Integrated Gradient as An Explainability Methods

1) How does Integrated Gradient work in NLP model?

   An Attribution method scores the input data based on the predictions the model makes, i.e. it attributes the predictions to its input signals or features, using scores for each feature:

   Sensitivity: If an input feature changes the classification score in any way, this input should have an attribution value not equal to 0.
   Implementation invariance: The result of the attribution should not depend on the design and structure of the neural network. Thus, if two different neural networks provide the same prediction for the same inputs, their attribution values should also be identical.
   Integrated Gradients using the Example of a Sentiment Analysis

   Reference: https://arxiv.org/pdf/1703.01365.pdf

2) Ready available package for model explainability based on integrated gradient algorithm, namely Sequence Classification Explainer

   Limitation: Input tokens are limited to 274 tokens.

   Workaround: Split the text into multiple subtexts, classify each of them and take the chunk with the maximum attribution scores as predicted class.

   Findings:
   (Please refer to excel for details support information compare)

   a) **Higher** accuracy rate with full text classification, truncated text (i.e. subtexts) may not paint the full picture although we have chosen the subtexts with highest attribution scores. *(Token<274 vs Token>=274)*

No	Attribution Score	Predicted Class	True Label	Label Index	num tokens bert	Prediction	Prediction Index	Accuracy
1	1.64	0	dismissal	1	706	Incorrect	0	65%
2	2.05	0	dismissal	1	693	Incorrect	0	
3	-0.06	1	dismissal	1	229	Correct	1	
4	2.1	1	approval	0	862	Incorrect	0	
5	2.1	1	approval	0	862	Incorrect	0	
6	3.58	0	approval	0	991	Correct	1	
7	1.73	0	approval	0	208	Correct	1	
8	1.44	0	approval	0	737	Correct	1	

Higher accuracy rate with full text classification, truncated text (i.e. subtexts) may not paint the full picture although we have chosen the subtexts with highest attribution scores.

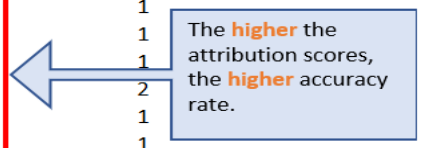
   b) Sequence Classification Explainer works **better** with less token. *(Token>=274)*

Count of Pred num token	Prediction Correct	Incorrect	Grand Total
183	1		1
201	2		2
207		1	1
208	1		1
210		1	1
214	1		1
221	1		1
224	1	1	2
228	1		1
229	1		1
340	1		1
414	1		1
422		1	1

Sequence Classification Explainer works **better** with less token.

c) The **higher** the attribution scores, the **higher** accuracy rate. *(Token<274)*

Count of Prediction Attribution Score	Prediction Correct	Incorrect	Grand Total	Count of num toke
-1.18	1		1	
-0.15		1	1	
-0.06	1		1	
-0.05	1		1	
0.2	1	1	2	
0.21		1	1	
0.36		1	1	
0.45	1	1	2	
0.62		1	1	
0.97	1		1	
1.13	2		2	
1.16		1	1	
1.33	1		1	



In addition, I only uses toy-size dataset for my project due to time limitation and running environment (always crashed / went idle once I use more than toy-size data). I think if we can run through all the data, we can conclude more solid hypothesis confidently. Hypothesis such as:

- If language will affect the interpretation as we have a pool of mixed language if we have separate language model we may have higher accuracy rate.
- Lengthy texts (i.e. number of tokens) will affect the accuracy rate of the model.
- we may set a attribution scores threshold to increase the confidence level of model interpretability.

3) SequenceClassification.ipynb file readme:

1) Set-up project environment and install package
2) Load Bert Model (multilingual) and load input data
3) Full-text Sequence Classification - 10 records per run*
4) Sub-text Sequence Classification - 10 records per run*
5) Full-text Sequence Classification – Visualization
6) Sub-text Sequence Classification – Visualization
7) Poster Content
8) Sequence Classification Workings
9) Captum Workings^

* I have summarised the result for item 3 and 4 into excel file as per attached.

^ I encountered roadblocks in running the captum package. Hence, I change to the sequence classification explainer introduced by Joel. Both packages use the same algorithm Integrated Gradient algorithm.

Reference: https://github.com/cdpierse/transformers-interpret#sequence-classification-explainer)

Special Thanks to Joel for the guidance and advice along the way of the project development.