rate of 5e-5 on 10K data points with 8:2 train-validation random split ratio, the final validation accuracy was ~3% lower than the ~98% training accuracy on average, while using 10 times more samples to train 1 epoch gives us almost identical training and validation performance at around ~97% for the same split approach. Similarly, in the original paper of BERT, J. Devlin et al. uses only 3 epochs to fine-tune BERT for down-stream GLUE tasks [2], which again validates the observed overfit tendency of models in the BERT family. In order to expose the model with more data variations as well as avoiding overfitting, we prefer training the encoder with more samples and less epochs.

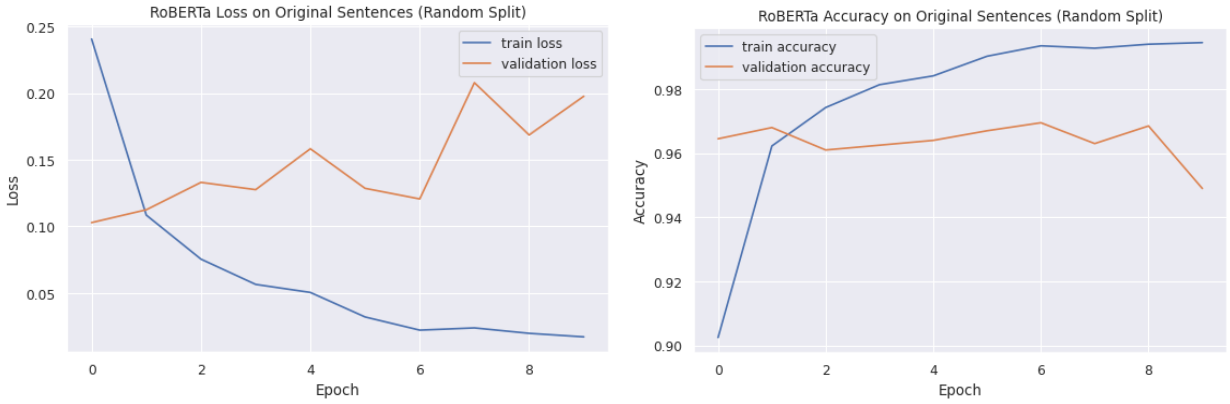


Figure 3-2: RoBERTa Performance Over 10 Epochs Using Random Split Approach

## III.III. Experiment Setup - Learning Rate vs. Batch Size

The performance of models in the BERT family is known to have a high correlation with the batch size and learning rate. Our GPU instance's memory capacity imposes an upper limit to batch-size of 32. We need a small learning rate to avoid possible model degradation on mini batches and make the model less susceptible to data outliers. Some common learning rates used by BERT models are: 3e-4, 1e-4, 5e-5, 3e-5. The positive trend in the validation loss in *Figure 3-2* indicates that the learning rate of 5e-5 might still be too large for our data.

In a previous trial run, we discovered that using a random train-validation split approach, the model has a very high accuracy and minimum spaces for improvements. On the other hand, the model performance on the dataset split by country has a large discrepancy of roughly 20-30% compared to the performance on random split dataset. So we focus on improving the model's performance on the dataset split by country when choosing a better learning rate. We fine-tuned RoBERTa using 10 different learning rates against 10K data points where a class-balanced 8K sample set from the minorities served as training and a class-balanced 2K set from majorities served as validation. *Figure 3-3* is the outcome of this experiment. We see a positive trend for loss and negative trend for accuracy as the learning rate increases. 3e-5 could be a better choice for our dataset as it produces the lowest loss and highest accuracy.

