Metaphor Detection

Group 22

Introduction

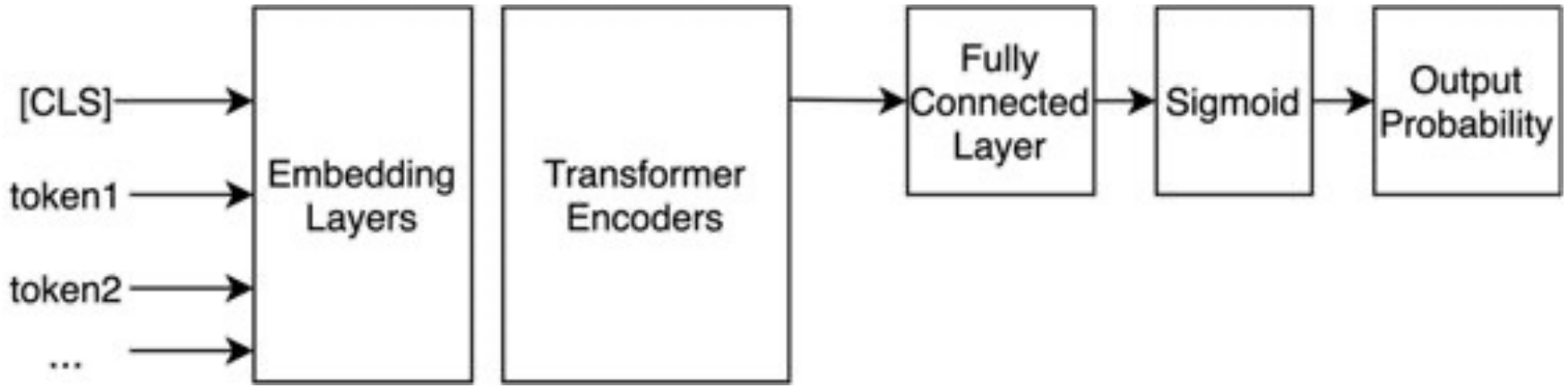
- Metaphors are an essential part of language and are often used to convey complex ideas and emotions in a more relatable and user-friendly way. However, detecting and understanding metaphors in text can be challenging for machines. This is where our project comes in.
- A successful metaphor detection system can help in language understanding as well as other tasks such as machine translation and sentiment analysis. These systems can also help extract high level information from texts.
- We aim to implement metaphor detection at the token level by developing a natural language processing model that can accurately detect metaphors in text and provide insight into their meaning and context. We have utilized XLM-Roberta model by Facebook, which was specifically designed to handle multiple languages, which we selected due to the dataset we are using. We fine tune the model and then create a pipeline for using the model for inference.

Dataset

- We are using parts of the **VU Amsterdam Metaphor Corpus.** The VU Amsterdam Metaphor Corpus dataset is a widely used resource for metaphor detection research because it is a large, high-quality dataset that provides a diverse range of metaphors from a variety of sources. main advantages of using the VU Amsterdam Metaphor Corpus is that it includes both linguistic and conceptual metaphors, providing detailed view of how metaphors are used in different contexts.
- dataset (around 182k records) contains following fields:
  - Id: Unique Id for each sentence.
  - Sentence : The actual text on which we must do token level metaphor detection.
  - Words: All the individual words for the sentence.
  - Labels: binary labels depicting if the word is a metaphor or not, 0 = not a metaphor , 1 = is a metaphor.
  - Sentences Train: 10898
  - Sentences Validation: 1211
  - Sentences Test: 4080

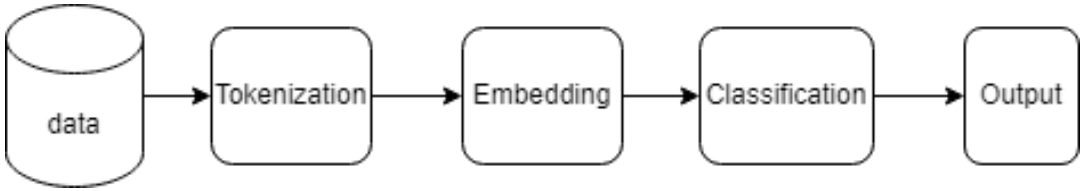
Introduction to XLM RoBERTa

- The XLM-RoBERTa (Cross-lingual Language Model with Robustly Optimized BERT Approach) is a variant of the BERT (Bidirectional Encoder Representations from Transformers) model architecture that was developed by Facebook AI Research (FAIR).
- It is based on the Roberta architecture, which in turn is a variant of the popular BERT architecture.
- XLM-Roberta was specifically designed to support cross-lingual language understanding, meaning it can effectively process and understand text in multiple languages.



