GLUE Tasks

•••

Group 3: Abdulaziz Gebril, Yuan Meng, Yuchen Ma

Table of Contents

- 1. Introduction.
- 2. Description of the data set.
- 3. Description of the NLP model.
- 4. Experimental setup.
- 5. Hyper-parameters.
- 6. Results.
- 7. Summary and conclusions.
- 8. References.

1. Introduction

- General Language Understanding Evaluation (GLUE) benchmark
 - Natural language understanding systems
 - Single-Sentence Similarity and Paraphrase Tasks, and Inference Tasks
- Utilizing the transfer learning method to apply the transformers models to solve the GLUE benchmark tasks
 - BERT and the RoBERTa pre-trained models
 - Customize and fine-tune
 - Load and process data
 - Train and evaluate models
- Widely used benchmark
 - Solve the same problem with different teams worldwide to get more inspiration

2. Description of the dataset - Corpus of Linguistic Acceptability

The Corpus of Linguistic Acceptability (CoLA) (Warstadt et al., 2018)

10,657 sentences from 23 linguistics publications

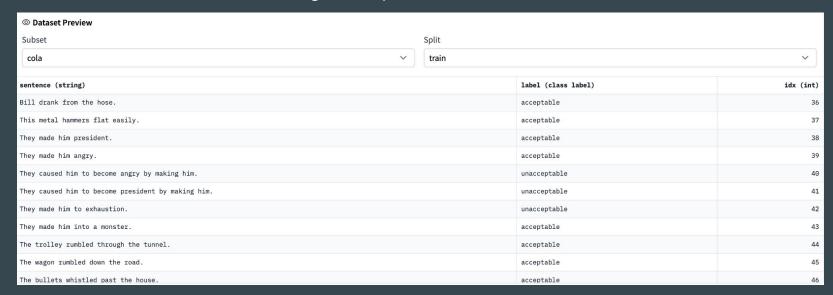
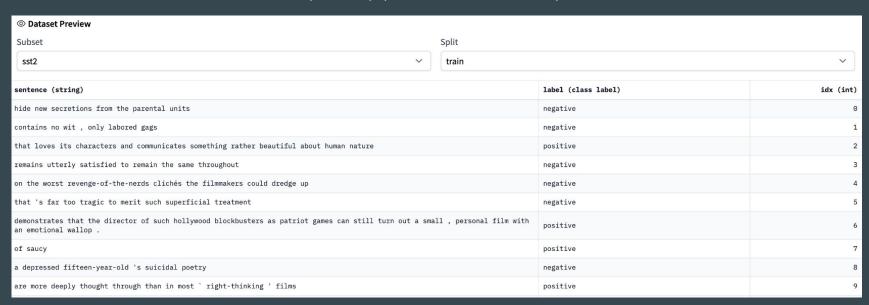


Figure from Hugging Face datasets viewer, https://huggingface.co/datasets/glue/viewer/

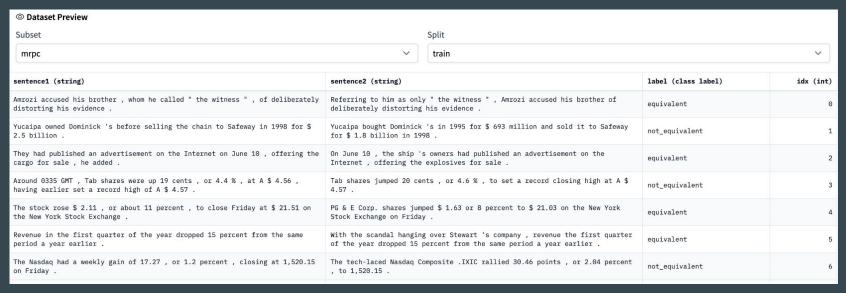
2. Description of the dataset - The Stanford Sentiment Treebank

The Stanford Sentiment Treebank(SST-2) (Socher et al., 2013) - 9,645 sentences



2. Description of the dataset - Microsoft Research Paraphrase Corpus

Microsoft Research Paraphrase Corpus for Similarity and Paraphrase Task (MRPC) (Dolan & Brockett, 2005) - 5,800 pairs of sentences



