NLP Course Project

Umberto Altieri
Master's Degree in Artificial Intelligence, University of Bologna
{ umberto.altieri}@studio.unibo.it

Project code repository from Github link code (click here).

Abstract

In this piece of work I tackled a multi-label text classification problem taken from the Touchè23 challenge. Feeding a model based on Bert with premises, stances and conclusions I have been able to reach an F1-score of more than 0.5. However, the model has an hard time in classifying poorly represented classes, on which its performance metrics' values plummet. Indeed, I identified that the main reason of poor results can be retrieved in the limitations in quality and size of the dataset itself, this is the reason why I manipulated the train and validation dataset.

1 Introduction

The addressed problem, which has been proposed as a challenge from the touchè company, falls within the scope of NLP, more precisely it is known as multi-labels text classification. In details, the problem asks for classification of a textual argument according to a right human value category. In particular, each textual argument has been classified according to 20 categories compiled from the social science literature. The dataset provide by the challenge is available at the following link

However, what differs this problem from the others is that the sentences to be classified are actually sentences constructed from a premise, a stance and a conclusion. In addition, the sentences do not have any recurrent words of a given label, but the words are many and vary continuously, so there are not any capable of binding the sample data to the corresponding classes. Therefore, an approach that recognizes the words would not have been correct. Indeed, the common way to solve this kind of problem is by making use of an NLP model. For this reason, me made use of Bert to tackle the problem.

The adopted approach consists in a first preprocessing of textual data, its tokenization, and feeding to a Bert model for the multi label classification. [1] Part of the code related to the optimization of the results are provided by a tutorial of Towardsdatascience website. [2]

2 System description

The system that I implemented is based on a Bert model, more specifically its "bert-base-uncased" version, available in HuggingFace. This model has been pretrained, and it exists a version for many common tasks of NLP. According to the request of this problem, I made use of the Bert model for sequence classification. Since it has already been pretrained, it requires only a few epochs to train in order to be finetuned to accomplish the current task, as the system has been built to perform well on the data provided by the challenge. The data format given by the challenge consists in sentences made by a "premise" followed by a "stance" and a "conclusion". In order to compose a sentence, this 3 sequences of text, have been preprocessed by a function. Firstly, this function removes punctuation, not so meaningful for the model. Then it makes the text in lower case to match the fact that the uncased version of Bert is being used Finally, it performs a lemmatization step of the words in text. The lemmatizer is used because the text contains some abbreviations or declination of the same word, so it tries to bring different inflected forms back to the same word. This initial preprocessing is useful for the tokenizer, because being its input the output of the preprocessing step, it will be cleaner. The tokenizer is available in HuggingFace too, and it is also pretrained and compatible with the Bert model in use. The sentences are then tokenized and returned as input_ids , token_type_ids and attention_mask. The data processed is the one contained in the training and validation sets. In the end, the model is finetuned performing a training phase. As

[1]Bert model used.
[2]Tutorial Towardsdatascience

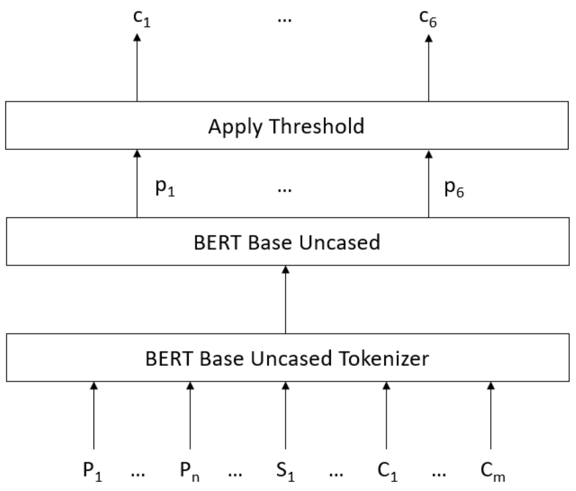
I will point out in 3, I computed performance metrics on the zhihu validation set according to the merge between the train set and one of the two validation ones. The model generates multi label classification, that converts for each label a float value to integer according a threshold set to 0.5, but after the generation on the validation zhihu dataset the performance are evaluated according the threshold. The threshold has been optimized in order to get better performances.

- architecture: includes preprocessing of data, the tokenizer (HuggingFace), Bert model (HuggingFace) and the optimizer of threshold.
- coding: The part of code related to the optimization of the threshold is inspired by Towardsdatascience tutorial, while the rest of the code is fully made by us.