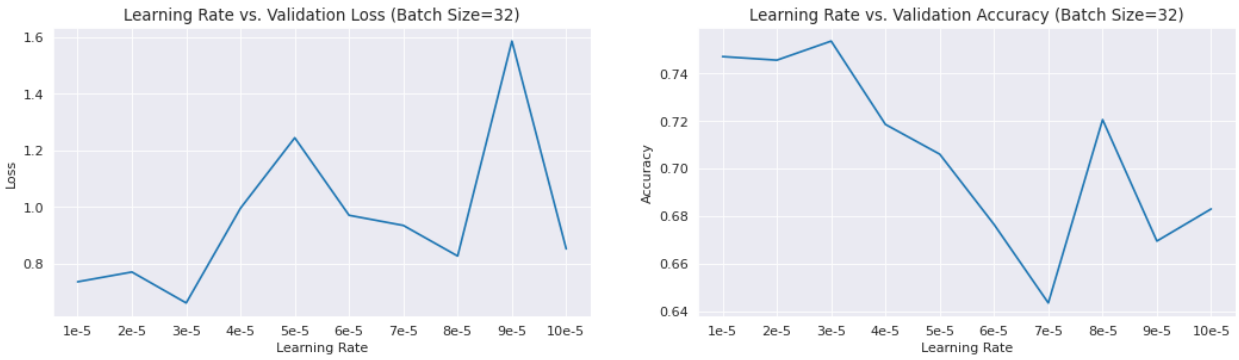


Figure 3-3: RoBERTa Performance vs. Different Initial Adam Learning Rate

We chose Adam optimizer with this initial value to schedule learning rate adaptively. We trained the model with a batch size of 32 for most of the time to speed up the training process. P. Goyal *et al.* [3] showed in their experiments the effectiveness of using *linear learning rate scaling rule*[2] in a broad range of minibatch sizes. In case we needed to adjust the batch size, this strategy is employed accordingly.

## III.IV. Results[3]

We trained RoBERTa with a logistic regression classification layer on 8K samples and validated it on 2K samples for each of the 10 stratified folds, then calculated the mean and C.I. for each metric listed in the plot below over the 10 validation evaluations. *Figure 3-4* shows all of the performance statistics distribution we observed. The performance on the random split dataset is better than the performance on split-by-country one by ~20%. Using the same train-validation split, the model's overall performance is significantly better on the original article than on the shuffled ones. With shuffling, we see ~5% performance drop in both pairs of tests due to the loss in context.

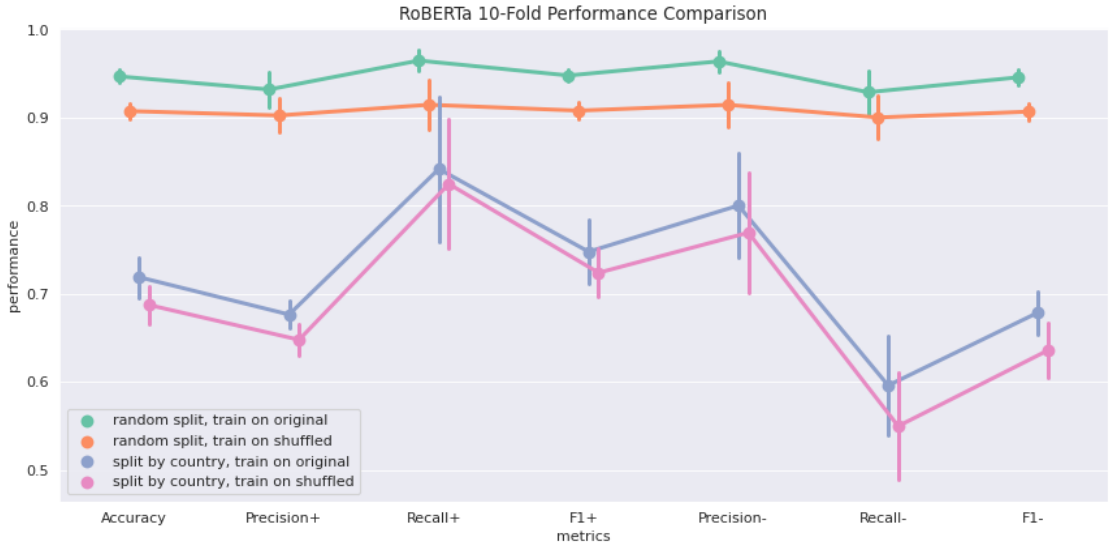


Figure 3-4: RoBERTa Fine-Tuning Performance Comparison
(+ : Positive class; - : Non-Peace class)

[2] Linear learning rate scaling rule: When the minibatch size is multiplied by k, multiply the learning rate by k.
[3] All detailed K-fold model performance statistics comparison for the three different experiment settings mentioned in this section are included in the Appendix - Table 1.