Implementation

- Pipeline implementation:



- Tokenization: The input text is first tokenized into a sequence of tokens, with each token representing a word or subword in the text. XLM-RoBERTa uses byte-pair encoding (BPE) to represent the input text as a sequence of subwords. The tokenized sequence is then processed by a series of Transformer encoder layers, which use self-attention mechanisms to capture contextual relationships between the words in the sequence.
- Embedding: Each token is then embedded into a high-dimensional vector space, where its meaning is represented as a vector. XLM-RoBERTa uses a multi-layer bidirectional transformer architecture to generate these embeddings.
- Classification: Once the tokens are embedded, XLM-RoBERTa predicts whether each token is metaphorical or literal. This is done by adding a classification layer on top of the transformer and training it on a labeled dataset, such as the VU Amsterdam Metaphor Corpus.
- Output: The output of the model is all the words of the sentence with their respective labels and score. The model takes in a sequence of words and assigns a label of either "metaphoric" or "literal" to each word in the sequence.

Classification

- A pre-trained AutoModelForTokenClassification model is used for token classification, which means that it assigns a label to each token in the input sequence.
- We initialized a Hugging Face Trainer object which was used to fine-tune the pre-trained model on the training dataset.
- The training loop runs for the specified number of epochs, and for each epoch, it iterates over the training dataset in batches, computes the loss on each batch, and updates the model parameters.
- Once the training is finished, the trainer object returns a TrainingOutput object that contains the training and validation loss, as well as other metrics such as the learning rate and the training time.

Results

Epoch	Training Loss	Validation Loss	Precision	Recall	F1
1	0.156600	0.152184	0.936410	0.940578	0.936667
2	0.118800	0.140348	0.944923	0.947895	0.944266
3	0.088500	0.145418	0.950982	0.952792	0.951560
4	0.062200	0.169173	0.949723	0.951747	0.950319
5	0.046200	0.191173	0.950063	0.952077	0.950647
6	0.034600	0.219432	0.952166	0.954003	0.952696
7	0.027200	0.240892	0.952177	0.953893	0.952724
8	0.019800	0.262339	0.951831	0.953232	0.952364

This image demonstrates general evaluation metrics that we use for determining performance

- In the context of metaphor detection, precision refers to the fraction of detected metaphors that are actually metaphors
- recall refers to the fraction of all actual metaphors that were detected by the model
- F1 score is a measure of the overall performance of the model on detecting both metaphors and literals

Code + Text

	precision	recall	f1-score	support
l	0.95	0.98	0.97	16215
m	0.80	0.60	0.68	1960
accuracy			0.94	18175
macro avg	0.88	0.79	0.83	18175
weighted avg	0.94	0.94	0.94	18175

	precision	recall	f1-score	support
l	0.96	0.99	0.97	16215
m	0.85	0.62	0.72	1960
accuracy			0.95	18175
macro avg	0.90	0.81	0.85	18175
weighted avg	0.94	0.95	0.94	18175

	precision	recall	f1-score	support
l	0.97	0.98	0.97	16215
m	0.82	0.73	0.77	1960
accuracy			0.95	18175
macro avg	0.89	0.85	0.87	18175
weighted avg	0.95	0.95	0.95	18175

	precision	recall	f1-score	support
l	0.97	0.98	0.97	16215
m	0.82	0.71	0.76	1960
accuracy			0.95	18175
macro avg	0.89	0.85	0.87	18175
weighted avg	0.95	0.95	0.95	18175

This image demonstrates the results for each epoch, we can also observe the precision, recall and F1-score for both Literal and Metaphor Class