The architecture of the model used is shown in the Figure below. In order to understand the image, take into account the following legend:

- P_i: i-th word of the premise;
- S_i: i-th word of the stance;
- C_i: i-th word of the conclusion;
- p_i: probability of class i;
- c_i: class i (0/1);

The model produces only 6 probabilities as output, one for each category value, because I decided to reduce the number of labels, as mentioned on the challenge website, and it is further discussed in the next paragraph.



3 Data

The datasets provided by the Touchè challenge are three: one is called training dataset, another one validation and a last one called validation "zhihu". These datasets contain the textual argument to be classified and the labels according to which textual argument should be classified. The textual arguments are divided in 3 columns, like: premise, stance and conclusion. Each textual argument is followed by 20 labels, according to which it is classified. This 20 labels are value categories compiled from the social science literature, assuming value of 1 when the argument matches with that category and of 0 when it does not. An argument can belong to more than one class. The textual arguments are merged together in order to get a sentence made by "premise"+' '+"stance"+' '+"conclusion". The obtained sentences can contain punctuation symbols and possibly have a non uniform case. As a result, all the data are processed to get lower-cased sentences without punctuation symbols. The sentences are then processed by a lemmatizer, and used as input to the model. The labels are of numerical types and have been one-hot encoded. Thus, the 20 classes are mapped to 20 0/1 features.

Inspecting the training dataset I have realized that its size is too small, especially if compared to the ones of similar classification problems. Indeed, there are 5393 samples, which are also not evenly distributed. To check this distribution you can refer to 1, where it is reported the count of occurrences for each of the twenty class labels.

As I can see, the three datasets are not balanced, but they seems to be representative. In addition, for every label, its number of occurrences is proportional to the size of the dataset. The training data is not balanced, so the model has difficulties at learning how to properly classify a textual argument, especially when it comes to predict labels with less samples. For instance, it can be seen that some value categories such as hedonism, stimulation, conformity interpersonal, face and humility have a too small number of occurrences in the training, validation, and validation zhihu dataset, so the model has problems in predicting them, as it is not able to sufficiently generalize these classes.

One could consider to apply some undersampling or oversampling techniques, but concerning the first technique, it would mean to dramatically decrease the size of the dataset, resulting in having datasets with too few samples for every class. For

	Train set	Val. set	zhihu set
Self-direction: thought	988	251	6
Self-direction: action	1395	496	11
Stimulation	247	138	0
Hedonism	172	103	2
Achievement	1512	575	39
Power: dominance	610	164	1
Power: resources	625	132	19
Face	382	130	1
Security: personal	2000	759	30
Security: societal	1728	488	31
Tradition	568	172	0
Conformity: rules	1177	455	15
Conformity: interpersonal	207	60	1
Humility	395	127	5
Benevolence: caring	1332	633	12
Benevolence: dependability	806	268	3
Universalism: concern	2081	687	21
Universalism: nature	427	127	8
Universalism: tolerance	664	223	2
Universalism: objectivity	1054	371	26

Table 1: Number of label occurrences in each dataset

what concerns oversampling, the model will overfit to the previously less represented classes due to the fact that the proportion between label occurrences would have been completely broken. As a result, the data would become also not representative of real scenarios, making us expect that in an eventual evaluation on a test set or on a real scenario, the results would have been completely unreliable.

This is the reason why I have decided to stick with the six categories suggested by the challenge, which are: Self-direction, Achievement, Security: personal, Security: societal, Benevolence: caring, Universalism: concern. Indeed, the mentioned classes are also the ones with the highest number of samples. Consequently, cutting down the number of classes to these six, I ended up with a more balanced dataset. Prior to switching from 20 labels to only 6, the results for the labels with a small number of occurrences were really bad compared to labels with more than 1200 occurences. Then I have also decided to include the first validation set into the training one, in order to have more samples to train the model and a dataset more representative than the training one alone. Indeed, this choice made us able to improve model performances, and after the merging of the validation and training sets, the f1 score gained around ten percentage points, reaching a value of around 59%, computed on validation zhihu dataset.

Furthermore, I enrolled to the challenge and requested the test set provided by TIRA, which unfortunately contains only the input samples and not the true labels, so there is no way to compute metrics on it. Finally, to make evaluations, I have decided to use only the second validation set put at our disposal, which is the "zhihu" one.

In order to prepare the dataset for multi-label classification, I have concentrated one-hot encoding of the class labels into one single feature, so that a sample will have a list of six 0/1 values corresponding to the six available classes.