There are two hypotheses which could explain the performance drop when changing the train-validation split method: 1) the imbalance between samples from the majorities vs. minorities in both train & validation set under a random split method may drive the model to just learn the language structures from the two majorities instead of all countries, and causing the random split model to have very good trian & validation performance due to the overwhelming proportion of majorities' data in each set, or 2) the countries in the same peace group are not sharing common language structures, and peace/non-peace countries could be peace/non-peace in different ways, thus causing the split by country model to have very bad performance.

To see which of the above two hypotheses is correct, we want to see how the model performs on average over all countries. We took a fixed number of articles from the minorities, and measured the median of articles from each country in the minorities. Then we added into this dataset the median number of articles from each of the majorities, which are Australia and India, and performed a random train-validation split over this compounded dataset. As shown in *Table 3-1*, the performance of the model is roughly the same as when performing the random split.

| | | Loss (%) | Accuracy (%) | Precision (%) | Recall (%) | F1 (%) |
|---|---|---|---|---|---|---|
| Original Sentence | Train | 17.76 | 93.13 | 93.89 | 96.88 | 95.36 |
| | Validation | 11.35 | 95.77 | 95.96 | 98.29 | 97.11 |
| Shuffled Sentence | Train | 27.49 | 89.08 | 89.78 | 95.93 | 92.75 |
| | Validation | 19.43 | 92.63 | 93.96 | 95.98 | 94.96 |

Table 3-1: RoBERTa Performance Over ~13.6K Training and ~3.4 K Validation (Country Balanced Random Split)

We further shrunk the number of samples from minority countries to 10K, repeated the above procedure 10 times, and calculated the mean and C.I. for each metric to see the significance in the performance similarity. *Figure 3-5* is an illustration for the procedure. We see from *Figure 3-6* that though the performance before & after balancing the data is at about the same level, the performance variance is larger when including less Australian and Indian data.

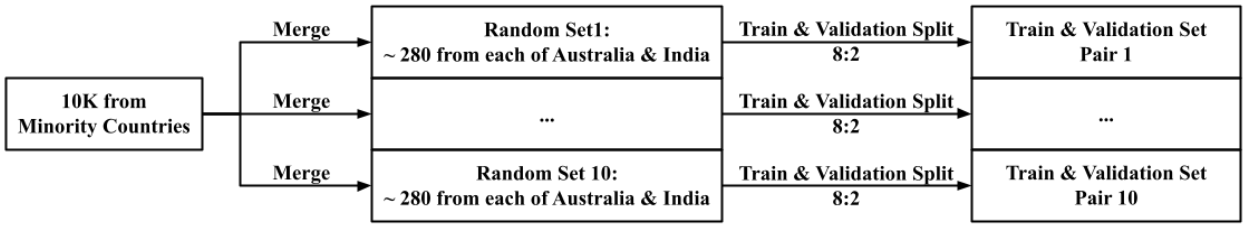


Figure 3-5: Procedure Illustration

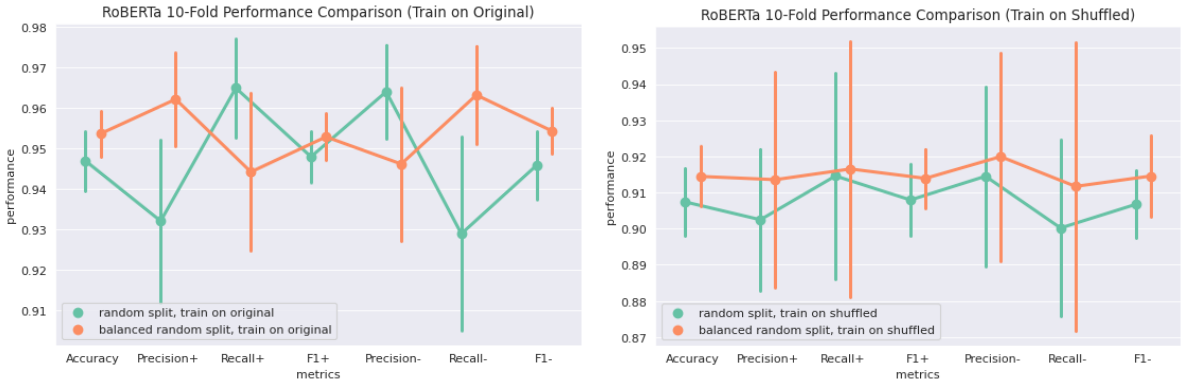


Figure 3-6: RoBERTa Fine-Tuning Performance Comparison
(+ : Positive class; - : Non-Peace class)