2. Description of the dataset - Semantic Textual Similarity Benchmark

Semantic Textual Similarity Benchmark (STS-B) (Cer et al., 2017) - 8,628 sentence pairs

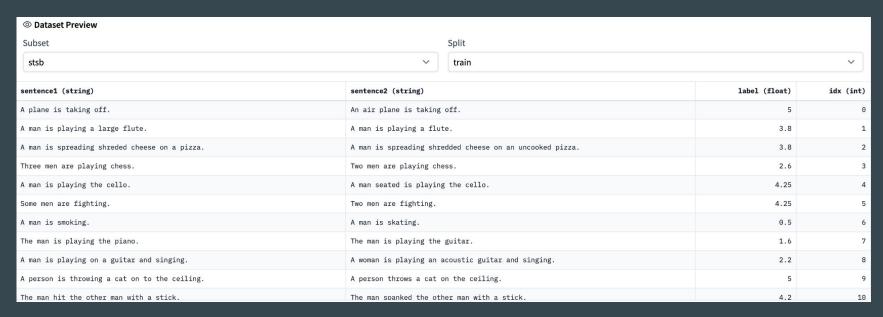
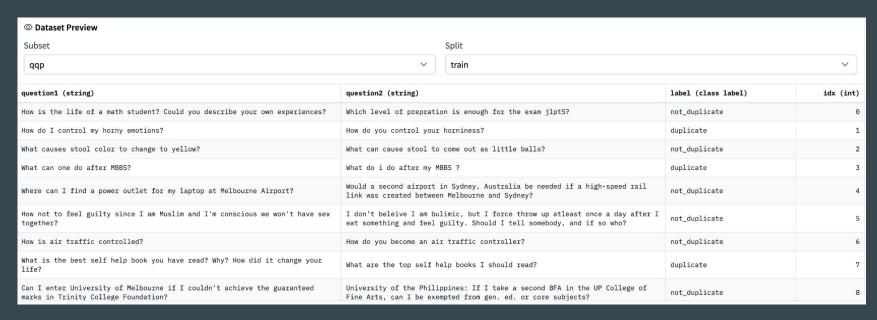


Figure from Hugging Face datasets viewer, https://huggingface.co/datasets/glue/viewer/

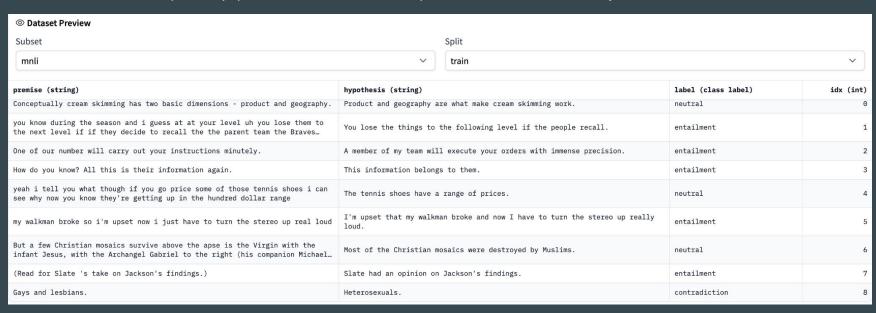
2. Description of the dataset - Quora Question Pairs

Quora Question Pairs (QQP) collecting question pairs from the website Quora



2. Description of the dataset - MultiNLI Matched

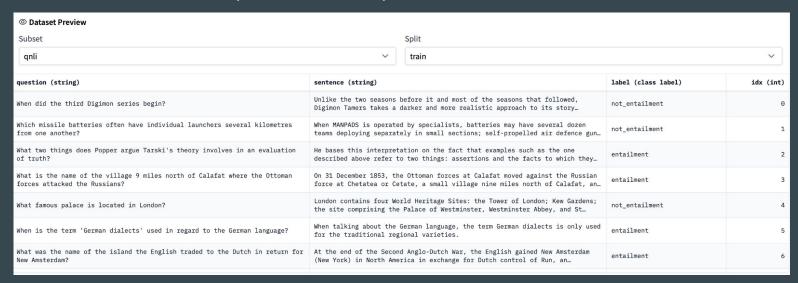
MultiNLI Matched (MNLI) (Williams et al., 2018) - 433,000 sentence pairs