4 Experimental setup and results

For a first try, I decided to use some Machine Learning algorithms to perform classification. Even though our expectation was to get not so good results, I wanted to compare them with the ones obtained using a neural architecture. I tried to use binary relevance and classifier chain, but but both tries lead us to poor results, as the f1 score reached by these 2 kind of algorithms was respectively 30 and 31 %.

Therefore I switched to a model based on LSTM and many dense and dropout layers, which all in all was a pretty simple architecture, with few parameters. In the end, it reached 27% of f1 score, so I dropped this idea and kept it just as a baseline. Before going for our final choice (Bert), the XLNet model (base version) has been tested many times but it never performed as the Bert model (bert base uncased) chosen for this task, with differences in terms of f1-score of more than 5 percentage points. In addition to the model, many tests have also been done on its input. Since the dataset offered 3 inputs "premise,stance and conclusion", it was not clear which was the best way to give the input to the model. Indeed, I started with very simple solution like passing only the premise, but this was virtually similar among many samples and thus did not return the desired result. So I tried passing only the conclusion as input, which changed for each sample; the results improved but not that much. Finally, I have decided to build a sentence for each sample, as 'premise'+' '+'stance'+' '+'conclusion'. This choice led to better results and it is the one I kept. Finally I tried making the whole text in lower case, removing punctuation, and finally using a lemmatizer to determine the lemma of a word based on its intended meaning, in other words reducing inflectional and derivationally related forms of a word to a common base form. The model returns as output the probability attributed to each label, which is then converted to 1 or 0 depending on whether or not it exceeds a threshold that has been firstly set to 0.5. Then this threshold was optimized via a threshold optimization algorithm to achieve better performances in terms of f1 score. All in all, using Bert and the rest of the setup, I have been able to get results the validation zhihu dataset that are better or at least comparable with the ones that can be found in the challenge leaderboard available on Tira website. As a disclaimer, I have to mention that most probably those performance metrics have been computed using all the label classes.

During the development phase, I had to make seveal choices. Some of them are:

- max_length: it is a parameter of the tokenizer, that has been set to 300 because the maximum of the average length computed on the provided datasets was 200 and then I added 100 in order to ensure that longer input sequences would to not be truncated.
- I used the optimizer AdamW, using which the models generalize much better than models trained with Adam.
- Use of BCEWithLogitsLoss in the training step: this loss combines the sigmoid layer and BCEloss into one class. This version is more numerically stable than using a simple sigmoid followed by BCELoss because by combining operations in one layer I exploit the log-sum-exp trick to achieve numerical stability.

According to the challenge I have computed precision, recall and f1-score for each label, but consider the f1-score with macro average as the main one to decide whether a model was good or not. As mentioned before, the model is trained on a dataset made by the merge of the training and validation sets provided by Touchè company, while the validation zhihu one is used to compute the metrics, because the test set provided by Tira does not come with ground truth labels.

The following tables contain the performances of our model, and the baseline model of Touchè company. The first table contains our results, and the second the baseline.

	Precision	Recall	F1-score	Support
Self-direction: action	0.53	0.73	0.62	11
Achievement	0.67	0.74	0.71	39
Security: personal	0.44	0.77	0.56	30
Security: societal	0.91	0.32	0.48	31
Benevolence: caring	0.44	0.58	0.50	12
Universalism: concern	0.77	0.48	0.59	21
micro avg	0.58	0.60	0.59	144
macro avg	0.63	0.60	0.57	144
weighted avg	0.66	0.60	0.59	144
samples avg	0.60	0.63	0.58	144

Table 2: Performance of our model on zhihu validation set

	Precision	Recall	F1-score	Support
Self-direction: action	0.11	1	0.198	8
Achievement	0.39	1	0.561	34
Security: personal	0.3	1	0.462	24
Security: societal	0.31	1	0.473	27
Benevolence: caring	0.12	1	0.214	7
Universalism: concern	0.21	1	0.347	16

Table 3: Performance of Touchè baseline model on zhihu validation set

Moreover I sent the results of the model on the test set but they will be evaluated by Touchè only at the end of the challenge.

5 Discussion

The results I have obtained may seem not excellent, indeed taking a look at the pure numerical values, they may seem low. Nonetheless, comparing these results with those of the baseline model provided by Touchè, they are really good. The main difference is given by the recall metric that in the case of the baseline is always 1, but in our changes from 0.29 to 0.62. Since the recall metric is made by $\frac{tp}{tp+fn}$, at first glance it is very likely that the baseline model is not classifying false negatives. On the hand of precision, given by $\frac{tp}{tp+fp}$, I can see that the values are pretty low, indeed the baseline model seems to predict 1 for every possible class. Anyway, this behaviour is useful to have a reference for the performance metrics. Indeed, the model I proposed seems to be well balanced, not biased to saying always 0 or 1 for the labels, but showing that it has been able to actually learn useful features. Taking a look at the f1-score, I can see that it is always much greater than the one of the baseline classifier, ensuring that a good work toward a good classification has been accomplished.