The good performance result indicates that our second hypothesis is correct. Peace/non-peace countries could be peace/non-peace in different ways. An analogy is when training the image classifier to classify dogs vs. cats, if all training images of dogs are Husky, then the model might fail to classify the image of Corgi during the validation process. Accordingly, there might be different types of language structures within each peace group. In addition, when splitting by country, the F1+ is significantly higher than F1- independently from whether the sentences are shuffled or not. In other words, F1- drops more than F1+ when training the model on minorities compared to those on random split dataset. A possible explanation is that the peace data, or articles from Australia, in the validation set has more shared language features with the peace data in training, while the non-peace data, or articles from India, in the validation set has less shared language features with the non-peace data in training.

## IV. Explaining the RoBERTa Models

We introduced the Local Interpretable Model-Agnostic Explanations (LIME) [4] technique to help explain what the RoBERTa Encoder had learnt during training. Rather than outputting the model's overall feature importance, LIME highlights the important words in one input sentence at a time using a given model's probability prediction function. For each input sentence, LIME randomly generates a set of neighbors of the input instance by deleting some words from the original sentence, then uses the model to get the probability prediction for each of the neighbors. These predictions are used to estimate a locally faithful decision function, where each neighbor is weighted by the proximity to the instance being explained. Though LIME outputs the individual words that are important to the decision function, we should take the context surrounding the words into consideration rather than interpreting the words alone by themselves.

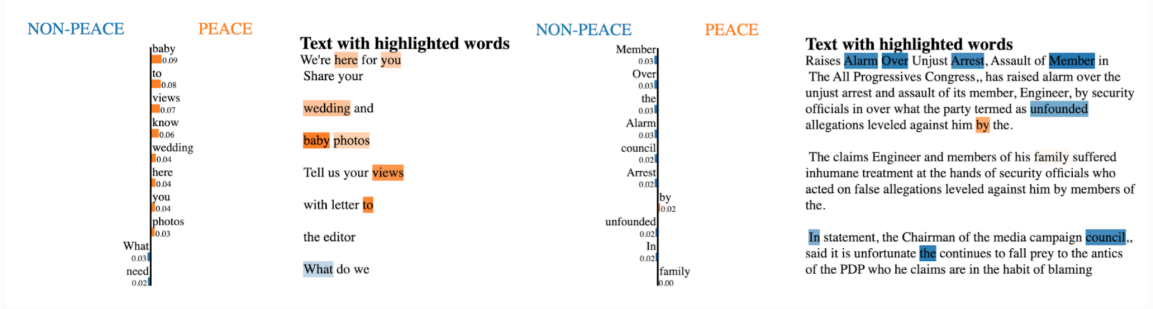


Figure 4-1: LIME Output on Training (Left: Australia, Right: Nigeria)

As suggested in the output of LIME (*Figure 4-1*), what RoBERTa encoder learned during training agrees with the meaning of peace indicators by human perception. However, the LIME algorithm is slow and non-deterministic, and human's interpretation is needed to evaluate LIME's output keyword ranking. Due to this bottleneck of LIME, it could only be used as an auxiliary tool to provide some suggestions for us to carry out our next step plan.

# V. Goals and Next Steps

Since the explanation power of LIME is limited in our topic of interest, our next-step top priority task is to keep finding ways to explain the RoBERTa model. We will continue to spend time on finding the explanations to RoBERTa's varying performance under different experimental settings as well as investigating the contents learned by the encoder. One method is to use the model trained on all countries and the model trained on dataset split by country to classify the extreme countries selected by *matched-pair*[4] method and compare their results. The set of extreme countries selected by *matched-pair* methods are disjoint from the set of extreme countries selected by *rule-of-5*. If the model's classification results on those matched-pair countries roughly match the manually classification result, then it could also prove that the model has captured the language structural differences between peace and non-peace countries.