2. Description of the dataset - Question NLI

Question NLI (QNLI) (Rajpurkar et al. 2016)

• more than 100,000 question-answer pairs from more than 500 articles



2. Description of the dataset - Recognizing Textual Entailment

Recognizing Textual Entailment (RTE) - combining the data from RTE1 (Dagan et al., 2006), RTE2 (Bar Haim et al., 2006), RTE3 (Giampiccolo et al., 2007), and RTE5 (Bentivogli et al., 2009)

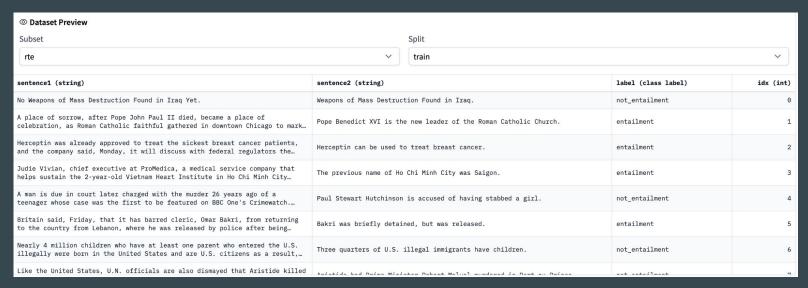
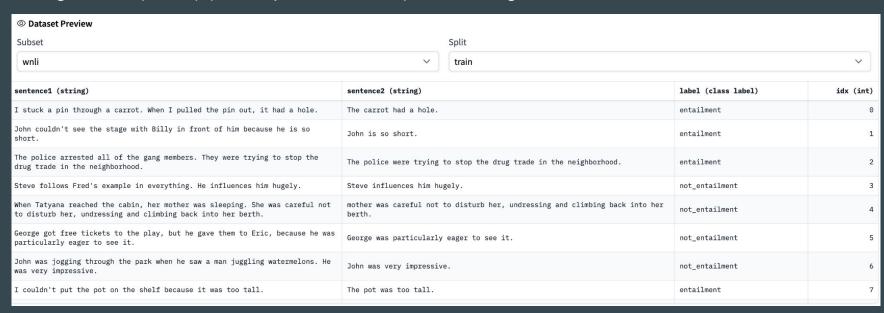


Figure from Hugging Face datasets viewer, https://huggingface.co/datasets/glue/viewer/

2. Description of the dataset - Winograd NLI

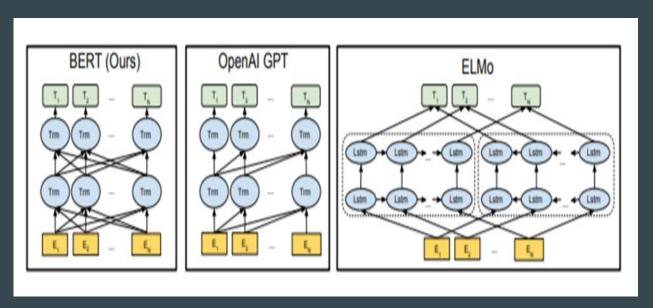
Winograd NLI (WNLI) (Levesque et al., 2011) -150 Winograd schemas