In addition, as I mentioned before the bert base uncased model returns as output the probability for each value category. However, I needed to convert this probability to a value of 0 and 1 to say whether the sample belongs or not to a class. For this reason I firstly set the threshold to a fixed value of 0.5 and then optimized it to maximize the performances choosing the best threshold. An interesting comparison can be made between the two results, reported in the following tables

	Precision	Recall	F1-score
Self-direct.: action	0.60	0.55	0.57
Achievement	0.68	0.67	0.68
Security: personal	0.47	0.70	0.56
Security: societal	1.00	0.26	0.41
Benevol.: caring	0.60	0.50	0.55
Universal.: concern	0.75	0.43	0.55
micro avg	0.62	0.53	0.57
macro avg	0.68	0.52	0.55
weighted avg	0.70	0.53	0.56
samples avg	0.60	0.56	0.54

Table 4: Performance of our model on zhihu validation dataset with a fixed threshold

After the optimization of the threshold the results became:

	Precision	Recall	F1-score
Self-direct.: action	0.53	0.73	0.62
Achievement	0.67	0.74	0.71
Security: personal	0.44	0.77	0.56
Security: societal	0.91	0.32	0.48
Benevol.: caring	0.44	0.58	0.50
Universal.: concern	0.77	0.48	0.59
micro avg	0.58	0.60	0.59
macro avg	0.63	0.60	0.57
weighted avg	0.66	0.60	0.59
samples avg	0.60	0.63	0.58

Table 5: Performance of our model on zhihu validation dataset with a threshold set to optimize the results

The performance of the model strictly depends on the quality and quantity of data, as there are too few samples, and some configuration of labels occur too rarely. This distribution of occurrences in the labels makes the model not so capable of classifying these rare cases.

It is also interesting to check some model's errors, for instance:

- The following sentence has been wrongly classified, but the error is acceptable: *"Putting all the blame on the game, demonizing the game, and putting the blame on society can ease parents' anxiety, but it does not help at all against I should restrict minors from playing online games."* has been classified as 'Security: personal' and 'Benevolence: caring', but the true labels is only 'Benevolence: caring'. Then it has been classified with recall 1 and precision 0.5.

- Another example is this sentence: *"Extremely serious information asymmetry. It is difficult for individuals to know what information is being collected about them, who is collecting it, and what it is being used for in favor of I should protect our privacy in the Internet age."* that has been classified totally wrong. It has been classified as 'Achievement', 'Security: personal' while the true label is 'Security: societal'. As a result, it has been classified with recall 0 and precision 0 .

6 Conclusion

The expectation before starting this project work was that it should have been easier to solve, but the data turned out to be a serious issue, heavily impacting the performance of any model I tried. The sentences given as textual input were similar one to the other and classified in different value categories by changing only a few words, which has been a great challenge to deal with. Luckily, I have found out that one of the many tasks that can be accomplished with the Bert model is the one on which this challenge is built upon. However, unfortunately I could not overcome the main the problem that relies on the data, as I discussed in the previous section. At least, I managed to soften it by merging the training and validation sets and then evaluatiing the model on the zhihu validation set, but it was still not enough to get a very good model.

Future implementations of this model may deal with the output of the Bert model, given as input to another deep learning model that takes sentences too as input, and performs a classification of them by analyzing the words in the sentences combined with the labels assigned by Bert model.

Finally, I have to mention also a limitation that I faced during the development phase, as I had to use Google Colab and thus to respect the constraints of GPU memory, that forcing us to keep a low memory usage during the training phase. Indeed, another parameter that could be deeper examined is the batch size, that I have been able to test only to 32.

7 Links to external resources

Data source: https://zenodo.org/record/7503506.Y7cwFdWZNz0

Tutorial useful: https://towardsdatascience.com/transformers-for-multilabel-classification-71a1a0daf5e1

Tira performance comparison: https://www.tira.io/task/valueeval-at-semeval-2023-human-value-detection

Challenge Touchè company: https://touche.webis.de/semeval23/touche23-web/index.htmltask-committee

XLNET : https://huggingface.co/docs/transformers/model_doc/xlnet

BERT : https://huggingface.co/bert-base-uncased