# VI. Contribution

- Hongling Liu: Main contributor of exploratory data analysis process and co-contributor of fine-tuning RoBERTA hyperparameters.
- Haoyue Qi: Main contributor of optimizing EDA process. Main contributor of generating prediction results under Split by Country and Country Balanced Random Split experiments.
- Xuanhao Wu: Main Contributor of exploring the concept and techniques of lime explainer and using it to extract the important set of decision words used by Roberta model
- Yuxin Zhou: Contributor of preprocessing runtime optimization and run experiments on the Roberta model with different datasets.
- Wenjie Zhu: Main contributor of fine-tuning RoBERTA hyperparameters and generating the train-validate results and analysis on the articles under Random Split and Split by Country experiment.

[4] Cluster the countries using geography first, then select the highest and lowest peace countries in each geographical region, where the lowest peace country in a region is the one with the lowest number of indices classified as high-peace and highest number of indices classified as low-peace. Same logic follows for high-peace.

## References

[1] “Smart Batching Tutorial - Speed Up BERT Training · Chris McCormick,” *Mccormickml.com*, Jul. 29, 2020. https://mccormickml.com/2020/07/29/smart-batching-tutorial/.
[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv.org*, 2018. https://arxiv.org/abs/1810.04805.
[3] P. Goyal *et al.*, “Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour,” *arXiv.org*, 2017. https://arxiv.org/abs/1706.02677.
[4] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘Why Should I Trust You?’: Explaining the Predictions of Any Classifier,” *arXiv.org*, 2016. https://arxiv.org/abs/1602.04938.
[5] Y. Pruksachatkun *et al.*, “Intermediate-Task Transfer Learning with Pretrained Models for Natural Language Understanding: When and Why Does It Work?,” *arXiv.org*, 2020. https://arxiv.org/abs/2005.00628.

# Appendix

| | | Accuracy | Precision + | Recall + | F1 + | Precision - | Recall - | F1 - |
|---|---|---|---|---|---|---|---|---|
| Original Sentence | Random Split | 94.68 (94.13, 95.22) | 93.19 (91.70, 94.69) | 96.48 (95.56, 97.40) | 94.78 (94.31, 95.25) | 96.39 (95.53, 97.25) | 92.88 (91.10, 94.65) | 94.57 (93.94, 95.20) |
| | Split By Country | 71.88 (70.17, 73.58) | 67.64 (66.49, 68.79) | 84.17 (78.04, 90.29) | 74.77 (72.12, 77.43) | 80.02 (75.59, 84.45) | 59.59 (55.42, 63.76) | 67.87 (66.04, 69.69) |
| | Country Balanced Random Split | 95.35 (94.94, 95.77) | 96.20 (95.34, 97.07) | 94.40 (92.97, 95.85) | 95.27 (94.85, 95.70) | 94.60 (93.19, 96.01) | 96.30 (95.40, 97.21) | 95.42 (95.00, 95.84) |
| Shuffled Sentence | Random Split | 90.74 (90.05, 91.43) | 90.25 (88.80, 91.70) | 91.46 (89.34, 93.58) | 90.80 (90.06, 91.53) | 91.45 (89.60, 93.30) | 90.02 (88.19, 91.84) | 90.67 (89.97, 91.37) |
| | Split By Country | 68.72 (67.16, 70.28) | 64.77 (63.41, 66.14) | 82.47 (77.04, 87.90) | 72.38 (70.35, 74.42) | 76.94 (71.86, 82.02) | 54.97 (50.43, 59.51) | 63.57 (61.22, 65.93) |
| | Country Balanced Random Split | 91.45 (90.83, 92.07) | 91.36 (89.15, 93.57) | 91.65 (89.02, 94.29) | 91.39 (90.79, 92.00) | 92.00 (89.86, 94.13) | 91.17 (88.20, 94.14) | 91.46 (90.63, 92.29) |

Table 1: RoBERTa 10-Fold Performance Comparison Detailed Numbers
(+ : Positive class; - : Non-Peace class)