3. Description of the NLP model

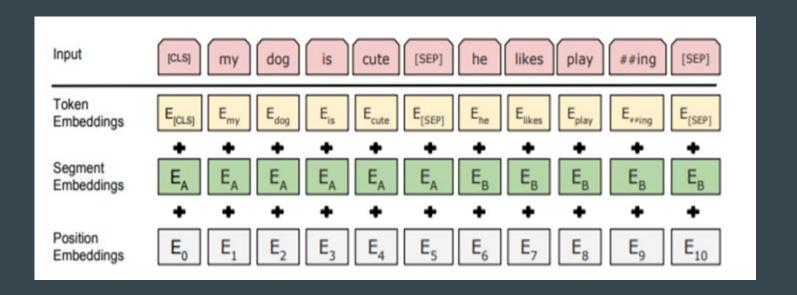
- BERT (Bidirectional Encoder Representations From Transformers)
 - Paper published by Google Al
 - BERT is pre-trained on both left and right context
 - Uses Masked language model and next sentence prediction objectives
 - Trained on Book corpus and English Wikipedia which contains 800 Million and 2,500 Million words respectively.

BERT Vs OPENAI GPT Vs ELMO



https://arxiv.org/pdf/1810.04805.pdf

BERT Input Representation



https://arxiv.org/pdf/1810.04805.pdf

3. Description of the NLP model (Cont.)

- RoBERTa (Robustly Optimized BERT Pre Training Approach)
 - Paper published by Facebook and University of Washington.
 - Showed that BERT was significantly undertrained.
 - Longer Training, bigger batches, removing NSP, Dynamic MLM
 - Trained on Book corpus and English Wikipedia as BERT, In addition, RoBERTa was pretrained on CC-News, OpenWebText and Stories.

RoBERTa With and Without NSP

Model	SQuAD 1.1/2.0	MNLI-m	SST-2	RACE
Our reimplementation	on (with NSP loss):			
SEGMENT-PAIR	90.4/78.7	84.0	92.9	64.2
SENTENCE-PAIR	88.7/76.2	82.9	92.1	63.0
Our reimplementation	on (without NSP lo	ss):		
FULL-SENTENCES	90.4/79.1	84.7	92.5	64.8
DOC-SENTENCES	90.6/79.7	84.7	92.7	65.6
BERTBASE	88.5/76.3	84.3	92.8	64.3
$XLNet_{BASE} (K = 7)$	-/81.3	85.8	92.7	66.1
$XLNet_{BASE} (K = 6)$	-/81.0	85.6	93.4	66.7

https://arxiv.org/pdf/1907.11692.pdf

Static Vs Dynamic Masking

Masking	SQuAD 2.0	MNLI-m	SST-2	
reference	76.3	84.3	92.8	
Our reimp	lementation:			
static	78.3	84.3	92.5	
dynamic	78.7	84.0	92.9	

https://arxiv.org/pdf/1907.11692.pdf

3. Description of the NLP model (Cont.)

- CustomBERT & CustomRoBERTa
 - Used both pretrained BERT & RoBERTa and added two additional layers.

Bidirectional LSTM and output layer for different GLUE tasks

CustomBERT

```
class CustomBERTModel(nn.Module):
     init (self, num_labels):
      super(CustomBERTModel, self). init ()
      self.bert = BertModel.from pretrained("bert-base-uncased")
      self.hidden size = self.bert.config.hidden size
      self.lstm = nn.LSTM(self.hidden_size, self.hidden_size, batch_first=True,bidirectional=True)
      self.clf = nn.Linear(self.hidden size*2, num labels)
def forward(self, **batch):
      sequence output, pooled_output = self.bert(batch['input_ids'],batch['attention_mask'])[:2]
      lstm output, (h,c) = self.lstm(sequence_output) ## extract the 1st token's embeddings
      hidden = torch.cat((lstm_output[:,-1, :self.hidden_size],lstm_output[:,0, self.hidden_size:]),dim=-1)
      hidden = F.dropout(hidden, 0.1)
      linear_output = self.clf(hidden.view(-1_self.hidden_size*2)) ### only using the output of the last LSTM cell to perform classification
      return linear output
```

4. Experimental setup - data preprocessing

- Auto Tokenizer
- Data Collator With Padding
- Remove & Rename Columns
 - o attention_mask
 - o input_ids
 - o labels
 - token_type_ids
- Data Loader
 - Train & validation