Results

```
trainer.evaluate(val_dataset)
```

[255/255 20:20]

	precision	recall	f1-score	support
literal	0.97	0.98	0.97	16215
metaphoric	0.82	0.73	0.77	1960
accuracy			0.95	18175
macro avg	0.89	0.86	0.87	18175
weighted avg	0.95	0.95	0.95	18175

```
{'eval_loss': 0.2408924102783203,
 'eval_precision': 0.9521768545967099,
 'eval_recall': 0.9538927097661624,
 'eval_f1': 0.952724374582013,
 'eval_runtime': 9.6323,
 'eval_samples_per_second': 125.723,
 'eval_steps_per_second': 7.89,
 'epoch': 8.0}
```

This image demonstrates our evaluation metrics on the validation dataset

```
trainer.evaluate(test_dataset) # best F1 of class "m": 0.
```

[255/255 01:06]

	precision	recall	f1-score	support
l	0.96	0.98	0.97	51540
m	0.82	0.70	0.76	6819
accuracy			0.95	58359
macro avg	0.89	0.84	0.86	58359
weighted avg	0.95	0.95	0.95	58359

```
{'eval_loss': 0.24491505324840546,
 'eval_precision': 0.9454023515662892,
 'eval_recall': 0.947737281310509,
 'eval_f1': 0.9459562991897741,
 'eval_runtime': 34.2714,
 'eval_samples_per_second': 119.05,
 'eval_steps_per_second': 7.441,
 'epoch': 8.0}
```

This image demonstrates our evaluation metrics on the test dataset.

Results

```
[34] pipeline_metaphors("This young man knows how to climb the social ladder")
0s

[{'entity_group': 'metaphoric',
  'score': 0.99068636,
  'word': 'This',
  'start': 0,
  'end': 4},
 {'entity_group': 'literal',
  'score': 0.9999874,
  'word': 'young man knows how to',
  'start': 5,
  'end': 27},
 {'entity_group': 'metaphoric',
  'score': 0.99974257,
  'word': 'climb',
  'start': 28,
  'end': 33},
 {'entity_group': 'literal',
  'score': 0.9999937,
  'word': 'the social',
  'start': 34,
  'end': 44},
 {'entity_group': 'metaphoric',
  'score': 0.9994843,
  'word': 'ladder',
  'start': 45,
  'end': 51}]
```

Interpretation of the metaphorical sentence picked up from google to see if our prediction is accurate:

In the input sentence "This young man knows how to climb the social ladder", there are two possible interpretations of the entities as either metaphors or literals, depending on the context and the intended meaning of the sentence.

Here is one possible interpretation:

- "This" could be a metaphoric entity, representing an abstract concept such as an opportunity or a chance to climb the social ladder.
- "Young man knows how to" could be a literal entity, indicating that the subject of the sentence is a young man who knows how to climb the social ladder.
- "Climb" could be a metaphoric entity, representing the act of rising in social status or achieving success.
- "The social" could be a literal entity, indicating that the social ladder being climbed is a real, physical ladder.
- "Ladder" could be a metaphoric entity, representing the idea of upward mobility or achieving a higher status or position in society.

Future Work

- Implement our model on Dutch language and compare the F1-score.
- use the output of metaphor detection model to identify metaphoric language in text and then use that information to inform the sentiment analysis. For example, if a sentence contains a metaphorical expression such as "My heart sank like a stone," a metaphor detection model would identify this expression as metaphoric. This information could then be used to adjust the sentiment analysis score of the sentence. Without the metaphor detection, the sentiment analysis model may interpret the sentence literally and miss the negative sentiment conveyed by the metaphor.
- Focus on improving performance for the sentences that have multiple metaphors (>3).
- Explore other options to perform metaphor detection such BI-LSTM, other BERT based models and compare it with our model.

References

- Motivation Reference (Systematic Analysis of Image-Schematic Conceptual Metaphors): https://aclanthology.org/2022.flp-1.7.pdf
- Nueral Metaphor Detection in Context: https://arxiv.org/pdf/1808.09653.pdf
- XLM roBERTa model: https://huggingface.co/xlm-roberta-base
- Unsupervised Cross-lingual Representation Learning at Scale: https://arxiv.org/pdf/1911.02116.pdf

Thank You!

Questions …..?