Datasets	Column 1	Column 2	
CoLA	Sentence	None	
SST-2	Sentence	None	
MRPC	Sentence1	Sentence 2	
STS-B	Sentence1	Sentence 2	
QQP	Question 1	Question 2	
MNLI	Premise	Hypothesis	
QNLI	Question	Sentence	
RTE	Sentence1	Sentence 2	
WNLI	Sentence1	Sentence 2	

4. Experimental setup - data modeling

- Customize transformer models
 - Load pre-trained models
 - Sequence output shape: (batch size, sequence length, hidden size)
 - Add LSTM layer
 - Batch first and bidirectional
 - Define hidden layer and drop outs
 - Concatenate the LSTM outputs
 - Attach final linear layer
 - Output feature = number of labels
- Optimizer: AdamW
- Criterion: Cross-Entropy Loss & Mean Squared Error
- Number of training steps: number of epochs * length of train dataloader

4. Experimental setup - model evaluation

Task	Metric
CoLA	Matthew's Corr
SST-2	Accuracy
MRPC	F1 & Accuracy
STS-B	Pearson & Spearman
QQB	F1 & Accuracy
MNLI	Accuracy
QNLI	Accuracy
RTE	Accuracy
WNLI	Accuracy

5. Hyper-parameters

Task	Batch size	Epoch	Learning rate
CoLA	64	30	
SST-2	64	5	
MRPC	16	5	
STS-B	8	5	5e^5
QQP	32	5	
MNLI	32	5	
QNLI	32	5	
RTE	16	5	
WNLI	4	30	

6. Results

Task	Batch size	Epoch	Learning rate	BERT	RoBERTa
CoLA	64	30		Matthew's Corr: 0.5931	Matthew's Corr: 0.6131
SST-2	64	5		Accuracy: 0.9243	Accuracy: 0.938
MRPC	16	5		F1: 0.9091 Accuracy: 0.8701	F1:0.927 Accuracy: 0.8995
STS-B	8	5	5e^5	Pearson: 0.8780 Spearman: 0.8757	Pearson:0.90488 Spearman Corr: 0.902
QQP	32	5		F1: 0.8789 Accuracy: 0.9095	F1: 0.885 Accuracy:0.9144
MNLI	32	5		Accuracy: 0.8338	Accuracy: 0.865
QNLI	32	5		Accuracy: 0.9028	Accuracy: 0.9218
RTE	16	5		Accuracy: 0.6245	Accuracy:0.711
WNLI	4	30		Accuracy: 0.5493	Accuracy: 0.577

6. Results (Cont.)

From the result, we can find that the BERT algorithm and RoBERTa algorithm are good for SST-2 task, MPRC task, QQP task and QNLI task especially. For these tasks, we all get high scores from our models. Comparing the results we get from the BERT model and RoBERTa mode, we can find that for each task, RoBERTa mode always has a better result, although the difference is not much. As mentioned, to improve the training procedure, RoBERTa removes the Next Sentence Prediction (NSP) task from BERT's pre-training and introduces dynamic masking so that the masked token changes during the training epochs. Moreover, RoBERTa uses larger text for pre-training. Therefore, we can conclude that RoBERTa model outperforms the BERTA model.

7. Summary and Conclusion

In the project, we learn how to build the architecture for Bert and RoBerta and analyze natural language through nine tasks. For each task, we use different metrics to evaluate their performance. When we run the code, we also met some problems due to large size of the task, so we change hyperparameters and run in GPU. From metrics, we conclude that Bert and Roberta both work well for nine tasks and Roberta works better. There are still many transformers which have similar functions. It is also interesting for us to explore in the future.

8. References

- 1. GLUE benchmark, https://gluebenchmark.com/tasks
- 2. Hugging Face, https://huggingface.co/models
- 3. BERT: Pre-training of Deep Bidirectional Transformers For Language Understanding. https://arxiv.org/pdf/1810.04805.pdf
- 4. RoBERTa: A Robustly Optimized BERT Pretraining Approach. https://arxiv.org/pdf/1907.11692.pdf